- *Definiált, szükséges és eszközspecifikus:*
  Ebben a kategóriában vannak megadva mindazon funciók és paraméterek, amelyeket egy eszköztípus minden implementációjának tartalmaznia kell. Erre a mindenkori operációs rendszerben egy meghatározott interfész áll rendelkezésre. A kamera esetében például minden meghajtónak meg kell tudni válaszolni egy az esetleges autofókusz mechanizmusra vonatkozó kérdést.

- *Definiált, de nem szükséges:*
  Minden eszköztípusra több funkció van definiálva, melyek az összes lehetőség fölös mennyiségét eredményezik. Ezek a funkciók maguk nem lehetnek és nem is lesznek minden kapcsolódó eszköz és eszközmeghajtó által felajánlva. A kamera példájában ilyen funkciók a pozicionálás és a fókusztávolság beállítása, mivel nem minden csatlakoztatott kamera rendelkezik ezekkel a lehetőségekkel. Az interfész az alkalmazáshoz mégis definiálva van. Ezen funkciók alkalmazása, amikor egy implementációban ez nincs támogatva, ismert hibakezeléshez vezet. Az alkalmazás erre beállhat, és így független lesz a csatlakoztatott készülékektől.

- *Nem definiált és nem szükséges:*
  Mindig lesznek új készülékek és különleges fejlesztések, melyeket nem lehetett előre látni. Az operációs rendszer ezek lefedésére egy negyedik kategóriát vezet be.

Ezen kategóriák világos elhatárolása könnyebbé teszi a mindenkori eszköz beillesztését a programozói környezetbe. Az operációs rendszerek mai multimédia-kibővítései az eszközvezérlést még enélkül a kifejezett funkcionális megkülönböztetés nélkül tartalmazzák.

Néhány zárómegjegyzés

A multimédia operációs rendszer processzos adatok feldolgozásához a valósidejű rendszerek területéről származó elveket alkalmaz, amelyek a multimédiaadatok követelményeihez vannak illesztve. A mai operációs rendszer bővítések ezen funkciók közül többet mint eszközmeghajtót, vagy mint járulékos részt valósítanak meg a rendelkezésre álló ütemezőre és más komponensre alapozva. Következő lépésként várható a valósidejű és a nem valósidejű feldolgozás integrálása a rendszer magjában.

A multimédia operációs rendszer összetevői ma a mostani rendszerek összes lehetőségeit tartalmazó alkalmazásoknak, valamint a folyamatos adatok feldolgozásának a területei. A két tartomány közötti interfész segítségével vezérli az alkalmazás a folyamatos médiumok adatfolyamait és eszközeit.

9. Kommunikációs rendszerek

A multimédia kommunikációs rendszerek lehetővé teszik a diszkrét és folyamatos médiumok adatainak cseréjét a számítógépek között. Ez a kommunikáció megfelelő szolgálatokat és protokollokat igényel az audio és video számára.

9.1 Szolgálatok, protokollok, rétegek

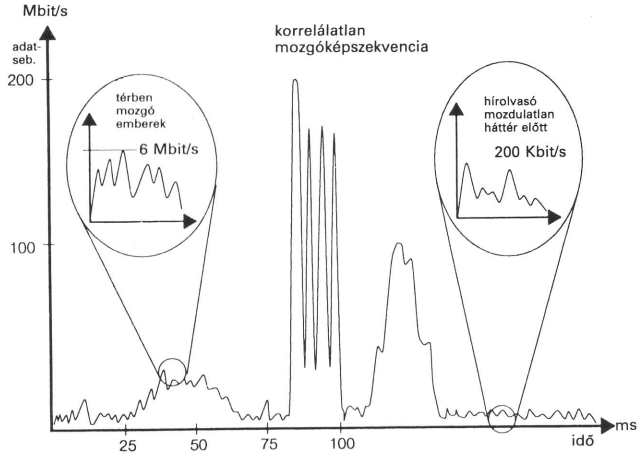
- A *szolgálatok* számos operációt kínálnak a mindenkori alkalmazásnak. A logikailag összefüggő szolgálatokat az OSI referenciamodell különálló rétegekbe foglalja. Az egyes rétegek a közvetlenül felettük lévő rétegek számára biztosítják a szolgálatokat. A szolgálatok írják le az egyes rétegek és szolgálati adatelemeik ***(Service Data Units* = SDUs)** viselkedését. A szolgálatok specifikációja nem tartalmaz utalásokat a konkrét implementációjukra vonatkozóan.

- A *protokollok* nagyszámú szabályból állnak, amelyek az egymással kommunikáló számítógépek adott rétegeinek funkcionális társelemei (entitásai) közt vannak érvényben. A protokollok tartalmazzák a kicserélendő adategységek, a protokoll-adatelemek **(*Protocol Data Units* = PDUs)** formátumát (szintaxis) és jelentését. A szolgálatok megvalósításához a különböző számítógépek funkcionális társelemei működnek együtt.

A szolgálatokkal és protokollokkal szembeni követelmények

A multimédia kommunikáció szolgálataival és protokolljaival szemben – függetlenül az adott rétegtől – a következő követelmények támaszthatók:

- Audio- és videoadatok kérése – alkalmazástól függően – adatfeldolgozás meghatározott időpontig, vagy időintervallumon belül. Az adatátvitelnek – alkalmazástól alkalmazásig – ebben az időszakban kell megtörténnie *(l. 15.1 fejezet, 392. oldal)*.
- A végpontok közti késleltetésnek csekélynek kell lennie. Ez különösen a dialóg üzemmódú alkalmazások esetében – a telefonhoz hasonlóan – indokolt *(l. 15. fejezet 391. oldal)*.
- Az adott időintervallumon belüli adatátvitel biztosításához szükséges garanciákat be kell tartani, többek között a szükséges processzorteljesítmény és a protokoll-feldolgozáshoz szükséges tárkapacitás tekintetében.
- A több médiumot is kezelő kooperatív folyamatok és konferenciarendszerek a kommunikációképes multimédia rendszerek legfőbb alkalmazási területei. A fenti alkalmazások esetében az erőforrások eltékozlásának elkerülése érdekében támogatni kell a multicast összeköttetéseket. Ennek eredményeképpen a konferencia során a küldő entitás gyakrabban váltogatható. Az ilyen jellegű összeköttetések esetében a partnereknek beilleszthetőknek, illetve újabb kapcsolatépítés nélkül a multicast csoportból eltávolíthatónak kell lenniük.
- A szolgálatoknak szinkronizációs mechanizmusokat kell rendelkezésre bocsátaniuk a különböző adatfolyamokhoz, illetve szinkronizációt lehetővé tenni az egyéb komponensek által vezérelt adatfolyamok esetében [Salm89].
- Mivel a kommunikáció az alkalmazott adatátviteli rendszerek kompatibilitásán alapul, az egyes számítógépeken ugyanazokat a protokollokat kell implementálni. Sajnos számos jelenleg létrejövő multimédia kommunikációs rendszer egyedi fejlesztés, és semmilyen várható (de jure vagy de facto) szabványon nem alapul.
- Az audio- és videoadatok kitüntetett vagy garantált átvitele a maradék adatok továbbítását nem akadályozhatja. A diszkrét adatok szokásos módon történő átvitelét továbbra is minden esetben biztosítani kell.
- A különböző alkalmazások, felhasználók és állomások közt a pártatlanság elvének (fairness) továbbra is érvényben kell maradnia.



9–1 ábra
Tipikus információ- és adatsebességek egy jelenet mozgóképkódolása során

- Az audio- és videoadatok információsebessége erősen változó, ami az adatsebesség ingadozásához vezet. A 9–1 ábra három eltérő szituációban szemlélteti a kialakuló adatsebességet, mégpedig korrelálatlan mozgóképek, egy szobában mozgó személyek, illetve a háttérben beszélő hírbemondó esetében. Az erősen ingadozó átlagok ellenére viszonylag alacsony értékeket figyelhetünk meg. A 9–1 ábra adatai a CCIR 601 szerinti kódolást feltételez *(l. 4.6 fejezet, 67. oldal)*.

Az ISO-OSI modell rétegei

Az eddig bemutatott követelmények a kommunikációs rendszerek legkülönfélébb komponenseire vonatkoznak. Ezért a multimédia kommunikációs rendszerekbe való bevezetést az ISO-OSI referenciamodell rétegeinek rövid áttekintésével zárjuk, különös tekintettel a multimédia szolgáltatásokra.

1. A *bitátvitelre szolgáló fizikai réteg* ***(Physical Layer)*** definiálja az egyes bitek fizikai közegen való átviteli módszerét. Itt a lényegesebb kérdések a modulációs módszer, és például a bit szinkronizáció.

   A fizikai közeg a beépített elektromos ármakörök jelátviteli sebességén keresztül a konkrét modulációs módszerrel összefüggésben késleltetést okoz az adattovábbításban, meghatározva az átviteli csatorna lehetséges sávszéles-

ségét is. Az audio- és videoadatok esetében a jelátviteli késleltetést általában minimalizálni kell, és viszonylag magas sávszélességet kell biztosítani.

2. Az *adatkapcsolati réteg* **(Data Link Layer)** biztosítja az egyes üzenetblokkok továbbítását. Itt definiáljuk fizikai közeg hozzáférési protokollját(*Medium Access Control* = MAC), a hibafelismerést, javítást, és a blokkok szinkronizációját. Ebben az összefüggésben különböző hálózatokról beszélünk.
A folytonos adatfolyamok megkövetelik az egyes átviteli szakaszok lefoglalását. A hosszabb késleltetések elkerülése megfelelő hibakezeléssel történhet, az átvitel megismétlésének nem mindig van értelme. Napjaink hálózatai az alacsonyabb hibaarány következtében – különös tekintettel az üvegszál-technológiára – támogatják a multimédia adatok átvitelét. Az üzenetblokkok rögzített hossza hatékony protokoll implementálást tesz lehetővé.

3. A *hálózati réteg* **(Network Layer)** viszi át az üzenetblokkokat (Paket) egyik állomástól a másikig akár több hálózaton át is. Itt az előtérben álló kérdések a címzés, az átvitel, az útvonalválasztás, a hibakezelés, a hálózatmenedzselés (*Congestion management*) és a csomagok szekvencializálása.
A későbbi erőforrás-foglalás igénylése a szolgálat jósági paraméterekkel történik, amelyek az egyes összeköttetési szakaszok folytonos adatfolyamai alapján vannak beállítva. Ez a foglalás a kommunikáló számítógépek közti teljes útvonalra is kiterjeszthető. Ilyenkor többnyire összeköttetésalapú szolgálatra *(Connection-Oriented Service)* van szükség, amely ugyanazt az útvonalat biztosítja az összes csomag számára. Így a végpont-végpont késleltetés csekély elcsúszással és helyes csomagsorrenddel könnyen biztosítható. A hálózatok csatolása, illetve a hálózatok közti átmenet során bizonyos átviteli rendszerek esetén multicast és broadcast üzemmódban bekövetkezhet a csomagok megkettőződése. A megkívánt szolgálatjóság a hálózati rétegben alakítandó ki.

4. A *szállítási réteg* **(Transport Layer)** processz-processz összeköttetést biztosít. Ebben a rétegben történik a nagyméretű csomagok szegmentálása, illetve a fogadó oldalon az eredeti méretre való újraösszeállítása. A hibakezelés processz-processz kommunikáció bázison megy végbe.
A megkívánt szolgálatjósági paramétereknek itt is folytonos médiára kell vonatkozniuk. A hibakezelési módszer többnyire nem lehet az adatátvitel megismétlése, mivel ez túl nagy végpont-végpont késleltetéshez és erős csúszkáláshoz (Jitter) vezet. A szinkronizáció, amely a különböző kapcsolatok LDU-i és ezáltal SDU-i közti időbeli vonatkoztatásokat is lehetővé teszi, a szállítási réteg elválaszthatatlan részének tekinthető.

5. A *viszonyréteg* **(Session Layer)** biztosítja az összeköttetés fennállását a kapcsolatok során. A következő viszonytípusokat különböztetjük meg: pont-pont, multicast (többeknek), multidrop (többektől).
A folytonos adatok átvitelét végző viszonyok esetében nem mindig célszerű a *kapcsolatok* automatizált *újrafelvétele*. Ebben az összefüggésben kidolgozható a multimédia-viszonyoknak megfelelő szemantika definíciója. Egy további aspektus az adatok kódolása. Az alkalmazásoknak a hálózatra való utólagos rákapcsolódás után tudniuk kell például, hogy mikor áll rendelkezésre egy olyan LDU, amelyet *Intraframe* eljárással tömörítettek. A prezentáció csak ilyen adategységgel kezdődhet [Meye92a].

6. A *megjelenítési réteg* **(Presentation Layer)** elvonatkoztat a különféle adatformátumoktól, és egységes formátumot kínál fel. Ehhez megfelelő átalakító rutinokra van szükség. Példa lehet az Intel és Motorola processzorok eltérő számábrázolása.
Az audio- és videoformátumok sokszínűsége adott esetben feltételezi a formátumok konverzióját. Ez a problémakör a kommunikációs komponenseken kívül a folytonos adatokat tartalmazó adathordozók cseréje esetében is előkerül, ezért gyakran a szállítási rétegen kívül kezelik le és vitatják meg.

7. Az *alkalmazási réteg* **(Application Layer)** tartalmazza az alkalmazásspecifikus szolgálatokat. Idetartozik pl. az ftp *(File Transfer Protocol)* állomány átviteli protokoll és az elektronikus posta.
Az audio- és videoadatok kezelése során itt is speciális kérdések merülnek fel. A távoli adatbank egyidejű hozzáférése esetében például a szinkron-megjelenítés a megadott paramétereknek megfelelő valósidejű átvitelt jelent.

A következő szakaszok a multimédia kommunikációs rendszerek problematikáját írják le az ISO-OSI rétegmodell nézőpontjából, *alulról felfelé* haladva. Elsőként a kommunikációshálózatokat tekintjük át, összefoglalva az első rétegre és a közeghozzáférési alrétegre *(Medium Acces Control = MAC)* vonatkozó tudnivalókat. A bemutatás főként az audio- és videoátvitel szempontjából lényeges tudnivalókat tartalmazza. Kiindulópontunk néhány fontosabb hálózat analízise lesz a multimédia sajátságok szempontjából. Az egyes hálózatokra vonatkozó alapismereteket közöljük, amelyek a részletesebb tankönyvek, szabványok, illetve termékismertetők révén egyszerűen elmélyíthetők.

9.2 Hálózatok

A 7.2 fejezet 161. oldalán bemutatotthoz hasonló integrált elosztott multimédia rendszerek az összes média adatait ugyanazon a hálózaton keresztül továbbítják. A kiterjedés szempontjából a következő hálózatokat különböztetjük meg: *Local Are Networks* (LAN), *Metropolitan Are Networks* (MAN), *Wide Are Networks* (WAN). A köztük való átmenet folyamatos.

LAN

A **LAN**-okat néhány kilométeres térbeli kiterjedés és magas adatátviteli sebesség jellemzi. A csatolások (állomások) száma korlátozott, a tipikus felső érték néhány száz (szervezet vagy cég). Több LAN csatolása lehetővé teszi jóval több állomás összekapcsolását is.

MAN

A **MAN**-ok kiterjedése többnyire városléptékű, adatátviteli sebességük általában magasabb, mint a LAN-oké, tipikusan több mint 100 Mbit/s. A hálózati adminisztráció nyilvános vagy privát. A csatlakozások száma ezres nagyságrendű. A MAN-okat gyakran különböző LAN-ok csatolására alkalmazzák.

WAN

A **WAN**-ok nagyobb távolságokra, akár több országra is kiterjedhetnek. Az adatátviteli sebességek azonban mostanáig viszonylag alacsonyabbak, többnyire 2 Mbit/s alattiak. A szélessávú ISDN ebben változást hoz majd. A WAN-okban számos szervezet, cég, magánszemély van összekapcsolva. A hálózatmenedzserek magán vagy nyilvános telekommunikációs vállalatok.

A szokásos, széles körben elterjedt lokális hálózatokat mint a Token Ring, Ethernet, mostanáig nem a valósidejű adatforgalom igényei szerint dolgozták ki. Az újabb LAN-ok, pl. az FDDI II messzemenően figyelembe veszik az audio- és videoátvitel által támasztott követelményeket. A nagytávolságú hálózatok egyik alapfeladata a LAN-okkal szemben az adatátvitelen túl a beszédtovábbítás is. Ezért ezek a telekommunikációs iparból kiinduló hálózatok fel vannak készítve a valósidejű átvitelre. Ezekben természetszerűleg olyan beszédspecifikus sajátságok is integrálva vannak, amelyek video és egyéb adatátvitelre nem alkalmasak. Egy példa erre a viszonylag hosszan tartó kapcsolatfelépítés.

9.3 Ethernet

Az Ethernet a legszélesebb körben elterjedt LAN, amelynek busza a **CSMA/CD** *(Carrier Sense Multiple Access Collision Detect)* módszer szerint működik: az adatátvitel megkezdése előtt az adni kívánó állomás ellenőrzi a hálózat állapotát. Az állomás ezek után akkor kísérelheti meg az adást, ha az adott pillanatban más állomás nem visz át adatot. Ilyen módon minden egyes állomás *szimultán hallgatózhat és adhat.* Ha egyidejűleg több állomás is adatátvitelt kezdeményez, akkor az adók ezt a tényt a buszon megjelenő elrontott adatkeretekből felismerik. Az állomások ezután egy véletlenszerűen megválasztott időtartam kivárása után kezdik újra az adást.

Felhasználás audio- és videoátviteli célokra

A folytonos adatok kommunikációja megköveteli egy maximális végpont-végpont késleltetés betartását. Az Ethernet elvileg nem képes ezt garantálni. Mégis több lehetőség adódik ennek a hálózattípusnak az audio- és videoadatok átvitelre való kihasználására:

1. Az első esetben a **folytonos adatokat ugyanúgy tekintjük, mint bármilyen egyéb adatot.** Ha a maximális terhelést meghatározott keretek közé szorítjuk, akkor a túl későn érkező adatok száma rendkívül csekély. Manapság ez a leggyakoribb megoldás, mivel nem kíván többletráfordítást. A fellépő hibák miatt azonban nem kielégítő.

2. A túl nagy hálózatterhelés esetén keletkező hiba a folytonos média adatforgalma esetében az **adatforgalom dinamikus szabályozásával** elkerülhető. Ebben az esetben nagy hálózatterhelésnél a folytonos média adatfolyamsebességét dinamikusan csökkentjük. Ez skálázható kódolással lehetséges *(l. 5.7.4 fejezet, 114. oldal)*. Azonban még így sem zárhatók ki a hibák az adatátvitel során.

3. További lehetőség, hogy egy Ethernet hálózatot kizárólag folytonos adatok átvitelére szánunk. Ebben az esetben egy alkalmas járulékos protokollt kell betartani. Ez a megoldás azonban **két független hálózat** meglétét követeli meg az összes multimédia-adatokkal kommunikáló számítógép között: egy hálózat ezen folytonos adatok számára, és egy másik a többi maradék adathoz. Ez a kiegészítés kísérleti rendszerek esetében értelmes, azonban a többletráfordítás miatt nem képez reális megoldást.

4. Egy rendkívül pragmatikus megoldás az installált hálózatkonfigurációból adódik: az Ethernet kábeleket az épületekben többnyire nem buszrendszerben fektetik le, hanem minden állomástól csillagstruktúrában vezetnek a kábelek

egy központi terembe. A kábeleket itt kapcsolják össze Ethernet bu
A busszá való konfigurálás helyett azonban minden állomás egyfajta köz
kapcsolóra ***(Hub)*** is rákötheti, ilyenkor minden állomás saját Ethernet-k
lel van összekötve a *Hub*bal. Ilyen módon minden egyes állomásnak rend
zésére áll a teljes 10 Mbit/s sávszélesség, és egy új multimédiaképes h
kiépítése nem minden körülmények között szükséges. Az egyetlen jár
költséget ilyenkor a *Hub* adja. Ez a megoldás azon a feltételezésen a
hogy az átvitelhez szükséges adatsebesség 10 Mbit/s alatt van. Az egyéb a
vitel azonban még ilyenkor is megzavarhatja az audio- és videoadatok tov
tását.

5. Az **izokrón Ethernet** esetében a folytonos adatokat ugyanazon a kábel
modulálják. Ilyen módon ugyanazon a fizikai hálózaton vihetők át a disz
folyamatos adatok is.

6. Ha az állomások minden adatátviteli formánál betartanak egy **kiegész**
tokollt, akkor a különféle média integrált átvitele ugyanazon **hálóza**
elérhető. Járulékos protokollt valósíthatunk meg pl. úgy, hogy **eltérő**
tású átviteleket engedélyezünk. A következő szakaszok ezzel a **mód**
foglalkoznak.

Prioritáson alapuló adatátvitel

A következő módszer lehetővé teszi Etherneten való változó prioritású ad
definiálását [Cour92a]. Ekkor tekintettel a *multimédiára* a folytonos adat
magasabb prioritást rendelünk, mint a diszkrét médiumok továbbításá
módon figyelembe vehetjük a valósidejűségre vonatkozó követelmények

Míg az Ethernet-nél minden állomás – mindaddig, amíg a hálózaton
folyik adat – tud küldeni, a prioritások tekintetbevételénél ki kell várni
definiált időtartamot. Ezen időszakasz után fogja az állomás szabad **hálóz**
adatait átvinni. A magasabb prioritású állomáshoz kisebb késleltetési idő
delve, mint az alacsonyabb prioritásúhoz. Ha egy magasabb prioritású **állo**
kezd, az alacsonyabb prioritásúnak várnia kell, mivel a LAN ezen **állom**
csonyabb prioritású) állapotának lekérdezésekor már foglalt. Minden **áll**
rendelnek az átvitel megkezdéséhez lehetséges legkorábbi időpont
olyan határértéket is, ami szerint egy meghatározott időablakon belül az
meg kell kezdődnie. Ha ez az ablak lefutott, akkor még a magasabb prior
másnak is várnia kell a következő *foglalt-szabad* átmenetre. Az állomás
lói ehhez az időponthoz lesznek visszaállítva, és a mindenkori ablakra fog
A *foglalt-szabad* átmenetet egy adatátvitel vége határozza meg.

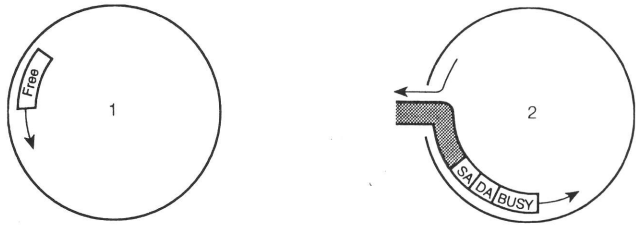
A szomszédos prioritások ablakainak időbeli távolsága a leghoss
idő kétszerese: az IEEE 802.3-nál ez az átviteli idő maximum 51 mikros
minimum 3 mikroszekundum [Cour92a]. Ez az eljárás arra van ala

fordul elő *foglalt-szabad* átmenet. Ezért egy maximális beállási időt állapí-
meg, amely hosszabb, mint a legalacsonyabb prioritás ablakának végei
távolság. Ha ez a beállási idő *foglalt* állapotba való átmenet nélkül telik el,
valamelyik állomás információmentes üzenetet küld a buszon keresztül.
vég végén minden számláló visszaáll és újraindul.
a prioritásokat multimédia rendszerben alkalmazzák, az egész aszinkron
mat alacsony prioritással kell azonosítani. Ily módon nem lehet a rend-
a tetszőleges meglevő Ethernet megvalósítást integrálni; minden állomás-
kell magát ezen konvenciókhoz.
juk továbbá a folytonos médiumok különböző összeköttetéseinek
prioritásokkal való kezelését. Így elkerülhetők az ütközések, amikor
másnak többször meg kell kísérelni a hozzáférést. Pontos sorrendet állíta-
adatok és a teljes korlátos adatátviteli igény periodicitása miatt a külön-
tásoknak nincsen hatásuk az alkalmazás valósidejűségére. Mindazonál-
bb prioritással kisebb közepes végpontok közötti késleltetés érhető el.
juk, hogy minden interaktív alkalmazás folytonos adatai magasabb
kerüljenek továbbításra, mint az elosztási és lekérdezési alkalmazások

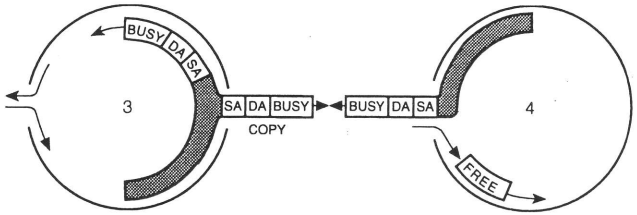
agy erőforrásigény elkerülésére még korlátozni kell a folytonos médiu-
bb prioritás) összeköttetéseinek maximális számát is, a megfelelő vég-
késleltetés és az adatsebesség függvényében.

Token Ring

egy lokális hálózat (LAN) 4 vagy 16 Mbit/s adatátviteli sebességgel.
hozzáférési algoritmuson alapul, s csak az ún. *Token (adási jog)*
küldheti el a mindenkori adó az adatait. Az összes állomás egy
gyűrűvé kapcsolódik össze.



9–2 ábra
Adatok küldése Token Ringen keresztül, az első két fázis, az elv bemutatása a szabványból származó jelölésekkel



9–3 ábra
Adatok küldése Token Ringen keresztül, az utolsó két fázis

Minden állomás veszi, elolvassa és elküldi a gyűrűn keringő adatokat. Az elolvasás után általában továbbadja a szomszédos állomásnak a mindenkori üzeneteket (l. a 9–2 és 9–3 ábrákat). Minden csomag tartalmazza a küldő címét (SA) és a cél címét (DA). Ha a célcím megegyezik a saját címmel, akkor az üzenet a lokális memóriába másolódik, amit a lokális LLC-komponensekkel *(Logical Link Control)* közölnek, és egy nyugtázó- vagy hibamező is megváltoztatásra kerül. Végül az üzenetet továbbküldik a szomszédos állomásra. A küldő állomás eltávolítja az általa küldött üzenetet, és interpretálja a nyugtázó mezőket. Az adási jog megkapásához a mindenkori állomásnak meg kell kapnia a *Tokent.* Ezt az állomások közül egy megfelelő prioritású előre foglalhatja le, így csak a legnagyobb foglalt prioritású állomás kapja meg az adási jogot. Ezek a prioritások a csomagok rögzített maximális megfordulási idejével együtt lehetővé teszik a folytonos médiumok garantált adatátvitelét.

Prioritások

A definiált prioritások növekvő fontosság szerint a következőek:

9–1 táblázat
Prioritások a Token Ringnél

Prioritás	Felhasználás
0	szabadon felhasználható, a legtöbb alkalmazás használja
1–3	szabadon felhasználható
4	a Bridge-ek használják
5,6	foglalt, de nem használt
7	a gyűrű adminisztrációjára használt

Felhasználás audio- és videoátvitelre

Itt is – mint az Ethernetnél – megfigyelhetők az integrált kommunikáció különböző változatai és néhány, az Ethernetnél leírt változat a Token Ringre is átvihető. Így elvégezhetjük a kódolás léptékezését vagy definiálhatunk egy izoszinkron Token Ringet. Még a következő lehetőségek állnak rendelkezésre:

1. A már meglévő különböző prioritású átvitel lehetővé teszi ennek a hálózatnak a használatát az eddigi komponensek bármilyen változtatása nélkül is. **A folytonos média adatai** mind ugyanazt, a diszkrét médiánál **magasabb prioritást kap**ja. A folytonos adatok túl magas részaránya esetén azonban az adatcsomagok nemkívánt késleltetése lép fel.

2. A folytonos adatfolyamokon belül egy durva fontossági megkülönböztetést is tehetünk: **Minden audiokapcsolat egy magasabb prioritási fokozatot kap,** mint az összes többi folytonos adat. Ehhez két magas prioritási fokozat szükséges, mint ahogy például rendelkezésre áll az 5 és a 6. Ez az elv saját vizsgálatokban sikeresnek bizonyult, azonban tiszta audiokapcsolatok nagy száma esetén hibákhoz vezetett.

3. Lehetséges az **audio és a video forgalomra egy közös magasabb prioritás engedélyezése,** és erre a kapcsolatra egy járulékos erőforrás-foglalás bevezetése.

   Ez lehet egy statikus, „a 8 multimédiaképes állomás egyenként mindig csak maximum 1,2 Mbit/s-mal adhat folytonos média adatokat" mértéke szerinti egyszeri elosztás. Ez az elv csak kis környezetekben és a folytonos médiumok alacsony adatforgalma mellett alkalmazható.

   A kapcsolat felépítésénél alternatívaként minden állomás egy központhoz fordul, hogy itt hozzájusson a korlátos erőforráshoz. Végül a hozzájárulásnak a mértéke szerint viszik át az adatokat. Ez a központi sávszélesség-kezelő rendelkezik minden szükséges adattal, hogy garanciákat adhasson ki. Ilyen központi hálózatadminisztrációs szolgálatokra már léteznek alkalmas protokollok, amelyek a jövőben a hálózati határokon túlmutató összehangolást is lehe-

tővé tesznek. Mindenesetre ebben az esetben a kapcsolat felépítése egy nem elhanyagolható időt vesz igénybe.

Ez a problematika részben megfelel az operációs rendszerek területén az erőforrásfoglalás problematikájának és ott már tárgyaltuk *(l. 172. oldal, 8.2 fejezet)*. Az első, nem megszakítható tervezési eljárásra vonatkozó tervezhetőségi tesztet a Token Ringre a [NaVo92a]-ban mutatták be. Ez érvényes az erőforrásokra az *operációs rendszerek* területén, mint ahogy a hálózatok sávszélességére is *(l. 198. oldal, 8.3.8 fejezet)*.

Decentralizált erőforrás-foglalás

Egy elosztott sávszélesség-kezelés lényegesen gyorsabban tud erőforrást az adott kapcsolat rendelkezésére bocsátani. A következőkben egy ilyen, a [Stei92b]-ből származó időoptimalizált lehetőséget írunk le röviden.

Minden multimédia állomás rendelkezik egy belső ***Available Resource Table „Art"*** táblázattal. Ebben a táblázatban bejegyzések vannak a hálózat rendelkezésre álló sávszélességéről a már futó audio és video kapcsolatokra felhasznált sávszélességgel együtt. Emellett például az összes állomás inicializálásához a teljes rendelkezésre álló sávszélességet a valódi 80%-ára csökkentik (ami 3,2 ill. 12,8 Mbit/s-t jelent a Token Ringnél). A maradék 20% fedezi a gyűrű adminisztrációjának forgalmát (ez tipikusan kb. 3%), ami a legmagasabb prioritással fut. Azonkívül így mindig marad kapacitás más médiumra. Ez a 80% egy tapasztalati érték, ami a más adottságokhoz hozzáilleszthető.

Minden multimédia állomás üzemi adminisztrátorai a Token Ringben egy közös címzésű csoporthoz tartoznak. Ezáltal minden ilyen üzemi adminisztrátor üzeneteket tud az összes többinek küldeni. Ehhez a Token Ringben például *funkcionális címeket* alkalmazhatnak.

Egy **kapcsolat felépítésé**nek kezdetén a helyi erőforrás-kezelő összehasonlítja a szükséges kapacitást az éppen rendelkezésre állóval. Ha az *Art*-ban rögzített és éppen rendelkezésre álló sávszélesség kisebb, mint a megkívánt, akkor a kapcsolati kérelem egy megfelelő utalással azonnal elutasításra kerül. Kielégítő, *Art*-ban nyilvántartott kapacitás esetén a foglalási kérés az összes erőforrás-kezelő csoportjához kerül továbbításra.

Példaként egy 1,41 Mbit/s kérést küldtek el a csoport összes tagjához. Legyen az ebben az időben rendelkezésre álló kapacitás 10 Mbit/s. Első lépésben a kérő állomás hozzáigazítja a helyi *Art*-ját a még rendelkezésre álló kapacitáshoz. Ebben a példában ez az érték 10 Mbit/s-ről 8,59 Mbit/s-ra csökken. A következő lépésben a kérő állomás elküldi az 1 410 000-es kapacitás foglalási kérését a csoportnak. Ezt az információt az összes többi állomás az *Art* aktualizálására használja.

A csomag (*Frame* a Token Ring-en) a foglalási kérelemmel egy kör után visszatér a kérő állomáshoz. Ezzel ez az állomás tudja, hogy a csoport minden tagját tájékoztatták és azok a helyi *Art*-jaikat hozzáillesztették.

Ha a foglalási kérelem kiküldése és a kérés lezárása között a kérő állomás *Art*-ja egy negatív értékre állítódott, a foglalási kérelem elutasításra kerül, és erről a csoport összes tagját tájékoztatják. Különben a foglalási kérelmet elfogadják és a tulajdonképpeni adatátvitel magas prioritással megkezdődhet. Az elutasításnak ez a (a saját negatív értéken alapuló) lehetősége az Early-Token-Ring eljárás miatt egyidejű kapcsolatkérelmek esetén különböző állomásoknál léphet fel. Egy ilyen **ütközés** esetében annak lehetővé tétele, hogy egy állomás a rendelkezésre álló sávszélességhez hozzáférjen, a hozzáférés Ethernet-hez hasonló minta szerint történhet. Engedjünk meg például maximum 3 hozzáférést, és osszuk el ezeket egy meghatározott stochasztika alapján. Így egy ütközéskövetkezmény valószínűségét drasztikusan csökkentettük.

A **kapcsolat felépítésé**nél minden állomást tájékoztatni kell; ezek az *Art*-jaikat ennek megfelelően visszaállítják. A fenti példában például a teljes csoportnak +1 410 000 kerülne elküldésre.

Egy **új állomás gyűrűbeli inicializálása** a többi résztvevő összes rendelkezésre álló *Art*-jának lekérdezésével történik. Ha ezek az állomások (egy adott időn belül) az adataikat visszaküldték és ezek ugyanazokat az értékeket tartalmazzák, akkor az új állomást ezzel az értékkel inicializálják. Az inkonzisztenciákat egy járulékos adminisztrációs protokoll kutatja fel és szünteti meg.

Itt **jegyezzük meg,** hogy ez az eljárás változó bitsebességű adatfolyamokra is alkalmazható. Ekkor a foglaláshoz legtöbbször a maximális értéket adják meg. Ekkor nem tűnik el erőforrás-kapacitás, mert az összes diszkrét adat a *lyukakat* kitölti.

Vegyük figyelembe, hogy ez az eljárás nem csak a sávszélesség foglalására alkalmas, hanem egy gyűrűn belül minden más szükséges erőforrás foglalására is alkalmazhatjuk. Ez lehet például a Token Ringben rendelkezésre álló korlátozott számú *funkcionális cím* is.

Az eljárás abból indul ki, hogy minden multimédia állomás csak a foglalásnál megadott kapacitást használja fel. A megígért kapacitás betartását felügyelő komponenst ennél külön kell megvalósítani.

Bithosszok

A lényeges, szűkös erőforrás a hálózatban a sávszélesség, amelyet a fenti eljárással oszthatunk el. A sávszélesség növelése nagyobb mennyiségű folytonos adatfolyamot tesz lehetővé. A mindenkori hozzáférési protokoll egy lényeges hatékonysági kritériuma a bithossz, a hálózat fizikai kiterjedése értelmében. A következő példával megvilágítva: 100 Mbit/s-nél egy bitnek az üvegszálban a következő kiterjedése van:

$$\text{Hossz} = \frac{\text{Terjedésisebesség az üvegszálban}}{\text{Adatsebesség}}$$

$$\text{Hossz} = \frac{2 \times 10^8 \text{ m/s}}{100 \times 10^6 \text{ bit/s}} = 2 \text{ m/bit}$$

Ezzel ellentétben egy 64 kbit/s adatátviteli-sebességű rézkábelben minden bitnek

$$\text{Hossz} = \frac{\text{Terjedésisebesség a rézben}}{\text{Adatsebesség}}$$

$$\text{Hossz} = \frac{2{,}5 \times 10^8 \text{ m/s}}{64 \times 10^3 \text{ bit/s}} = 3{,}9 \text{ km/bit}$$

kiterjedése van.

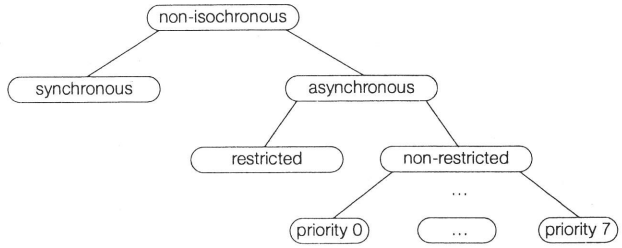
Itt jegyezzük meg, hogy minden állomás a továbbításnál belül legalább 64 bitet pufferel. Így magasabb adatátviteli sebességnél több frame lehet egyidejűleg egy gyűrűn. Ez a nagyobb adatsebességre való átmenet az ***Early-Token-Release elv*** bevezetéséhez vezetett. Itt a gyűrűn egymás mögött több csomag is lehet. Az elküldött adatok után a küldő állomás a *tokent* újra a gyűrűre helyezi.

9.5 FDDI

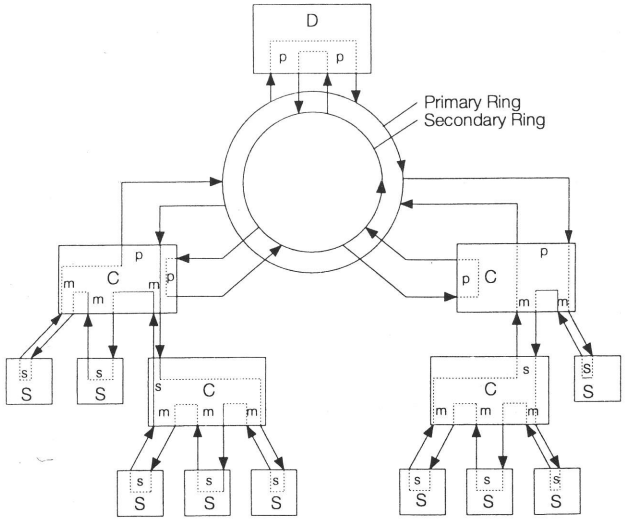
A ***Fiber Distributed Data Interface*** (FDDI) az alkalmazott IEEE 802.5 protokoll szempontjából a Token Ring továbbfejlesztésének is tekinthető. A szabványosítás az *American National Standards Institut* (ANSI) X3T9.5 csoportjában kezdődött 1982-ben. A lényeges döntéseket 1988-ban hozták, azóta léteznek az első implementációk. Az FDDI alapja egy csekély csillapítással és hibagyakorisággal rendelkező fényvezető kábel. Ilyen módon az aktív állomások közti távolság megnövelhető.

Míg a Token Ring sebessége 4 vagy 16 Mbit/s, addig az FDDI 100 Mbit/s-mal működik. Az FDDI max. 500 állomást szolgálhat ki; a Token Ringre tipikusan maximum 50–250 állomást kapcsolnak. A dupla kábelezésű FDDI gyűrű hossza a szabvány szerint maximum 100 km lehet, és az egyes állomások közti távolság nem haladhatja meg a 2 km-t. Ezek az értékek világosan jelzik az FDDI előnyét az elődökkel szemben.

Az FDDI multimédia kommunikációt lehetővé tevő átviteli lehetőségeit a 223. oldal 9–4 ábrája alapján értelmezhetjük. A szinkron mód lehetővé teszi a sávszélesség foglalást, az aszinkron mód a Token-Ring-protokollhoz hasonlóan viselkedik, mindazonáltal ma sok megvalósítás csak az aszinkron módot támogatja. A különböző módok ezen fontos áttekintése előtt a következő szakaszokban röviden ismertetjük még a topológiát és a komponenseket, a részleteket meg lehet találni a szabványban vagy a [MiKh93a]-ban is.



9–4 ábra
Kommunikációs lehetőségek FDDI-ben. Az adatátviteli módok áttekintése



9–5 ábra
Lehetőség különböző FDDI-állomások összekapcsolására: egy duális gyűrű több fával

Topológia

A topológia alapvető jellemzője a két egymással szemben forgó gyűrű. Az első (primer) gyűrű adatátvitelre szolgál, a második a szekunder a hibatűrés növelésére. Az egyes állomások csatolhatók – de nem kell őket csatolni – mindkét gyűrűhöz.

Az FDDI-nél különböző állomástípusok különböztethetők meg

- A ***Dual Attachment Station***-t (kettős csatolású állomás) *A osztályú* állomásnak is hívják. Ezt vagy közvetlenül a fő gyűrűhöz *(Trunk Ring)*, vagy koncentrátoron át a primer és szekunder gyűrűhöz van csatolva.
- A ***Single Attachment Station*** (egyszeres csatolású állomás) (*B osztályú* állomás) a koncentrátoron át csak a primer gyűrűhöz csatlakozik. A koncentrátor-állomás több mint két állomáshoz csatolható, és mindig össze van kötve a primer és a szekunder gyűrűvel. A 9–5 ábra a 223. oldalon komplett konfigurációt mutat a különböző állomásokkal. Ez egy duális gyűrű fákkal. Hiba esetén az FDDI-gyűrűt újra konfigurálják. Így egyes állomások transzparens átkapcsoló funkciót kapnak.

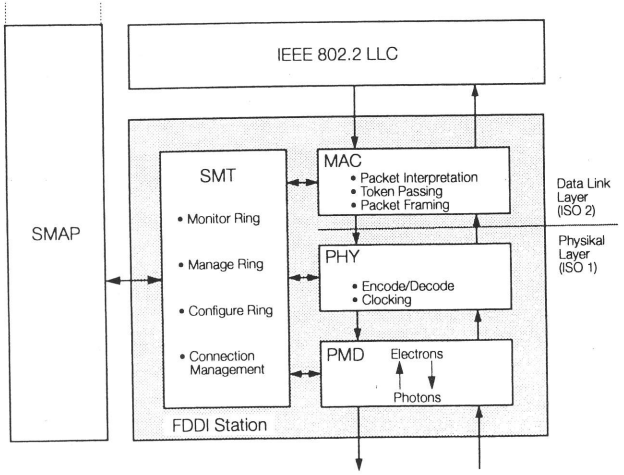
Komponensek

Az FDDI a 225. oldalon a 9–6 ábrán ábrázolt komponenseket különbözteti meg:

- PHY *(Physical Layer Protocol)* (fizikai réteg protokoll) az „ISO 9314-1 Information Processing Systems: Fiber Distributed Data Interface – Part 1: Token Ring Physical Protocol" szerint.
- PMD *(Physical Layer Medium Dependent)* (fizikai réteg, médiumfüggő) az „ISO 9314-3 Information Processing Systems: Fiber Distributed Data Interface – Part 1: Token Ring Physical Layer, Medium Dependent" szerint.
- SMT *(Station Management)*
  A gyűrű kezelési funkcióit definiálja, az „ANSI Preliminary Draft Proposal American National Standard X3T9.5/84-49 Rev. 6.2, FDDI Station Management" szerint.
- A hozzáférés a hálózathoz egy
  MAC *(Media Access Control)* komponensen keresztül történik
  az „ISO 9314-2 Information Processing Systems: Fiber Distributed Data Interface – Part 2: Token Ring Media Access Control" szerint.

A fizikai szint ***(PMD, PHY)*** végzi az illesztést az optikai szálra. Itt ma 62,5 µm átmérőjű multimodusú szálakból, vagy 125 µm átmérőjű monomodusú szálakból

indulnak ki. Az adó egy 1,320 nm hullámhosszal rendelkező LED. Ezen a legalsó szinten *Non Return to Zero Inverted* (NRZI)-en keresztül 4-ről 5 bitre kódolás található. Ezzel a fizikai adatsebesség 125 Mbit/s-ra nő.



9–6 ábra
Egy FDDI-állomás: az elemek a szabványban megadott jelölésekkel

Az ***SMT*** feladata a csatolt állomásoknak és magának a hálózatnak a vezérlése, felügyelése és kezelése. Ide számítanak az állomásokhoz kapcsolódva az inicializálás, aktivizálás, a szolgáltatott teljesítmények felügyelete és a hibakezelés. A hálózat számára ellátott lényeges funkciók a címzés, a sávszélesség foglalás és a konfiguráció.

A ***MAC*** LAN-hozzáférési komponens dönti el, hogy melyik állomás kap hozzáférést a gyűrűhöz: címfelismerést hajt végre és frame-ket ismétel, vesz ki vagy tesz be a hálózatba. Az FDDI-ben a csomagot ***Frame***-nek nevezik. Ezen csomagok hossza változhat, de sosem lépi túl a 4500 byte értéket. A címzés a pont-pont kommunikáció mellett a Multicastot és a Broadcastot is megengedi.

Timed Token Rotation Protocol

A Timed Token Rotation Protocol (időzített token rotációs protokoll) az ***FDDI-LAN hozzáférési protokoll***-ja. Csak akkor lehet egy FDDI-frame-t elküldeni, ha az állomás megkapta a tokent. Adáskor az adni akaró állomás leveszi a tokent, elküld egy vagy több frame-t, és legvégül visszahelyezi a tokent a hálózatra. Így ugyanab-

ban a token-futamban további állomások is küldhetnek adatot. Az adási jog tokennen keresztüli birtoklása mellett az állomásnak más feltételeket is teljesítenie kell ahhoz, hogy adatokat küldhessen. Ezeket a továbbiakban mutatjuk be.

A protokoll számára bevezetik az ún. ***Target Token Rotation Time (TTRT)***-t. Ez az érték a csomag kívánt tipikus gyűrűátfutási idejét fejezi ki, ami a gyűrű inicializálásakor kerül megállapításra az összes állomás SMT komponenseinek lekérdezése során, és minden állomás megjegyzi magának. A TTRT-nek az FDDI specifikáció szerint 4 ms-nél nagyobbnak és 165 ms-nél kisebbnek kell lennie. Tipikus TTRT érték a cca. 50 ms. Ilyen nagyságrendű értékekkel lehet pl. találkozni nagy kihasználtságú hálózatnál 75 csatlakoztatott állomással és cca. 30 km üvegszállal.

Token Rotation Time

Mindegyik állomás folyamatosan méri a tokenek valódi átfutási idejét és **Token Rotation Time (TRT)** néven tárolja. Így a TRT a mindenkori mért utolsó időtartamot adja, amely az állomásra vonatkozó utolsó token átfutáshoz volt szükséges. A következő szabályok érvényesek:

- **Aszinkron forgalom** csak akkor bonyolítható, ha a hálózaton van még szabad kapacitás. Erre kritériumot ad a mindenkori TRT és TTRT összevetése. Egy állomás mindaddig küldhet aszinkron adatokat, amíg igaz a következő összefüggés: **TRT < TTRT.**

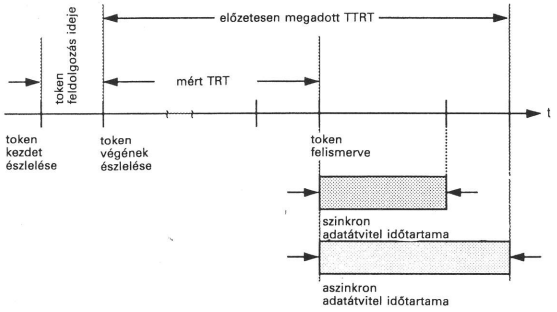
  Így a LAN-on legfeljebb a TTRT időtartamáig léphet fel aszinkron adatátvitel.

- A **szinkron adatforgalom** garantált adatáteresztést jelent. SMT-eljárások során kerül megadásra a szinkron átvitel sávszélessége az egyes állomásoknak, melyet ezeknél szinkron allokációnak (Synchron Allocation = SA) nevezünk. A szinkron adatkapcsolatok időtartamainak összege nem haladhatja meg a TTRT értéket. Ezt az SMT komponens vizsgálja meg, és egyúttal biztosítja is. Az állomások a token megkapását követően mindig küldhetnek szinkron forgalmat a megadott SA-nak megfelelően. A LAN összességében maximálisan a TTRT időtartamáig lesz szinkron forgalmazásra lefoglalva.

A **TRT**-nek ez az értéke a 9–4 ábrán a 223. oldalon szemléltetett szinkron és aszinkron átvitelek alapján legfeljebb a **TTRT** kétszerese lehet. Az 50 ms-os TTRT példájánál az átfutási idő maximuma így 100 ms-ra lesz korlátozva. A TRT emellett mérőszámot ad a gyűrű pillanatnyi terhelésére is.

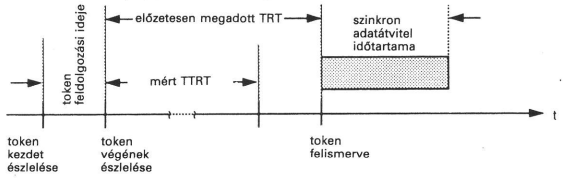
Az aszinkron forgalom a 223. oldalon a 9–4 ábra szerint a ***Restricted Token***nel is működik. Ekkor a teljes aszinkron sávszélesség két állomás párbeszédére kerül lefoglalásra: Az adni kívánó állomás tájékoztatja a vevőt a párbeszéd iránti kívánságáról. Ez normál *(Not Restricted)* aszinkron módon megy végbe. Az adni kívánó állomás egyúttal átviszi a megfelelő adatokat mint járulékos csomagokat a *Restricted Token*-nel együtt. Ekkor más állomás nem használhatja az aszinkron sávot. A vevőállomás értelemszerűen folytathatja a párbeszédet. A párbeszédet a kezdeményező állomás a *Restricted Token* levételével és egy *Non Restricted Token* hálózatra való visszahelyezésével fejezi be. A ***Non Restricted Token***nel való aszinkron adatforgalomnál a Token Ring-hez hasonlóan szintén 8 prioritást lehet megkülönböztetni.

Az audio- és videoadatok átvitele így előnyösebben hajtható végre szinkron módon. Azonban nem szabad elhanyagolni a sávszélességfoglalás idejét. Az adatátvitelnél a TRT és TTRT viszonyok között a következő esetek fordulhatnak elő:



9–7 ábra
Szinkron és aszinkron forgalom az FDDI-n TRT < TTRT esetben

1. A 9–7 ábrára a következő összefüggés érvényes:
   TRT < TTRT.
   Ezzel az állomás mindaddig adni tud, amíg a helyi TRT-számláló a TTRT értéket túl nem lépi. Emellett még a szinkron adatait is átveheti.



9–8 ábra
Szinkron forgalom az FIDDI-n TRT > TTRT esetben

2. A 9–8 ábrára érvényes a következő összefüggés:
   TRT > TTRT,
   ezért ez az állomás már csak szinkron adatait tudja elküldeni.

Végül álljon itt ***példaként*** a Timed Token Rotation Protocol. Legyen a TTRT-nek 8 időegysége. Az FDDI-re legyen 3 állomás csatolva. Mindegyik állomásnak egy időegysége van szinkron adatforgalomra (SA) lefoglalva és ezt el is küldi. Minden állomás annyi nemkorlátozott aszinkron forgalmat akarna lebonyolítani, amennyit a hálózat megenged. A következő táblázatban láthatók a TRT-számlálók mindenkori értékei állomásonként, az átviendő egységekkel együtt. Az 1. állomás után a 2. kapja meg az adatokat, utána a 3., majd ismét az 1. Ebből a következő összjáték adódik, ami egyúttal szemlélteti az aszinkron forgalom megfelelő elosztását:

9–2 táblázat
Példa – TRT-számlálók szekciónként, forgalomfajták szerint

1. állomás			2. állomás			3. állomás		
TRT	syn	asyn	TRT	syn	asyn	TRT	syn	asyn
0	1	8	9	1		10	1	
11	1		3	1	5	8	1	
A rendszer „beindult", kezdődik az első ciklus								
8	1		8	1		3	1	5
8	1		8	1		8	1	
3	1	5	8	1		8	1	
8	1		3	1	5	8	1	
Újra kezdődik a ciklus								
8	1		8	1		3	1	5

Az FDDI további ismérvei

Az FDDI támogatja a csoportcímzés minden követelményét, amelyek a kooperatív multimédia alkalmazásokhoz szükségesek.

A különböző adatfolyamok közötti **szinkronizáció** nem része a hálózatnak és ezt külön kell megoldani. Különösen a szinkron és az aszinkron módon átvitelre kerülő adatokra kell ügyelni. Egy, az adónál meglévő időbeli viszony a Timed Token Rotation Protocol miatt a vevőnél már nem feltétlenül fog fennállni.

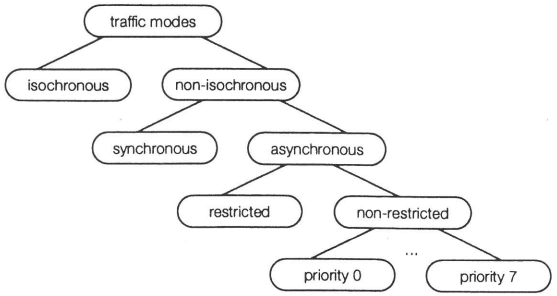
Az alkalmazott **csomagméret** közvetlenül befolyásolhatja az adatok két alkalmazás közötti késleltetését, ha az átviendő adatok kis LDU-k formájában állnak rendelkezésre. 8 KHz-es, 64 Kbit/s átviteli sebességű beszédmintavételezés esetén mindig annyi hangcsomagot kell összegyűjteni, amíg egy FDDI-csomag komplett nem lesz. Itt egy kisebb FDDI csomagméret a kívánatos.

Sajnos sok FDDI megvalósítás nem támogatja a szinkron üzemmódot, ami jól használható folyamatos médiumok adatainak átvitelére. Aszinkron üzemmódban ugyanaz az eljárás alkalmazható, mint a Token Ringnél. Mindenesetre a szinkron adatátvitel fennállása esetén az aszinkron adatátvitel folyamatossága csorbát szenved. Így, mint a Token Ringnél, csak akkor adhatók garanciák, ha egyetlen állomás sem használja a szinkron üzemmódot.

Ha egy időpontban csak két állomás visz át folytonos adatokat, akkor arra ***Restricted Tokennel*** az aszinkron üzemmód is alkalmazható. Ez kisebb végpont-végpont késleltetéshez vezet, viszont megakadályoz minden további aszinkron forgalmat a lokális hálózaton, így ez az üzemmód kizárólag folytonos médiumok adatainak átvitelére használható fel.

FDDI II

Szinkron üzemmódok használatánál figyelembe kell venni a szinkron sávszélesség foglalásának nem elhanyagolható idejét. Itt jegyezzük meg, hogy a szinkron üzemmód egy sávszélességet egy maximális késleltetéssel garantál. Ez az érték sohasem lépi át a TTRT értékének a dupláját. Mindenesetre a szinkron üzemmódban az adatok a vevőhöz lényegesen korábban is megérkezhetnek. A késleltetés ingadozásai a gyűrű leterheltségétől függenek, így a késleltetést a maximális értékére kell venni, ez a 100 ms tartományban található és így párbeszédes alkalmazások számára nem jelentéktelen. Ehhez jön még az, hogy a túl korán érkező adatokat átmenetileg tárolni kell, amihez a vevőnél legalább egy, egy TTRT időtartamára elegendő puffer szükséges. Ezért az FDDI II-vel egy pótlólagos izoszinkron üzemmódot vezettek be (l. 9–9 ábra, 230. oldal). A [ZoKa91]-ben azonban az eredeti FDDI hangadatok átvitele kerül felhasználásra.



9–9 ábra
Kommunikációs lehetőségek az FDDI II-ben: Az adatátviteli módok áttekintése az irodalomból ismert fogalmakkal

Az FDDI II tervezése 1984-ben kezdődött, és az FDDI kiegészítésére kellett volna szolgálnia. Lényeges szempont egy garantált sávszélesség izoszinkron adatfolyammal. Ezért definiáltak 16 ***Wide Band Channels***-t (WBC) egyenként 6,144 Mbit/s-mal. Ez az érték adja a legkisebb közös többszörösét az amerikai (négyszeres) és az európai (háromszoros) keskenysávú ISDN elsődleges adatátviteli sebességnek [TeGv90].

WBC-k egyenként vagy kombináltan allokálhatók és átviteli csatornaként használhatók duplex kapcsolatokban 2 és több állomás között. Egy WBC (nem a részei) vagy ***izoszinkron*** (FDDI II) vagy FDDI típusú lehet. Egy izoszinkron használt WBC fel tudja osztani a rendelkezésre álló adatátviteli sebességet a 8 kbit/s többszöröseire mint virtuális kapcsolatokra. Így megvalósíthatók például 16 kbit/s, 64 kbit/s, 128 kbit/s, 1536 kbit/s vagy 2048 kbit/s sebességű csatornák. A sávszélesség az SMT-n keresztül kerül igénylésre. Az izoszinkron forgalomban lehetséges WBC-k maximális száma az inicializálás időpontjában kerül rögzítésre.

Ezzel az FDDI II nagyon jól alkalmazható folytonos médiumok adatainak átvitelére. Az FDDI II kereskedelmi fontossága, ellentétben az eredeti FDDI-vel még vitatott. Ennek egyik oka a két rendszer inkompatibilitása lehet: egy FDDI II lokális hálózathoz a meglévő FDDI rendszereket nem lehet közvetlenül hozzákapcsolni, ezeket valami mással kell helyettesíteni.

9.6 DQDB

A token továbbadásának mechanizmusa 100 Mbit/s adatsebesség felett és a hálózat 100 km-nél nagyobb térbeli kiterjedése mellett már nem nagyon hatékony [RuBa90]. Ezen a háttéren keletkezett egy további hálózat, melyet először ***Queued Packet Synchronous Exchange***-nak ***(QPSX)*** neveztek [NBHu88]. Ez a későbbi Queued Packet Synchronous Exchange nevű cég, a Nyugat-Ausztráliai Egyetem és a Telecom Australia kooperációjából származik. Az egy cégnév és egy szabvány közötti névadási konfliktusok miatt ezt a hálózatot azóta ***Distributed Queue Dual Bus (DQDB)*** néven nevezik.

Az IEEE 802.6 MAN-szabvány egy 2 × 150 Mbit/s adatátviteli sebességű buszt ír le, amely különböző kábelfajtákkal alkalmazható. Ahogy a 9–10 ábra a 232. oldalon mutatja, a DQDB két egymással ellenkező irányú buszrendszeren alapszik. Szemben az FDDI-vel mind a két busz információt hordoz és ezzel nem kizárólag a hibatolerálásra szolgál. Az adatok a buszon – mint ahogy az FDDI II-nél is – 125 µs hosszúságú keretekben *(Frames)* állnak rendelkezésre. Minden keret ismét maga is több fix hosszúságú időszeletet *(Slots)* tartalmaz. Az időszelet szállítja az információt két csomópont között. Az adatfolyam a két fejvégen kezdődik és végződik, a keretek itt generálódnak, majd oszlanak fel később újra.

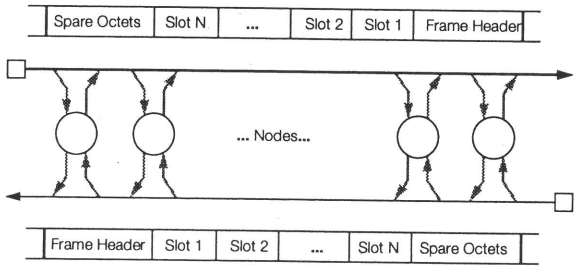
Izokron adatátvitel

A folyamatos adatok szempontjából fontos izoszinkron adatátvitel egy ***Pre Arbitrated Function*** keresztül történik. Ezért a fejállomásokon bizonyos időszeletek egy megfelelő módon megjelölésre kerülnek, a *Slot Type SLT* bejegyzés az 1 értéket kapja, ami által csak az előre lefoglalt forgalomnak állnak rendelkezésére. Az időszeletek minden 125 µs-ben rendelkezésre állnak, ami egy 8 kHz-es frekvenciának felel meg, amivel a DQDB izoszinkron üzemmódja éppen úgy kapcsolható a távolsági forgalom PCM hierarchiával, mint a FDDI II-vel. Azonkívül ezzel a 8 kHz-es raszterrel minden audio- és videoadat integrálható.

Aszinkron adatátvitel

Az aszinkron adatátvitel a ***Distributed Queueing Algorithmus*** szerint dolgozik. Az alapot az elosztott várakozási sorok képezik: Minden állomás folytonosan figyeli a hálózatot és adott esetben beilleszt egy adási igényt, amihez a *Requeset-mezőt* állítja be. Minden irányban minden állomáson egy számláló segítségével egy várakozási sor kerül implementálásra, amelyben prioritás is lehet. A következő 4 lépés példaként mutatja ezt a folyamatot:

1. Minden, *nem adni akaró állomás* (*Node* a 9–10 ábra szerint) az alsó buszon számolja a jobbról jövő kéréseket. Ezek a belső várakozási sorban (ill. számlálóban) kerülnek elhelyezésre. Amint a felső buszon a bal oldalon egy szabad időrés megjelenik, a várakozási sorban legrégebbi kérés kielégítésre kerül és egyúttal ez a bejegyzés kikerül a várakozási sorból. A várakozási sor így ezek után az összes többi jobbról még nyitva lévő adási igényt tartalmazza, amelyek a felső buszra átvihetők. Ugyanez az elv érvényes az alsó buszra a balra fekvő állomásokkal és egy második várakozási sorral.



9–10 ábra
Distributed Queue Dual Bus. Elvi topológia a szabvány szerinti megnevezésekkel

2. Egy *adni akaró állomás közli az adási kérését:* e célból az alsó buszon egy szabad kérőmezőre vár. Amint ez megérkezik, a *Request-mező* segítségével jelzi a kérését az alsó buszon és besorolja a saját adási igényét a felső busz várakozási sorába.

3. Az *adni akaró állomás vár,* hogy az adatait aszinkron üzemmódon el tudja küldeni. Azonkívül ez az állomás mindig maximum egy saját kérést helyezhet el a fent leírt elv szerint a várakozási sorában. A kéréseket folyamatosan, a szabad időszeleteknek megfelelően kell a sorból kivennie, a leírás első pontjának megfelelően, azonkívül a további az alsó buszról bejövő kéréseket is felveszi a sorba.

4. Az *állomás küldi a saját adatait:* A saját adási kérés áll a legfelső helyen a várakozási sorban. A felső busz első szabad időrésénél kerülnek ezek az adatok átvitelre és a várakozási sorból eltávolításra. A szabad időrés mint foglalt *(Busy)* kerül megjelölésre. Az ezentúl beérkező adási igények besorolásra és mint azt fent leírtuk, feldolgozásra kerülnek.

Ennél az eljárásnál fellép az elérhetőség problémája. Minden állomásnak tudnia kell, hogy más állomásokat mely buszon keresztül érhet el, azonkívül az állomás a fejállomáshoz viszonyított helyzete lényegesen kihat a **méltányosság**ra. Azok az

állomások, amelyek a fejvég közelében találhatók, inkább tudják adataikat ennek a fejnek az irányába küldeni, mint a többi állomás. A leterheltséggel kapcsolatban fontos, hogy az adatok mindig a fejvégig továbbításra kerülnek, akkor is, ha ez a tulajdonképpeni adatátvitelhez nem lenne szükséges.

A DQDB szinkron üzemmódja kiemelkedően alkalmas folytonos médiumok adatainak átvitelére. Mindenesetre a legtöbb megvalósítás még itt sem támogatja ezt az üzemmódot. Több állomásnak a hálózati adatokhoz való egyidejű hozzáférése nagyobb kiterjedésű hálózatoknál is lehetővé teszi a hatékony, nagy áteresztést.

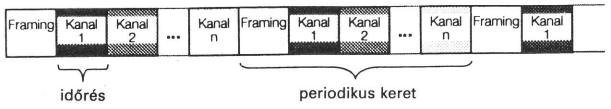
Az összes eddig tárgyalt hálózat a biztonsági réteg, az IEEE 802.2 ***Logical Link Control*** szabvány szerint a legfelsőbb szinten nyújtja szolgáltatásait. Ez a réteg nem mutathat egyetlen multimédiás adatok átvitelére szolgáló sajátságot sem, csak a megfelelő MAC szolgáltatásokhoz való hatékony hozzáférést kell a szükséges paraméterekkel biztosítania.

9.7 B-ISDN

A ***Broadband Integrated Services Digital Network****nek* (B-ISDN), mint az ISDN-t követő fejlesztésnek egy felhasználói interfész és hozzá tartozó forgalmi hálózat definiálása a célja. Különböző beszélgetési, elosztó, lekérdező és üzenő szolgáltatások kerülnek támogatásra azzal a céllal, hogy nem kapcsolatorientált és kapcsolatorientált szolgáltatásokat bocsássanak különböző médiumok rendelkezésére. A B-ISDN a CCITT Study Group XVIII-ban jött létre. Ez kb. 1987-ig az STM *(Synchronous Transfer Mode)* átviteli üzemmódon alapult. 1988 óta viszont az ATM *(Asynchronous Transfer Mode)* az alapja a B-ISDN-nek [HaHu91a, Stal92a]. Itt egy kb. 140 Mbit/s sávszélesség került rögzítésre.

Synchronous Transfer Mode (STM)

Az STM kapcsolatorientáltan, fixen hozzárendelt sávszélességgel dolgozik az időmultiplex eljárás szerint. Itt már a legalsó szinten támogatható egy meghatározott sávszélességű garantált átvitel, mint az ISDN-nél. A végpont-végpont késleltetés csekély.



9–11 ábra
Időszeletek hozzárendelése az STM-ben (időmultiplex eljárás)

Az időszeletet itt *Slotnak* nevezik. Ezek az időszeletek egy kapcsolat tartamára foglaltak. Ezek egy periodikusan ismétlődő struktúrán, egy kereten belül fekszenek. A *kapcsolat időszelethez* való hozzárendelése annak a kereten belüli helyzetéből adódik (l. a 9–11 ábrát a 233. oldalon). Minden időszeletnek rögzített időtartama van.

Az STM jól illik a **PCM átviteli hierarchiá**kba és egyúttal a keskenysávú ISDN-be is. A következő csatornák kerültek a megadott adatátviteli sebességekkel rögzítésre:

9–3 táblázat
STM csatornák a megfelelő adatsebességekkel

Csatorna	Sávszélesség
B	64 kbit/s
H_0	385 kbit/s
H_1	1920 kbit/s (Európa)
H_2	32 768 kbit/s (Európa)
H_4	132 032–138 240 kbit/s

Ez mindenesetre a rögzített adatátviteli sebességek és a sávszélességnek a kapcsolatokhoz való rögzített hozzárendelése miatt nem egy flexibilis struktúra. Ezért azután egy videoátvitel esetén például az 5.6 fejezetben a 102. oldalon leírt tömörítési eljárást kellene alkalmazni. Sok multimédia rendszer azonban más kódolási eljárásokat alkalmaz, mint pl. a JPEG vagy MPEG. Mint megoldás bevezethető **sok időszelet,** például kb. 2000 8 bites időszelet 150 Mbit/s esetén. A megnövekedett adminisztráció sok kommunikációs lehetőségnél hátrányos következményekkel jár. Mint kompromisszum, egy konténer-megoldás is megvitatásra került, amelyik korlátozott számú felosztást enged meg. Például a $H_4 + 4 \times H_1 + n \times B + D$ **konténerek**kel a H_1 és a H_4 csatornában lehetne, bár ezáltal a *merev* STM mechanizmus megmaradna és a keletkező részterhelések miatt a megmaradó sávszélesség sem kerül összefüggő kihasználásra.

Ha ezek az adatok fix adatsebességgel állnak rendelkezésre, akkor az STM nagyon jól alkalmas folytonos médiumok adatainak kommunikációjára.

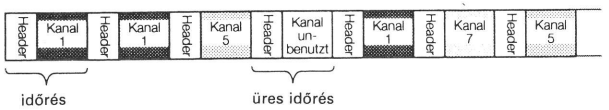
Asynchronous Transfer Mode (ATM)

Az ATM egy csomagszerű közvetítés, amelynél minden csomag mint rekesz kerül megnevezésre. Elsődlegesen kapcsolatorientáltan dolgozik, nagy sávszélességgel és relatív kis késleltetéssel. Az az alapgondolata, hogy a hálózatból lehetőleg minden funkciót ki kell hozni, hogy egy egyszerű, hatékony és gyors hálózatot kapjunk. *(A [Choi89, GRVe90, ISCh92, John89, MiSp89, NaKa89, NOOh89, OhKi90, TKTS92, WoKo89, YSSa92] elmélyül néhány fontos kérdésben.)*

A sávszélesség fix hosszúságú **rekesz**ekre kerül felosztásra. Minden rekesz fejből és a tulajdonképpeni adatokból áll. A rekeszek szükség esetén kerülnek allokálásra és nem előre lefoglaltak, mint az STM-nél, így a *kapcsolatnak az időszelethez (vagy rekeszhez)* való rendelése nem a kereten belüli helyzetből adódik; minden rekeszfej tartalmazza egy virtuális kapcsolat nevét. Ez a 24 bites *Virtual Path Identifier* azonosítja a virtuális kapcsolatot. Egy rekesz fix hossza összesen 48 byte, ezelőtt ennek a T1S1 szerint 64 byte-nak és az ETSI szerint 32 byte-nak kellett volna lennie. Itt tisztán politikai okokból egyeztek meg 48 byte-ban.

Az útválasztás az ATM-nél a tulajdonképpeni adatátvitel előtt történik. Minden rekesz, amelyik egy kapcsolathoz tartozik, ezen az előzőleg rögzített úton kerül átvitelre, így az adatok átvitele mindig a helyes sorrendben történik. Ezzel az ATM-nél nem garantált az átviteli csatorna 100%-os leterhelése, ám emiatt csak jelentéktelen hibák keletkezhetnek. Azonkívül az ATM réteg nem ismer adatfolyam-irányítást, ami folytonos adatok átviteléhez a sebességirányítással ellentétben nem is szükséges. Egy ATM-hálózatban minden hálózati kapunál az adatok 48 byte nagyságú rekeszekbe kerülnek becsomagolásra. Egy telefonminőségű, 64 kbit/s PCM kódolású átvitel esetén ez egy nem elhanyagolható késleltetést jelent.

$$\text{Késleltetés} = \frac{\text{48 byte}}{\text{8000 byte/s}} = \text{6 ms}$$



9–12 ábra
Időszeletek hozzárendelése az ATM-ben

ATM Adaption Layer (AAL)

Az *ATM Adaption Layer (AAL),* az ATM bázisán, szolgáltatások több osztályát bocsátja a különböző médiumok rendelkezésére:

- Az *A osztályt* egy állandó végpont-végpont késleltetés jellemzi állandó bitsebességgel. Ez egy kapcsolatorientált szolgáltatás, amely például videoadatok rögzített adatsebességű átvitelénél szükséges.
- A *B osztály* ugyanúgy, mint az A osztály egy állandó végpont-végpont késleltetéssel rendelkezik, ám ennek változtatható a bitsebessége. Ennek a kapcsolatorientált szolgáltatás felhasználásának egy példája egy változtatható adatsebességű videoátvitel.

- A *C osztály* változtatható bitsebesség mellett nem garantálja az adatfolyam állandó végpont-végpont késleltetését. Ez is kapcsolatorientáltan dolgozik.
- A *D osztály* változtatható bitsebesség mellett nem garantál állandó végpont-végpont késleltetést egy nem kapcsolatorientált szolgáltatásban.

A [CoVi89, Mas$91, MSTa89, ONTK90, WFMS89] az ATM-et különböző adatfolyamok kontextusában szemléli.

B-ISDN – egy összehasonlító összefoglalás

Az STM a multimédiális adatátvitelhez a rögzített sebességű adatfolyamok hatékony realizálásának előnyeit kínálja. Ez a hatékonyság mind a sávszélesség kihasználására, mind a végponttól végpontig mért késleltetésre vonatkozik. Az ATM hatékonyabb a változó sebességű adatfolyamok átvitelénél. Itt jobban ki lehet használni a hálózat teljes sávszélességét. Az ATM *flexibilisnek* számít. A BERKOM projekt keretében az első fázisban különböző gyártók STM és ATM hálózati csatlakozásokat mint prototípusokat valósítottak meg. Ennél minden gyártó a saját végrendszerét alkotta meg. A BERKOM most induló második fázisában a gyártók egymással közvetlenül a B-ISDN-en keresztül kommunikálnak és ugyanazokat a protokollokat magasabb szinteken is alkalmazzák.

9.8 A hálózati réteg és a transzportréteg

Nem minden, az összes folytonos médiumok átviteléhez szükséges funkció rendelhető egyértelműen a kommunikációsrendszer egyetlen rétegéhez. Ezért ezeket konzisztencia-okokból a 241. oldalon a 9.9 fejezetben mutatjuk be. Ebben a szakaszban mindenekelőtt néhány funkciót magyarázunk meg, amelyek elsődlegesen a *hálózati rétegben,* a folytonos médiumokkal összefüggésben fordulnak elő.

9.8.1 A hálózati réteg

Egy **csomagszerű,** összeköttetésmentes **átvitel** az erőforrások nagyfokú kihasználását hozza ingadozó sávszélesség-igény esetén. Azonkívül kiesik a kapcsolat felépítésének késleltetése. Nem tartalmazza viszont a folyamatos adatokhoz szükséges megfelelő időben történő átvitelt, amelynél a vevőhöz minden csomag egy meghatározott sorrendben, egy előre definiált megbízhatósággal érkezik meg. Az izoszinkronitást prioritásokkal kell kikényszeríteni, és a sávszélesség csak feltételesen garantálható.

Egy **kábeles átvitel** egy izoszinkron csatorna meghatározott sávszélességének minden kapcsolatra való lefoglalásával garantálja a megfelelő időben történő átvitelt. A csomagokat a helyes sorrendben és csomagveszteség nélkül (azaz nagyon nagy valószínűséggel) viszi át. Ehhez jön viszont még adott esetben alkalmazott változó adatsebességű tömörítő eljárás adatátviteli csatorna-pazarlása, ami többek között a hálózatok sávszélességét is érinti. Azonkívül a kapcsolat fel- és leépítése viszonylag sokáig tart. Ezért egy ilyen koncepció példának okáért rövid, tranzakciószerű adatátvitelre nem alkalmas. Egy ilyen, kapcsolatorientált szolgáltatás ennélfogva jobban alkalmas audio- és videoadatok átvitelére.

Egy, az adatkommunikációra koncepcionált közvetítőréteg nagyon hatékony módon teszi lehetővé heterogén rendszerek összekapcsolását. Példák az OSI, illetve az Internet architektúrából származó X.25 és IP.

Ma a folytonos adatok kommunikációjára szolgáló közvetítőréteg különböző koncepciói és megvalósításai ismertek. *(Emellett az OSI-modell bővítései is keletkeztek, l. [Cra$92].)* Itt mint legérdekesebb rendszereket, szemléljük meg közelebbről az *α-Channelt* és az *Internet Stream Protocol Version 2-t* (ST-II). A közvetítőréteg folytonos adatok szempontjából fontos aspektusai a minden adat ugyanazon a módon való átvitele és az erőforráshoz való hozzáférés elérése és biztosítása elosztott környezetben, illetve adott esetben több hálózaton keresztül.

Az α-Channel

Az α-Channel egy **egyirányú kapcsolat**ot (szimplex) fejez ki **egy forrás és egy nyelő között** a közvetítőréteg szintjén [FiZn91a], ez idáig még nem tartalmaz Multicast-lehetőséget, ami különösen a multimédia-konferenciák és elosztószolgálatok területén nagyon fontos, ha nem éppen előfeltételnek tekinthető. Amennyire a szerző előtt ismert, ezen a bővítésen Pittsburghben dolgoznak. Az adó transzport rétege egy, az α-Channel által megvalósított szolgáltatást (végponttól végpontig kapcsolat meghatározott tulajdonságokkal) használ egy kapcsolat felépítésére. Ezt a kérést a hálózatok leterheltségének és a megadott szolgáltatás-jósági paramétereknek megfelelően fogadják el és ezzel garantálják, vagy a hiányzó átviteli kapacitás miatt elutasítják.

ST-II

Az ST-II az alkalmazás – a transzportréteg – rendelkezésére bocsátja a garantált adatátvitelt, mint egyirányú kapcsolatot egy adó és több vevő között [HeDe92a, Topo90, PaPi91]. Két résztvevő közötti kommunikációra Duplex kapcsolatok is felépíthetők. Az új protokollarchitektúrákba való átmenet megkönnyítésére ez az Internet környezet első implementációiban történhet. Azonban ebben az átmeneti fázisban csak egy feltételesen garantált átvitel lesz lehetséges. Az ST-II adatfolyamok egy viszonylag ráfordításigényes kapcsolatfelépítési protokollal épülnek fel, amely nagyon jól összefogja a különböző erőforrások foglalását. Az ST-II-vel külö-

nösen folytonos adatok vihetők át előnyösen. Abból indultak ki, hogy egyidejűleg diszkrét adatok átvitelére szolgáló protokollok, pl. az IP is rendelkezésre állnak. Az ST-II illetve az SCMP maga is használja ezeket a kapcsolat felépítésének fázisában. Az ST-II egy ***Flow-Specification*** formájában kapja meg a kapcsolattal szemben támasztott követelményeket, a csatornaparaméterekkel a végpont-végpont késleltetés, a PDU sebesség, a PDU méret és az adott kapcsolat fontossága kerül megadásra. Ebből minden számítógép-csomópontra kiszámításra kerülnek az erőforrás-foglalási paraméterek. Egy ST-II-s közvetítő rétegnek minden számítógép-csomópontra van egy ***ST-II Agentje*** (ügynöke), amelyik az adott lokális erőforrások lefoglalásával törődik. Az ST-II definiálja az interfészt ezekhez a komponensekhez. Magát az *Agentek* implementációját nem rögzítették és így az a mindenkori környezethez optimálisan hozzáilleszthető. Minden implementáció szabadon választhat, hogy hogyan integrálja például a számítógépekben a valósidejű feldolgozáshoz szükséges memória és CPU-teljesítmény követelményeket. Az ezen erőforrás-foglalás és a mindenkori operációs rendszer közötti szoros kapcsolat miatt a lényeges lépéseket a 8.2 fejezetben, a 172. oldalon részletesen leírjuk.

9.8.2 A transzportréteg

A közvetítőrétegre támaszkodva a transzportréteg a mindenkori médiumokhoz hozzáillesztett szolgáltatásokat bocsát az alkalmazás rendelkezésére *(L. pl. [HeSt91])*. Ez a transzportréteg LDU-k formájában kapja meg a folytonos adatokat az alkalmazástól. Ebben egy LDU egyenként digitálisan kódolt 8 bit hosszú hangmintákat vagy egy teljes intraframe kódolását 10 kbyte-nál hosszabb egyedi képet tartalmazhat. A transzportréteg ezekből alkalmas csomagokat képez, amelyeket a közvetítőrétegnek továbbít.

Csomagolási késleltetés

Folytonos médiumoknál csomagolási késleltetést figyelhetünk meg: ha például 8kHz-cel mintavételezett 8 bites értékekből álló 2 kbyte méretű csomagokat állítunk össze egy lokális hálózaton, akkor csak egyedül ez a folyamat kb. 250 ms-ig tart, ami a legtöbb alkalmazás számára elviselhetetlen. A transzportrétegnek itt adott esetben kisebb csomagokat kell képeznie, vagy a csomagokat üres adatokkal feltölteni. A vevő ennek megfelelően bontja a csomagokat, és küldi tovább az adatokat. A hasonló csomagolási késleltetéseket a hálózati átmeneteknél is el kell kerülni.

A nagyon nagy LDU-kat a transzportrétegnek az alsóbb rétegek több csomagjaira kell felosztania *(szegmentálnia)*. Célszerű az alsóbb szint egy csomagjában mindig egy LDU adatait továbbadni. Ennek megvannak az előnyei a hibakezelésnél, mert adott esetben mindig csak egy LDU adatai hibásodnak meg.

Hibakezelés

Egy további lényeges szempont a hibakezelés [FeBi90]. Ennél egy üzenetblokk megváltozhat, elveszhet, megduplázódhat, elavulhat vagy kézbesíthetetlen lehet. Az adatkommunikációnál fellépő hibákat a legtöbb multimédia alkalmazásban nem lehet az adott csomagok újbóli átvitelével elhárítani. Ennek a szűk végpont-végpont késleltetési határ az oka. Hibák fellépése esetén a következő intézkedések tehetők, amelyeket a mindenkori tömörítő eljárással szoros összefüggésben kell szemlélni:

- Az első variációban bár egy hiba zavaró, de nem teszünk további intézkedéseket. Ennek különösen egy nem tömörített videoátvitel esetén van értelme. A csomagban fellépő hiba csak **egy hibás képkocka** formájában nyilvánul meg. Ez 25 Hz-es vagy 30 Hz-es képváltásnál nagyobb frekvencia esetén minden további nélkül tolerálható. Csak az a fontos, hogy bár a videkomponensek ezt a hibát jelzik, de ez nem zavarja funkciójukat.
- Ennek a megoldásnak **egy javítási lehetősége**, a mindenkori alkalmazás hiányzó vagy meghibásodott adata helyett az előzőleg kapott adatok megismétlése. A konkrét működési mód erősen függ az alkalmazott tömörítési eljárástól, ezért célszerű az alkalmazás ezen helyére – és ezzel a kódolókomponensek számára is – a hibás adatblokkra való utalás továbbítása. Egy hatékony megvalósítás feltételezi, hogy a kódoló komponensek az aktuális adatblokkot mint az adatok egy bizonyos részét egyértelműen azonosítani tudják. A hibás adategységnek a kódolókomponensekhez mérten egy logikusan strukturált tartománynak kell lennie. Ez például mindig egy teljes képkockának az előzővel való helyettesítését jelentheti. Audiojeleknél itt zajt vagy az előző hangszeletet lehetne bejátszani.
- Időbeli különbségi kódolás esetén az eddig leírt eljárások csak feltételesen alkalmazhatók, mert itt egy bizonyos hosszúságú sorozathiba keletkezik. Itt vethető be a ***Forward Error Correction*** (FEC). Így viszont egy ellentmondás keletkezik a tömörítési technika és az utólag bevezetett redundancia között. Ha mind a két szempontot egymástól függetlenül tekintjük, akkor
  – a hatékony tömörítés az adott médiumra van meghatározva *(l. 5.3 fejezet, 77. oldal)*,
  – a FEC az adott átviteli csatornához és a hibakarakterisztikához van illesztve.
  Egy kombinált tömörítés és FEC éppenséggel előnyös lehet. A legegyszerűbb hibafelismerési eljárás a ***Cyclic Redundant Check*** (CRC). A CRC-t mind hardverben, mind szoftverrel megvalósíthatjuk. Hibakorrekcióra használhatjuk például a CD-technológiához a hálózatok problematikájában kifejlesztett eljárást.

Az adatátviteli sebesség skálázása

Az adatfolyamban fellépő hibák egy nem 100%-osan garantált átvitel esetén az adott hálózat túlterheléséből származhatnak. A folytonos adatok továbbra is ugyanazzal az adatátviteli sebességgel a hálózaton való küldése, és ezzel további hibák provokálása helyett alkalmas rendszerek ebben az esetben az adatátviteli sebességet csökkentik az adóoldalon. Ezt nevezzük az adatátviteli sebesség skálázásának. Ennek érdekében magának a tömörítőalgoritmusnak kell skálázhatónak lennie. A transzportréteg az adónál ütközik ebbe a folyamatba, amely elsődlegesen a videoadatokra van kihatással. **Egy skálázás különböző paraméterek**re vonatkozhat, és történhet

- időbeli skálázással, ami a másodpercenkénti képszámot változtatja meg,
- térbeli skálázással, ami a bitmélység vagy a függőleges és a vízszintes felbontás csökkentésével történik,
- további paraméterek skálázásával, aminek minőségi kihatásai vannak. Korlátozott számú DCT-koefficiensek figyelembevétele vagy különféle kvantálások szolgálhatnak példaként.

A skálázás maga ekkor a kódoló komponensek feladata. Egy érdekes kérdésfelvetés ebben a kontextusban annak a kritériuma, hogy mikor vezessük be a skálázást. Erre a vevőhöz megérkező csomagok feladási idejét használhatjuk. Ha egy csomag *túl későn* érkezik meg, akkor csökkenteni kell az adási sebességet. Más kiindulások az adónál elküldött csomagok várakozási sorának feltöltöttségi szintjére hivatkoznak. Torlódás esetén csökkenteni kell az adási sebességet.

Szinkronizációs kapcsolatok

A transzportréteg eddig az egyes kapcsolatokat egymástól független adatoknak tekintette. Szinkronizációs kapcsolatokon keresztül ezek az adatok többek között egy időbeli relációban állnak egymással (*l. 13. fejezet, 325. oldal*). Itt szolgáltató elemek (SDU-k) és protokolladategységek (PDU-k) formájában egy üzenetblokkok közötti kapcsolat látható. Ezt rávetíthetjük a transzportrétegre, hogy lehetővé tegyük a 13.4 fejezetben a 335. oldalon leírt élő-szinkronizálást a különböző kapcsolatok között. Azonban ennek a szinkronizációnak a kapcsolatok közötti implementációja elvben nem különbözik a többi élő-szinkronizációtól. Ezért a multimédia rendszerek ezen fontos aspektusának nem feltétlenül kell a transzportrendszer részének lennie.

9.9 Rétegeken túlmutató szempontok

Egy multimédia kommunikáció sok fontos funkcióját nem lehet egyértelműen egy kommunikációs réteghez hozzárendelni, ezért ezeket ebben a szakaszban átfogóan megmagyarázzuk.

Szolgálat-jósági paraméterek

Egy alkalmazás követelményei kommunikációs rendszerekben a legtöbbször a szolgálat-jósági paraméterek formájában konkretizálódnak. Ezek a paraméterek a különböző szintek különféle kifejezési formáiban és a rendszerkomponensekben is érvényesek *(l. 8.2.2 fejezet, 175. oldal)*.

9–4 táblázat
Megbízhatósági osztályok, egy példa

Osztály	Bithiba	Csomaghiba
0	nem érzékeli	nem érzékeli
1	nem érzékeli	felismeri és jelzi
2	felismeri és jelzi	felismeri és jelzi
3	nem érzékeli	javítja
4	javítja	javítja

Az **ST-II**-nek szüksége van a késleltetés, a PDU-sebesség, a PDU méret és a kapcsolat fontosságának értékére. A célul kitűzött PDU-sebességet, például 30 csomag per másodpercet egy alsó korláttal együtt adják meg. A PDU méretre is két értéket definiálnak, a kívánt méretet és egy kisebb, még elfogadható méretet. A kapcsolat fontossága határozza meg, hogy melyik kapcsolat tud túlterhelés esetén a leginkább hibát tolerálni. A késleltetés a maximális érték ST-II formájában kerül megadásra. Ezen a helyen az ST-II specifikációt a jósági paraméterek vonatkozásában ki kellene bővíteni: az abszolút határérték mellett még egy elérendő értéket is meg kellene adni. A garanciák jobb betartása érdekében az egyes csomagok előre feldolgozása *(Workahead-módban)* nagyon sokat segíthet *(l. 8.2.5 fejezet, 178. oldal, és 8.3.9 fejezet, 201. oldal)*. E célból az ST-II-t ki lehetne egészíteni egy *Workahead*-paraméterrel, ami a PDU-k számát vagy egy bizonyos adatmennyiséget adna meg.

Az ST-II a kapcsolat felépítése alatt elvégzi ezeknek az egymástól függő értékeknek az optimalizálását. A kapcsolat felépítési-kívánság eredményeként rendelkezésre állnak az erre a kapcsolatra érvényes értékek. Itt azután többek között még

a „Jitter", az elérhető megbízhatóság, és egy bithiba-gyakoriság kerül megadásra. A megbízhatóság annak a valószínűségére vonatkozik, hogy a csomagok a vevőt nem bithiba nélkül vagy nem a megadott időbeli korláton belül érik el. Egy transzportprotokoll, mint például a HeiTP [DHHS92a] definiálja az átviendő adatok megbízhatósági osztályait. Ezeket az alkalmazás adhatja meg. A transzportréteg összehasonlítja az ST-II által elért megbízhatóságot az alkalmazás követelményeivel, és megteszi a megfelelő lépéseket. A 241. oldal 9–4 táblázatában a [HSSt89b, HSSt90] szerinti megbízhatósági kategóriák lettek megadva.

Az **α-Channel** esetén az α értéke fejezi ki az átviendő adatok megbízhatóságát. Ezt az értéket mind a *hálózati sávszélesség* erőforrásnál, mind a csomagok számítógép-csomópontokon való továbbításakor figyelembe veszik.

$\alpha = 1$	Egy kapcsolat minden csomagja időben a vevőhöz érkezik. Az α-Channelt ekkor *determinisztikusnak* nevezzük.
$0 < \alpha < 1$	Az α-val megadott érték meghatározza az egy kapcsolat adatcsomagjainak százalékában azoknak az adatcsomagoknak az arányát, amelyeknek időben el kell jutniok a vevőhöz. A maradék 1-α adatnak kellene – de nem muszáj – ennek a feltételnek megfelelnie. Az α-Channelt ekkor *statisztikusnak* nevezzük.
$\alpha = 0$	Egy ilyen kapcsolat csomagjai a determinisztikus vagy a garantáltan statisztikus adatokkal szemben háttérbe szorulnak, ám ezeket az adatokat a diszkrét adatokkal szemben mégis előnyben kell részesíteni. Az α-Channelt ekkor *Best-Effortnak* nevezzük.

Ez a gondolat nagyon jó a közvetítő csomópontok programozásánál és jól kihasználható a részt vevő hálózatok adatátviteli-csatorna foglalásánál. Vele jól meghatározható és garantálható az adatátviteli csatorna közepes leterheltsége. Tömörítetlen adatfolyamokra szintén jól használható. Tömörített, folytonos médiumok gyakran különböző fontosságú LDU-kból állnak: egy teljes kép átvitele lényegesen fontosabb, mint a rá következő különbségképeké, mert az összes rá következő adat a teljes képre hivatkozik. Emellett a teljes képet közvetlenül követő információ fontosabb, mint a később átvitelre kerülő adatok, mert a különbségek mindig az előző képre vonatkoznak. Az α-Channel α értéke mindig egy kapcsolat összes csomagjára vonatkozik, miáltal az összes különbségi adat ugyanolyan fontosságot kap, mint a teljes kép. Ezért használja ki a statisztikus vagy a Best-Effort α-Channel koncepciója a médiumtulajdonságokat csak feltételesen. Az is meggondolandó, hogy folytonos adatok kommunikációjakor egyáltalán meg kell-e engedni egy bizonyos veszteség lehetőségét. A gyakorlatban a felhasználói oldalon felléphet egy olyan hatás, hogy mindenki egy determinisztikus csatornát szeretne kapni.

Az α-Channelnél egy csomag betartandó végpont-végpont késleltetése és az ebben az időszeleten belül elküldésre kerülő csomagok maximális száma is megadásra kerül. Ez a mennyiség a szükséges memória meghatározására és egy számítógép-csomópont maximális leterhelésének kiszámítására szolgál a hálózatban. Ebben a modellben nem feltételeztük a megérkező csomagok elosztását.

Fix útválasztás

Az egy kapcsolat csomagjai számára rögzített végpont-végpont késleltetést, a sávszélességet és ezzel az időbeni és korrekt sorrendű átvitelt is garantálni tudjuk. Minden csomag számára egy és ugyanazon útvonal kerül rögzítésre. Ezt *fix* útválasztásnak is nevezzük. Néhány esetben a hálózat már átveszi ezt a feladatot. Az ISDN vagy B-ISDN ATM-mel és STM-mel példák erre *(l. a 9.7 fejezet, 233. oldal)*. A legtöbbször azonban ez a közvetítőrétegben megvalósítható. Ekkor a közvetítőrétegből egy ilyen ATM kapcsolatot mint egy, a 2. szinten lévő kapcsolatot láthatunk. A mindenkori útválasztás a megkívánt átviteli erőforrás-kapacitás kérdésében erősen alkalmazkodik a követelményekhez és az erőforrás-foglalással összefüggésben történik.

Az **α-Channel** ilyen célú algoritmusa 2 fázisban dolgozik:

1. Az első lépésben minden lehetséges útvonal meghatározásra kerül. A szerző véleménye szerint ez csak korlátozott számú lehetőséggel rendelkező rendszerekre praktikus.
2. A következő lépésben az összes, az egy kiválasztott út (az út kiválasztásának kritériumai az α-Channelnél még fejlesztés alatt vannak) által jelentett kapcsolat lefoglalásához szükséges erőforrás rendelkezésre állása lekérdezésre majd adott esetben lefoglalásra kerül.

Az α-Channelnél a lefoglalás a közbenső csomópontokon fellépő maximális terhelés bázisán történik. Ez az eset akkor fordul elő, ha minden α-Channel a saját maximális definiált adatsebességével adatokat küld, és az összes α-Channel csomagjai egyidejűleg érkeznek a közbenső csomópontra. Ezeknek az értékeknek az alapján meghatározható az egy számítógépben keletkező csomagkésleltetés és a csomaghoz szükséges maximális feldolgozási idő. Ezek az értékek összevetésre kerülnek a rendelkezésre álló kapacitással és megállapításra kerül, hogy a szükséges garanciák betarthatók-e. Ehhez minden kapcsolat α értékét minden köztes csomópont megkapja. A teljes maximális végpont-végpont késleltetés sem lehet nagyobb, mint az egyenként megállapított értékek összege.

Az **ST-II** alkalmazni tudja az IP-Routing funkciókat. Ennél az útvonal mentén az adótól a vevőig az erőforrással szemben támasztott követelmények rendelkezésre állása rögtön ellenőrzésre kerül. A kapcsolat teljes kiépítéséért és a jósági paraméterek eléréséért tulajdonképpen nem az ST-II hanem az *ST Control Message Protocol* (SCMP) a felelős. Az SCMP hasonló viszonyban van az ST-II-vel mint az ICMP az IP-vel (többek között a vezérlési adatok elküldésében). Az SCMP-n keresztül kerülnek a kapcsolatok (amiket itt *Streameknek* is neveznek) fel- és leépítésre, azonkívül a szomszédos ST-II állomások között státusinformációk kicserélésre. Ha minden követelmény kielégíthető, akkor a követelmény *visszaútján* (a vevőtől az adóig) egy adott esetben a követelmények még lehetséges meghatározása történik.

Példaként tekintsük a végpont-végpont késleltetést. Ez nem haladhatja meg összesen a 80 ms-ot. A foglalás odaútján az egyes komponensek legrövidebb garantálható időit tételezzük fel. A vevőnél például 35 ms-ot kapunk a késleltetésre. Ezzel a visszaúton a követelmények fokozatosan enyhíthetők, úgy, hogy az előírt 80 ms-ot ne lépjük túl.

Adatátvitel

Az adatátvitel alatt különböző kapcsolatok csomagjai érkeznek meg, és ezeket a garantált teljesítménynek megfelelően kell feldolgozni. Folytonos médiumok adatai feldolgozásának ezt a feladatát a 8. fejezetben a 167. oldalon leírt módszerekkel az operációs rendszer veszi át.

Az **α-Channel**nél a követelmények (a beérkező csomagok) egy, a FIFO-elv alapján létrehozott várakozási sorba kerülnek besorolásra. Minden egyes kapcsolat fontosságát az α határozza meg. Ha a várakozási sorban egy *determinisztikus* attribútumú csomag található más *Best-Effort* csomagok mögött, akkor az összes előző *Best-Effort* csomag eldobásra kerül és a feldolgozás a *determinisztikus* csomaggal folytatódik. A szerzőnek ez nem tűnik feltétlenül szükségesnek: előnyben kell részesíteni a determinisztikus csomagokat, az előttük álló *Best-Effort* csomagokat viszont nem kell eldobni. A determinisztikus csomagok feldolgozása után még elegendő szabad processzorteljesítmény állhat rendelkezésre a Best-Effort csomagok feldolgozására. Viszont nem szabad az egy kapcsolathoz tartozó csomagok sorrendjét megváltoztatni. Egyébként a csomagok a várakozási sorban elfoglalt sorrendjük szerint kerülnek feldolgozásra. Egy csomag minden feldolgozási folyamatához hozzárendelhető a fontosságának megfelelő prioritás.

Elosztott multimédia rendszerekben az adatátviteli sebességet legtöbbször közvetlenül a forrásnál szabályozzák azáltal, hogy az adatokat egy fix, közepes sebességgel küldik el a hálózatokon. Folyamvezérlést a szükséges végpont-végpont késleltetés és az emiatt video esetén nagyobb hálózatoknál adott esetben szükséges magas átmenetitár-kapacitás miatt nem lehet értelmesen alkalmazni. Itt egy ***sebességszabályozás*** szükséges. A sebességszabályozás egy egyszerű variációja egy rendszerben úgy történhet, hogy minden, egy kapcsolat csomagjainak elosztásában

részt vevő számítógép-csomóponton az adatátviteli sebesség a csomópont bemenetén a kimenettel megegyező illetve a középértéknek megfelelő marad egy hosszabb perióduson keresztül.

Az **α-Channel** minden csomagot a közvetítő szolgáltatásnak való átadásakor az adónál egy időjelzéssel lát el. Két egymásrakövetkező időjelzés különbségét minden közbenső csomópont a csomagok azonos sebességgel való továbbítására használ fel. Emellett figyelembe kell venni az alsóbb szintek időbeli viselkedését is. Itt úgy késleltethetők a csomagok, hogy a közepes sebesség egy kapcsolat ideje alatt kezdetben csökkenjen és később ismét érje el a korrekt értéket. Ekkor viszont a végpont-végpont késleltetés is nő. Magában az ST-II-ben egyetlen explicit feltételezés sincs az egyes állomások adatfeldolgozására. Itt például az operációs rendszerek mechanizmusaira támaszkodhatunk, amelyek a 8. fejezetben a 167. oldalon kerültek leírásra. A gyors adatátvitel támogatására az ST-II-nek az adatcsomagban egy 8 byte hosszú *Headerje* van a prioritásmegadásra, a közelben fekvő állomásokkal összefüggésben a kapcsolat egyértelmű azonosítására és a hibafelismerésre szolgáló ellenőrző összegre vonatkozó információra. Kiegészítőleg egy, az átviteli csatorna kiosztását a 8.3 fejezetben a 215. oldalon leírt módon megkönnyítő időjelzést is kaphat egy csomag. Az ST-II kibővítéseként egy *Workahead* is bevezethető, ami a csomagok előre feldolgozását teszi lehetővé [CFKS92a, VHNa92a].

Az adatátviteli sebesség szabályozásánál a legtöbb esetben abból indulnak ki, hogy az adónak és a vevőnek azonos fizikai sebessége van, de ez a legtöbb esetben nem áll, és ennek megfelelően a kommunikációban hibákhoz vezet. Ennek a fejezetnek a befejező részében egy új eljárást mutatunk be, amely ezt a problémát még multicast-összeköttetések esetén is megoldja [Far$92a].

Minden LDU egy D időtartamra érvényes és ez alatt az idő alatt kerül a vevőnél kiadásra. Ezzel az adatfolyam sebessége

$$R = \frac{1}{D}$$

A létrehozó adatsebessége – $SEND_s$ és az adatok felhasználójának adatsebessége – $EMPF_s$ különböző lehet. Multicast esetben minden egyes vevőnek különböző sebessége lehet. Az adatátviteli sebesség szabályozásánál abból indulunk ki, hogy az adatsebességek elvileg azonosak:

$$SEND_s = EMPF_s$$

Elosztott multimédia rendszerekben az adó és a vevő a legtöbbször különböző számítógépeken helyezkedik el. Ezek a számítógépek gyakran egymástól távol lévő helyekre szétszóródtak és különböző hálózatokkal kapcsolódnak egymáshoz. Ez a tény megnehezíti az egyes adatsebességek egyensúlyba hozását. A nagyon pontos

kvarcoszcillátoros órajeladók dacára a különböző rendszerek adatsebességei egy kicsit mindig eltérnek egymástól.

$$|\mathbf{SEND_s} - \mathbf{EMPF_s}| = \delta > \mathbf{0}$$

A következő eseteket különböztetjük meg:

- $EMPF_s < SEND_s$, a vevő sebessége kisebb, mint az adóé.
  Ezzel hosszú távon az adó, a saját sebességével átvitt csomagokkal a vevőnél *túlcsordulást* okoz. A vevőnek mind több és több adatot kell átmenetileg tárolnia, azonkívül folyamatosan nő a végpont-végpont késleltetés. Ez feltétlenül a puffer túlcsordulásához vezet, és ezután a legtöbbször újra kell inicializálni a prezentáció komponenseit.
- $EMPF_s > SEND_s$, a vevő sebessége nagyobb, mint az adóé.
  Itt az adó átvisz csomagokat, amelyek nem rögtön jelennek meg a vevőnél. Egy bizonyos időpontban a vevő megkezdi az adatok megjelenítését, de mivel a sebessége nagyobb, mint az adóé, ezért egy későbbi időpontban már nem lesznek adatok a kijelzéshez, és feltétlenül hiba következik be az adatok kiadásánál.

Első közelítésben a következő módon küzdhetünk meg ezekkel a problémákkal:

1. A vevő az adótól függetlenül dolgozik, a mindenkori adatsebességeket nem egyeztetik egymással. Audio- és videoadatok bemutatásánál így hibák keletkeznek a visszaadásnál, amelyeket a felhasználónak és a dekódoló komponenseknek tolerálnia kell(ene):
   A vevő bizonyos időközökben tudatosan átugorja a rendelkezésre álló adatokat ($EMPF_s < SEND_s$) vagy szünetek keletkeznek a további LDU-kra való várakozás alatt ($EMPF_s > SEND_s$).
2. Egy további megoldási lehetőségnél a vevőnek elegendő memóriája van legalább egy LDU tárolására. Mind az adó, mind a vevő a saját sebességével dolgozik.
   A vevő vagy bizonyos időközökben tudatosan átugorja a rendelkezésre álló adatokat ($EMPF_s < SEND_s$) vagy az új LDU-ra való várakozás alatt az előző információt használja fel még egyszer ($EMPF_s > SEND_s$).
3. A különböző kommunikációs példányok sebessége a hálózat által megadott sebességhez alkalmazkodik. A hálózatoknak itt ugyanazzal a sebességgel kell rendelkezniük, és a számítógépek sebességeinek támogatniuk kell az ehhez a jelhez való alkalmazkodást. Ez egy konkrét esetben például azt jelenti, hogy a hálózati kártya vezérli a tömörítőhardver órajelének frekvenciáját.

4. Továbbá lehetséges egy globális időbázis, mint a DCF 77 hosszúhullámú adó használata. Ekkor minden számítógépbe be kell építeni a megfelelő hardverkomponenseket.

A két első megoldás a felhasználótól elvárja a fellépő hibák tolerálását. A két utóbbi megoldás minden számítógépben plusz hardvert igényel, ami így általánosan nem tételezhető fel.

Egy jobb megoldás adaptálja a vevő sebességét az adó sebességére. A legtöbb alkalmazásban ezeket a folytonos adatokat a vételt követően rögtön továbbadják, és nem csak regisztrálják, mint egy videorekorderben. Ezért kell ennél az adatok azonnali visszaadásánál az órajelet, ami a visszaadás sebességét meghatározza, a vevőnél kívülről a szoftveren keresztül beállíttatni. Ezt sajnos nem tételezhetjük fel az összes rendszernél.

A vevő helyett az adó sebességét is befolyásolhatjuk, azaz a *vevő(k) vezérli(k) az adó sebességét.* Néhány alkalmazásnál az átvitel időpontjában az adatokat kamerával vagy mikrofonnal *játsszák be* a számítógépbe. Erre az *előadásra* a videotelefon, egy kamerás ellenőrző rendszer és az audiokonferencia szolgál. A 15.3 fejezet 396. oldala szerint ez a *nem perzisztens információelőállítás.* Más alkalmazásoknál az adatok már rendelkezésre állnak, és egy tároló médiumból kiolvasva kerülnek átvitelre. Video-adatbanki hozzáférések és filmek lejátszásai ilyen alkalmazások. Ezek a 15.3 fejezet 396. oldala szerint *perzisztens információelőállítások.* Az alkalmazások mind a két osztálya ugyanolyan gyakran fordul elő. Nem perzisztens információelőállítás esetén a kommunikációs rendszerben már rendelkezésre álló lokális átmeneti tárolót használhatjuk az adatsebesség kiegyenlítésére. Tárolómédiumból való átvitel esetén a sebességet a szoftverből határozhatjuk meg. Így ebben az esetben a vezérelt adó részesítendő előnyben, mert az adó sebessége mind a két esetben szoftver úton állítható be.

Az adó $SEND_s$ adatsebességét meghatározott a_{SEND} lépésekben megváltozhat. Amint a vevő megállapítja az adatátviteli sebesség eltérését, parancsot tud küldeni az adónak az adatátviteli sebesség növelésére vagy csökkentésére. Ez például

$$\mathbf{SEND_s} = \mathbf{SEND_s^{previous}} - \mathbf{a_{send}}$$

alakban jelentkezhet. Mind ez idáig csak ponttól pontig kapcsolatokra ismerünk ilyen mechanizmusokat. Ez az eljárás nem vihető át közvetlenül egy multicast környezetre, mert egy multicast összeköttetés esetében az adó általában a különböző vevőktől különböző parancsokat kapna.

Ezért a [Far$92a]-ban az eljárást az adónál egy járulékos ***arbiterrel*** bővítették ki. Ez a komponens veszi több vevő kívánalmait és ennek alapján dönt az adó sebességéről, amibe mint összehasonlító méret bevonható a vevő pufferjének mindenkori közepes feltöltöttsége. Ez az adatsebességek különbségeinek egy mértéke.

A memóriaméret mint mérték alkalmazása helyett szükség esetén más kritériumok is alkalmazhatók. Az arbitert tartalmazó eljárás továbbra is változatlan marad.

Ezek az értékek vagy periodikusan átvihetők a vevőtől az adóhoz, vagy minden vevő jelentkezik, mihelyt egy határérték túllépésre vagy átlépésre kerül. Audio, video és audio-video kommunikációra a 15% illetve a 85% adható meg kiinduló határértékként. Amint a multicast kapcsolat egy résztvevője határértéktúllépést jelez, az adó az ennek a kapcsolatnak az összes résztvevőhöz intézett globális kérdésére azok elküldik a vonatkozó közepes feltöltöttségi értékeket.

A kérdésre érkezett válaszokból egy csomó, a sebességek eltéréseire vonatkozó adat áll rendelkezésre. Az arbiter feladata, hogy az adott esetben az egymásnak ellentmondó információkat az adó sebességének egy értelmes – legjobb – illesztésévé foglalja össze. Ez a komponens különböző algoritmusok szerint működhet. Például az adó sebessége alkalmazkodhat a

- legmagasabb vevősebességhez,
- legalacsonyabb vevősebességhez,
- a közepes vevősebességhez,
- azokhoz a vevősebességekhez, amelyek puffereinek vevőoldali feltöltöttsége leginkább a megadott határértékeken belül marad.

Ennélfogva ez az eljárás a hiányzó globális órajel miatt egy hosszabb idő eltelte után néhány vevőnél a folytonos adatok visszaadásának hibájához vezet, ám a fellépő hibák számát ezzel az eljárással minimumra csökkenthetjük. Azonkívül ehhez a technikához nincs szükség járulékos hardverre és a ma létező rendszerekbe szoftverkomponensként integrálható. Ugyanez az eljárás illeszthető a hálózati kapuként dolgozó különböző számítógépekre is.

9.10 Zárómegjegyzések a multimédia kommunikációs rendszerekhez

A legtöbb ma termékként kapható kommunikációképes multimédia rendszert olyan számítógépekre koncepcionálták, amelyeken egy időben csak egy felhasználó dolgozik. Ugyanakkor a hálózatokat és protokollokat bárminemű adaptáció nélkül alkalmazzák. Alacsony hálózati leterheltség esetén – azaz amíg csak kevés ilyen alkalmazás fut egyidejűleg – a rendszerek korrektül működnek, ám ez a hálózatba kötött alkalmazások és a közösen használt munkaállomások növekvő számával megváltozik. A protokollokat alkalmas módon meg kell változtatni, amint azt ebben a fejezetben leírtuk.

Egy multimédia kommunikációs rendszer nem lealkudható feltételének a másik oldalon tévesen gyakran egy legalább 100 Mbit/s sebességű nagysebességű hálózatot neveznek. Mint azt ebben a fejezetben kifejtettük, multimédia adatokat, beleértve 25 vagy 30 képet másodpercenként, át lehet vinni meglévő lokális vagy Token Ring hálózatokon. Egy nagysebességű hálózat jobb minőséget és több független szimultán multimédia kapcsolatot tesz lehetővé. Azonkívül ezeket a hálózatokat már multimédia adatokra koncepcionálták amivel elkerülik a létező hálózatok összes hátrányát.

A multimédia rendszerekben a kommunikáció súlypontja az alkalmas hálózatok és transzportrendszerek rendelkezésre állása. Ennek alapján jönnek létre az OSI referenciamodell szerinti 5–7. réteg kommunikációs szolgáltatásai és protokolljai. Különösen figyelemre méltó a dokumentumokkal összefüggésben az elektronikus posta minden fajtája és a *CSCW* kooperatív munkája (l. többek között AEKP89, AGDa88, Egid88, EGRe91, EnLe88, Fisc90, Grei88, GrSa87, KrKi88, Lant88, Mas$92, Ohm$92, Wils88).

10. Adatbankrendszerek

Az adattárolás technikája ma már lehetővé teszi szöveg és egyéb diszkrét adatok mellett valósidejű audio- és videoinformációk beolvasását, tárolását és valósidejű kiadását is. Ez a régebben létező dedikált külső készülékek, valamint a rendszerintegrált másodlagos tárolók segítségével egyaránt lehetséges. A külső készülékek a stúdió- és szórakoztató elektronika területéről származnak, és nem diszkrét média tárolására fejlesztették őket. Erre példa a videorekorder, amely számítógép által vezérelhető digitális interfésszel rendelkezik. A rendszerbe integrált másodlagos audio- és videotárolók egyebek között optikai technológián alapulnak *(l. 6. fejezet, 125. old.)*; itt a kompaktlemez kiemelkedő szerepre tett szert. Az operációs rendszer az integrált tárolóeszközöket eszközmeghajtókon és file-rendszeren át köti össze az egész rendszerrel *(l. a 8.6. fejezet, 205. old.)*. A külső készülékek a beintegrált komponensekhez hasonló adatleírással érhetők el.

10.1 Multimédia adatbáziskezelő-rendszer

A multimédia alkalmazások rendszerint különböző szinten való absztrakciókkal címezik meg ezeket az adat és adatkezelő interfészeket.
ODA dokumentum architektúra az összeköttetést egy **MDBMS** *(Multimedia Database Management System)* multimédia adatbankkezelő-rendszerrel valósítja meg [She$89a]. Az ilyen absztrakciók ez idáig többnyire rendszerspecifikus formában léteztek, és konferenciarendszerek esetében nem voltak megfelelően kifejlettek.

Nézzük a következő három **példát:** a hipertext alkalmazásoknál csomópontokkal és élekkel lehet manipulálni. A második alkalmazás egy audioeditor, amelyik audiosorozatokat tud beolvasni, visszaadni és manipulálni. A harmadik pedig egy audio-video-elosztószolgálat, itt a *szerverrel* foglalkozunk. Ez a maga objektumorientált környezetében a rendelkezésre álló objektumokat és osztályokat működteti, amelyek tárolt videoinformációkat jelenítenek meg.

Ebben a három, igen különböző alkalmazásban elsőre nem sok *közöset* láthatunk. A tárolt adatok kezelése csomópontokban, audio- és videoobjektumokban mégis megoldható egységes módon egy MDBMS-en keresztül. Emellett nem kell törődnünk az adat-interfészek részleteivel: a DBMS fő feladata az elvonatkoztatás az adattárolás részleteitől. A multimédia rendszer 1–1 ábra szerinti szintek alapján az MDBMS egyrészről az alkalmazások, dokumentumok és absztrakciók között, másrészt a tárolási technológia, tömörítés és multimédia számítógépes technológia oldalán helyezkedik el. Emellett együttműködik az operációs rendszerrel és a kommunikációs komponensekkel.

A *Database Management System* **(DBMS) *feladata*** a szekunder tárolás és az adatkezelés módjától való elvonatkoztatás mellett az adatok megmaradásának biztosítása is. Az adatok túlélik a programokat és a processzeket. A többfelhasználós üzemnél az adatok konzisztens áttekintését minden időpontban az egyidejű adatbank lekérdezések szinkronizálásával biztosítják. Az átviteli elv hiba fellépése esetére biztosítja az adat- és integritásvesztés elleni védelmet; a korrekt állapot visszaállítható. Az adatbank lekérdezéseket lekérdező nyelvek, pl. az SQL formalizálják.

A *multimédia* szemlélet adatbankoknál ***nagy adatmennyiségek kezelését jelenti adott esetben különböző adathordozókon.*** Egy relációs adatbanknál egy Tupel egy attribútuma mozgóképsorozattal és a hozzá tartozó hanggal több Mbyte tárkapacitást igényelhet. Ezenkívül processzos adatok DBMS-sel való *feldolgozásánál* valósidejű követelmények is vannak. Ezeket a követelményeket az MDBMS tervezésénél figyelembe kell venni. Meg kell jegyezni, hogy külső tárak alkalmazásánál a valósidejű követelmények a DBMS-ből vannak *alapvetően definiálva*, majd implicite átvéve a külső készülékek által. Integrált rendszereknél – pl. a kompaktlemez technológiánál – a számítógépnél és így adott esetben a DBMS-nél kell a valósidejű követelményeket figyelembe venni. *(Az MDBMS és multimédia problematika mélyebb ismertetését az adatbank szempontjából l. [Meye91a]-nál. Az említett szempontokból sok szerepel itt és a rá következő munkákban.)*

A következő rész az adatbankrendszer feladatait mutatja be a multimédia szempontjából pontosítva, valamint nagy vonalakban ismerteti a rendszerarchitektúrát. Ezt követi a multimédiaadatok és a velük való műveletek áttekintése. Ezután következik az adatanalízis bemutatott eredményeinek és egy valóságos adatmodellen végzett műveleteknek az integrálása. A zárómegjegyzések a terület még csak várható eredményeire koncentrálnak.

10.2 Egy MDBMS leírása

Az MDBMS jól jellemezhető teljesítendő feladatain keresztül. Az MDBMS multimédia-specifikus feladatai levezethetőek a DBMS-sel szemben támasztott követelményekből a multimédiaadatok irányába való specializáció, illetve kiterjesztés útján.

1. *Megfelelő* ***tároló média***
   A multimédia-adatokat speciális jellemzőiknek megfelelően kell a különböző hordozókon tárolni és kezelni. Másodlagos tárolóként szerepelhetnek akár számítógépbe integrált alkatrészek, akár külső készülékek. Alkalmazhatók itt csak olvasható (mint a CD-ROM), valamint egyszer és többször írható tároló-eszközök.

2. *Leíró* ***keresési eljárások***
   Ha egy adatbankban szöveget keresünk, ezt különböző lekérdezések és megfelelő keresési eljárások útján tudathatjuk a DBMS-sel. A multimédiaadatok visszakeresése leíró, tartalomorientált, a „Kép a piros sálas hölgyről" formán alapul. Ez a keresési módszer minden médiára vonatkozik, beleértve a mozgóképet és az audiót is.

3. *Készülékfüggetlen* ***interfész***
   Az adatbank-alkalmazáshoz készülékfüggetlen interfészt kell megvalósítani. Itt pl. egy paraméterrel lehet megadni, hogy a következőkben beolvasandó audio- és videoadatokat már nem kell megváltoztatni. Ebből tudja az MDBMS, hogy a tároláshoz egyszer írható kompaktlemez *(Compact Disc Write Once)* jöhet számításba.

4. *Formátumfüggetlen interfész*
   Ugyanazon interfésznél az adatbank-alkalmazás nem igényel formátummegadást sem. A programozás formátumfüggetlenül megy, de szükség esetén hozzáférhetőek részletek a konkrét formátumról. Ezzel a programozó meghatározza az absztrakció szintjét annak előnyeivel és hátrányaival. Alkalmazásokat a jövőben is hozzáférhető formátumban állítsuk elő. A készülék- és formátumfüggetlenségnek köszönhetően új tárolótechnológiákat lehet alkalmazni anélkül, hogy a meglévő multimédia-adatbank alkalmazásokat meg kellene változtatni.

5. *Nézet (view) specifikus és* ***szimultán adathozzáférés***
   Ugyanazon multimédiaadatok különböző alkalmazási nézetekből (views) is rendelkezésre állnak. Így több felhasználó konzisztensen hozzáférhet egy dokumentum közös adataihoz együttes szerkesztés céljából.

6. *Nagy mennyiségű adat kezelése*
   A DBMS-nek képesnek kell lennie **nagy mennyiségű adat kezelésére**, amely adatok csak egy reláción vagy annak egy attribútumán keresztül kapcsolódnak [Feli90].

7. *Kapcsolattartó adatkezelés*
   Egy vagy több különböző média ***adatai között a kapcsolatnak*** meg kell maradnia a megadott specifikáció szerint. Az MDBMS kezeli ezeket a kapcsolatokat és alkalmazni tudja az adatok lekérdezésénél és kiadásánál. Így támogatja a dokumentumokban való *navigálást* és kezeli az egyes részek közötti kapcsolatokat. Különböző kapcsolatok különböztethetők meg [Meye91a]:
   - Az *attribútumkapcsolat* ugyanazon objektum különböző leírásait és előállításait jellemzi. Így pl. egy ornitológiai lexikonban minden madárnál a hang mint audiojel, a repülés mint mozgóképsorozat, különböző képek és egy leíró szöveg fog szerepelni. A relációs adatbank szempontjából minden madárhoz egy Tupel lesz hozzárendelve, amelyik a különböző ábrázolásokat mint attribútumokat fogja össze.
   - A *komponenskapcsolat* tartalmazza az adatobjektumhoz tartozó összes részt. Erre a kapcsolatra példa egy személygépkocsi-alkatrészkatalógus és egy dokumentum.
   - A *helyettesíthetőségi kapcsolat* ugyanazon információ különböző módon való prezentációit definiálja. Így lehet ábrázolni adott adatokra alkalmazott formulát pl. táblázatként, gráfként vagy animációként.
   - A *szinkronizációs kapcsolat* az adategységek közötti időbeli kapcsolatot írja le. Erre példa a szájmozgás-szinkronizáció a hang és kép között.

8. ***Valósidejűséget** támogató adatátvitel*
   A processzos adatokat valósidejűen kell be- és kiolvasni. Ez a processzos adatátvitel előnyt élvez az egyéb adatbank kezelési műveletekkel szemben. Itt visszalehet nyúlni a multimédia-képes operációs rendszer primitívjeihez.

9. *Hosszú **tranzakciók***
   Az MDBMS-ekben a hosszú, nagy adatmennyiségeket adó átviteleknek hibamentesen kell végbemennie. Hosszú átvitelre példa egy film tartalmának lehívása.

Az MDBMS eddig előállított prototípusai, amelyek a processzos médiumokat is támogatják, a multimédia eszközök és adapterek vezérlését, valamint a tárolást az MDBMS feladatává teszik. Ennek a pragmatikus megoldásnak az az előnye, hogy a multimédia rendszert teljes egészében az adott DBMS tudja vezérelni. Gyakran ezt a megoldást választják dokumentumok és hipertextek megtekintésére. Ügyeljünk arra, hogy más oldalról pl. a kommunikációs és konferenciarendszerek területéről a megoldások az MDBMS-t többnyire figyelmen kívül hagyják. Ennek ellenére itt is ugyanazokat a be- és kiviteli egységeket kell vezérelni. Ebben az esetben a vezérlés pl. a lokális és távolsági eszközöknek a kommunikációs rendszer részeként való integrációjával valósítható meg.

Az előző szakaszban megnevezett két multimédia szakterületen folyó operációs rendszer fejlesztések nem lesznek teljesen tekintetbe véve, mivel az eszközök vezérlése tulajdonképpen az operációs rendszer feladata lenne. A **DBMS**-nek jelen leírásban tárgyalt **architektúra modelljénél** minden be- és kiviteli eszköz az operációs rendszeren keresztül van csatlakoztatva (l. még az 1–1 ábrát a 16. oldalon):

- Az operációs rendszer interfészt biztosít minden eszköz vezérléséhez. Emellett az absztrakció foka különböző lehet.
- A DBMS, akárcsak a multimédia-adatok nélküli esetben, átveszi a tárolt adatok és a megfelelő készülékek absztrakciója előállításának a feladatát.
- A kommunikációs rendszer a kommunikáció absztrakcióját több számítógépre vonatkozó előírással az OSI rétegmodell szerint állítja elő.

Egy, a DBMS és az operációs és kommunikációs rendszerek fölötti síkon ezek a különböző absztrakciók egységesíthetők és például egy objektumorientált környezetben mint „toolkit" kínálhatók fel. Mindemellett egy alkalmazásnak hozzáféréssel kell rendelkezniük a mindenkori absztrakció különböző síkjaihoz.

Ez a modell nagy vonalakban megfelel a [Meye91a]-ban bemutatott MDBMS architektúra-javaslatnak. További finomítást jelent, ha minden kezelendő médiumhoz egy saját kezelőegységet rendelünk, ami médiumspecifikus szervert alkalmaz és egy közös lekérdező processzor áll rendelkezésére [Lock90]. Ezt a lekérdező processzort egy adatbank alkalmazási interfészmenedzser vezérli, ami egy alkalmazás számára egy adott adatbank képét mutatja. Belül ***„Long Field Data Server"***-ek is lehetnek, amelyek elvileg korlátlan méretű adatobjektumokat tesznek lehetővé. Egy további kezelőegység feldolgozza a kapcsolatokat különböző médiumok adatai között. Ez a durva strukturálás koncepció gyanánt szolgál, de nem tartalmaz implementációt a komponensek ugyanilyen felosztásával. A [Lock90]-ben mindazonáltal mindegyik kezelőegység számára editor funkciót is beterveztek. Ennek a jelen munkában tervezett struktúra szerint a dokumentumterületről vagy egy alkalmazásból való eszköznek kell lenni, mivel az editornak nagyon sok specifikus problémája összefügg a dokumentumarchitektúrával, az adatok megjelenítésével és bevitelével a felhasználói interfésznél. Mindenesetre az editornak az egyes médiumok kezelőegységeit tudnia kell használni adatkeresésre és -tárolásra.

10.3 Az adatok elemzése

Meyer-Wegener [Meye91a]-ban többek között kimerítő analízisét adta a multimédia-adatoknak, a súlypontot a tömörítetlen egyesképekre helyezve. Itt a tömörített adatok és egyéb médiumok problematikáját is be kell vonni. Mindkét esetben **absztrakt adattípusok**at kapunk eredményül. Ennél a vizsgálatnál két kérdés került előtérbe:

1. *Hogy vannak ezek az adatok strukturálva?*
   Tisztázni kell, mely járulékos információk tartoznak a tulajdonképpeni adatokhoz, amelyek lehetővé teszik a multimédia-adatok *feldolgozását* egy MDBMS-ben.
2. *Hogy lehet ezekhez az adatokhoz hozzáférni?*
   Ez alatt a megfelelő műveletek definiálását értjük. Lehet akár médiafüggő, akár a médiától független műveleteket definiálni. További lépésként el lehet képzelni egy osztályhierarchiát az objektumorientált programozás értelmében. Ekkor első lépésként az elsősorban a mindenkori médiumra releváns műveleteket kell megállapítani.

Ebben a szakaszban az első kérdéssel foglalkozunk, a második kérdést a 10.4 fejezetben a 260. oldalon tárgyaljuk részletesebben.

A multimédia-adatokat az adatbankban való hatékonyabb feldolgozás céljából *nyers adatok, regisztrálási adatok* és *leíró adatok* formájában lehet megadni. Az egyes adatbevitelek pedig mindig több ilyen adattípust tartalmaznak. Bevezető példaként tekintsünk egy egyesképet.

Nyers adatok

Ez a tömörítetlen egyeskép számos egyedi képpontból áll. A képpontok byte-ok és bitek alakjában jelennek meg a *nyers adatban*. Ezek azok a formázatlan információegységek, melyek mindenkor előállítják a szimbólumok, pontok, letapogatott értékek stb. hosszú sorozatát vagy mennyiségét. A *formázott* vagy *strukturált* adatokat változók, mezők, vagy attribútumok tárolják a hozzájuk rendelt értékek segítségével. Az adatok összetevői itt tulajdonságaikkal vannak jellemezve.

Példa:

```
A.Tanuló.Vezetéknév=Engler
A.Tanuló.Keresztnév=Clemens
A.Félév=8
```


A formázatlan vagy ***strukturálatlan*** **adatok** egy egységként jelennek meg, melyek tartalmát nem kell strukturálással feltárni.

Példa:

```
Clemens Engler úr a 8. félév hallgatója.
```

Regisztrálási adatok

Egy ilyen egyeskép kiolvasásához és korrekt interpretálásához ismerni kell egyebek között a kódolás részleteit és a kép nagyságát. Ebben a példában minden pontot 8 bit kódol fényességre (Luminance) és a két szín-differencia (Chrominance) jelre. A felbontás pl. 1024 × 1024 pont. Ezen *regisztrálási adatok* szükségesek a nyers adatok korrekt értelmezéséhez. A klasszikus DBMS általában *csak* jeleket és számokat ismer szigorú szemantikával, amelyek nem igényelnek kimerítő leírást. Az egyeskép-, audio- és videoadatok mindazonáltal nagyszámú attribútumot engednek meg kódolásnál és strukturálásnál. Ennek ismerete nélkül az adatokat csak nagyon nehezen vagy egyáltalán nem lehet korrekten visszaadni. A multimédia rendszernek több komponense és alkalmazása implicit ismeretet tételez fel, ami az egyes esetben még kielégítő. Az MDBMS-nek azonban, mint általánosan hozzáférhető szolgáltatásnak minden alkalmazás és a többi komponens számára is rendelkezésre kell állnia. Ezért a legkülönbözőbb formátumokat kell tudni kezelni. Több média esetén a regisztrálási adatok szemantikáját ki lehet terjeszteni egy vagy több médium adatobjektumai közötti kapcsolatok definiálására.

Leíró adatok

Szöveg- és számadattartalom keresése ma már igen hatékonyan végezhető, az egyesképek, audio- és videoinformációk keresése lényegesen nehézkesebb. Ezért minden adategységhez opcionális *leíró adatok* tartozhatnak. Egy egyesképnél a jelenetet le lehet írni szöveggel. Ezek a leíró adatok járulékos redundáns információt nyújtanak és megkönnyítik az adatok megtalálását egy későbbi keresés során. Lehetnek strukturálatlan, leíró szövegek vagy strukturált adatok.

„Text" médium

Első példaként tekintsük a *text* (szöveg) médiumot

1. A *nyers adatok* itt karaktersorozatok.
2. A *regisztrálási adatok* a kódolást írják le, pl. ASCII-ként. Ezenkívül adat kell a hosszúságra vagy egy „vége" jelet kell definiálni.
3. A *leíró adatok* lehetnek információk az elrendezésről (Layout) és a szöveg logikai szerkezetéről. Ezenkívül kulcsszavak is lehetségesek.

„Egyeskép" médium

Az egyeskép mint képpont, pontok seregéből áll.

1. A *nyers adatok* rendszerint pontmátrixként vannak megadva. Tömörített egyeskép előállhat transzformált képpontok *(pl. a diszkrét koszinusztranszformáció együtthatói a 2-dimenziós frekvenciatérben)* seregéből, vagy további tömörítéssel entrópiakódolt adatokból.
2. A *regisztrált adatok* tartalmazzák a kép szélességét és magasságát. Itt részletek is szerepelnek a kódoláshoz. Egy JPEG-eljárással tömörített kép esetében a modust adják meg először. Ez a kiterjesztett veszteséges modus, ami a diszkrét koszinusztranszformáción alapul. Ezenkívül pl. 8 bit/letapogatási érték kell a kép előkészítéséhez, a szekvenciális képfelépítéshez és a Huffman-entrópiakódhoz. Az alkalmazott táblázatokat a kvantálásnál és az entrópiakódolásnál kell megadni.
3. *Leíró adatként* meg lehet adni egyes vonalak, felületek, tárgyak, vagy a szituáció mint egész leírását. Példák erre: „Születésnap és szilveszter ünneplése Lizánál"; vagy

```
B.Alkalom = szilveszter
B.Dátum = 82.12.31.
B.Hely = ...
B.Név = Liza
B.Vezérszó 1 = szilveszterünnep
B.Vezérszó 2 = ...
```

„Video" médium

Egy ***mozgóképsorozat*** igen különböző időtartamú lehet. Ez a következő információkat tartalmazza:

1. A *nyers adatok* a legegyszerűbb esetben pontmátrixok sorozataként vannak definiálva. A mozgókép kódolásnál több képen át legtöbbször adatredukcióra használják ki a redundanciát (interframe kódolás) úgy, hogy nem tartalmazza minden egyeskép a kibontáshoz szükséges összes adatot. Lehetséges a változó sebességgel való adatáramlás is.
2. A *regisztrált adatok* egyebek között megadják a másodpercenkénti képek számát *(l. 5. fejezet, 73. old.)*. Egy a CCITT H.261 szerint kódolt adatfolyamban a QCIF *(Quarter Common Intermediate Format)* felbontásai: fényességre 176 × 144 képpont, színdifferencia-komponensre 88 × 72 képpont. Az MPEG szerint kódolt képeknél az állóképként való és a differenciaképként való kódolás közötti különbséget adják meg (minden teljes egyesképet 9 differenciakép követ). Lehetővé kell tenni a szabad hozzáférést az egyesképként kódolt képhez.
3. A *leíró adatok* egy jelenet leírását adják vissza, pl. a következőképpen: „Jan születésnapja a barátaival az óvodából. Perecevés kéz nélkül."

Az „audio" médium

Az audio médium egyes adatcsoportjait a következő séma szerint lehet osztályozni:

1. A *nyers adatok* PCM-kódolásnál a digitalizált mintavételi értékek. Tömörítés alkalmazásakor a tömörített értékeket tekintik nyers adatnak.
2. A *regisztrálási adatok* az audiokódolás jellemzőit mutatják. PCM esetén ezek a mintavételi sebesség, a kvantálási jelleggörbe és az egyes mintavételezési értékek felbontása. A tömörített audioadatok itt járulékos információt tartalmazhatnak, melyeket egy paraméterezett dekódernél alkalmaznak. Ezek gyakran már a nyers adatokban megvannak, amire példa az ADCPM.
3. A *leíró adatok* az audiocsatorna tartalmát rövid formában írják le. Zene esetén itt a mű, a szerző és az előadók állhatnak. Beszédszekvenciánál a rövid leírás mellett az egész szöveg is ott lehet.

Megjegyzések az adatanalízishez

Egy feltételezett adatanalízishez a következő megjegyzéseket kell tennünk, amelyek még a továbbiakban megmagyarázandó szempontokat tartalmaznak:

- A tárolómédián az audio- és videoinformációk gyakran csomagokban fordulnak elő. Ez a visszajátszásnál processzoros adatáramlást biztosít. A DBMS szempontjából a nyers-, reprezentációs- és leíró adatok formájában kombinált médiumokat is fel lehetne venni. Ezeket azonban az elemi video- és audioadatok egy kombinációjaként is lehet definiálni és a médiumokba egy hierarchiát bevezetni.
- Az alkalmazás szempontjából lehetségesnek kell lennie a formátumfüggetlen programozásnak. Ezért hasznos lehet, ha az alkalmazáshoz az interfésznél a hozzáférést mindig tömörítetlen adatokra definiáljuk, akkor is, ha ezek valójában tömörítettek.
- A tárolásnál az adatok hármas felosztásakor nyers, regisztrálási és leíró adatokra magasfokú elméleti tudás szükséges ahhoz, hogy értelmes leíró adatokat kapjunk. Ehhez kell járulnia egy definiálandó rendszertámogatásnak és a szövegleíráshoz kontextusfüggő keretek generálásának [ez találkozik a 10. fejezet (251. o.) értékelésével].

- Ha pl. egy utazási katalógus számára mozgóképeket vettek fel, és már bizonyos adatok rendelkezésre állnak, az új leíró adatok strukturálásánál a többi bevitel addigi leíró adatait mintának lehet venni.

10.4 Műveletek az adatokon

Az MDBMS-nek késznek kell lennie a 10.3 fejezet 256. oldalán bemutatott adattípusnak megfelelő összes műveletre. Ezeket a médiafüggő műveleteket a jövőben a lekérdező nyelvek, mint pl. az SQL, részének vagy kiterjesztésének kell tekinteni. Az adatbank szempontjából durván minden médium számára különböző műveleti osztályokat lehet tervezni: bevitel, kivitel, módosítás, törlés, összehasonlítás és kiértékelés.

Bevitelnél az adatokat beviszik pl. az adatbankba. Itt mindig csak a nyers- és regisztrálási adatokat viszik be. A leíró adatokra később kerül sor. Mozgókép és audio bevitelnél előfordulhat, hogy a hossz előre nem állapítható meg, és ezért az MDBMS-nek nehéz a megfelelő szerver és a lemez kiválasztása.

A nyers adatok ***kiadása*** a regisztrálási adatok interpretálása után következik. Egy a JPEG szerint kódolt egyes kép Huffman-táblája előbb átvihető a dekóderen. Ezután következik a nyers adatok átvitele.

A ***módosítás*** rendszerint csak a nyers adatokat érinti. Az egyesképtípus módosításait egy editornak kell átvennie. Videoadatokhoz rendszerint elektronikus vágás szükséges (be- és kiúsztatás). Audio-adatoknál a fel- és lekeverés mellett a hangerő, magasság, mélység és adott esetben a balance is állítható. Audio esetén ez a regisztrálási adatoknál történhet meg. Itt ezeket az attribútumokat időfüggő funkcióként definiálják, amely az adat kiolvasásakor lép működésbe. Érthetünk módosítás alatt egy másik formátumba való konverziót is, ekkor a regisztrálási adatok is érintve vannak. További variáció, ha átalakítás történik egyik médiumból a másikba. Pl. előfordulhat egy szöveg-beszéd átalakítás. A funkciót, éppúgy mint az editorokat, az MDBMS-en kívül látjuk. Egy ilyen *transzformáció* az adatok olvasásával, a külső átalakítással másik médiumba és végül az adatbankba való *beolvasással* valósul meg.

Az adatok ***törlésénél*** csak a konzisztenciára kell ügyelni, a nyers, regisztrálási és leíró adatokat ezután töröljük.

Az MDBMS-sel kapcsolatos kívánalmak jó része előzőleg eltárolt adatok megtalálására vonatkozik. Ez ***összehasonlítás*** segítségével lehetséges. Itt, akárcsak a text médiumnál, az adott médiumban a keresett mintát összehasonlítják a tárolt adattal. Ez a fajta keresés azonban nem különösebben eredményes. Egy másik módszer a nyers adatokon alapuló mintafelismerést alkalmaz. Itt azt állapítják meg, hogy egy keresett minta szerepel-e a nyers adatok között. Ezen eljárás eddig

elért hatékonysága egy általánosan alkalmazandó MDBMS számára túl kicsi, vagy csak bizonyos alkalmazásokra korlátozódik. Lehet összehasonlítást végezni a megfelelő formázott leíró adatok alapján is. Itt minden audioszekvenciát egy egyértelmű, maximum 16 karakteres névvel és a keletkezési dátummal lehet azonosítani.

További összehasonlító módszerek a tartalomorientált leíró adatokat használják fel *(itt [LuMe89, LuMe90, Meye91a] eredményei használhatók)*. A felhasználó itt nominál kifejezéseket visz be korlátozott szókinccsel. Az MDBMS ezeket belül állításokká (predikátum) alakítja. Szinonimákat is lehet kezelni és kihasználni. Ez az elv így tartalomorientált keresést tesz lehetővé, amit eddig az egyesképekre alkalmaztak és ami nehézség nélkül átvihető minden médiumra.

A ***kiértékelés*** célja a nyers és regisztrált adatoknál a megfelelő leíró adatok előállítása. Pl. hasznosítani lehet a fakszimile dokumentumok eltárolásánál a gépi szövegfelismerést (OCR = *Optical Character Recognition*). Többnyire az alkalmazónak kell explicit módon bevinni az adatokat.

10.5 Integráció az adatmodellben

Az MDBMS megvalósításánál és alkalmazásánál alapvető szempont az adatmodell. Ahogy kiderült, elsőrendű fontossága van az alkalmazandó adattípusoknak és a műveleteknek. Következő lépésként ezek az adatfajták és műveletek beintegrálhatók akár egy relációs modellbe, akár egy objektumorientált sémába [SmZd87].

Az összes médium absztrakt adattípusait ezért leíró attribútumokkal lehet definiálni a mindenkori formátumhoz és tulajdonságokhoz. Példaként tekintsük a *tömörítetlen video* médiumot, az alább megadott attribútumokkal. Ezek a műveletekkel lekérdezhetők, és adott esetben megváltoztathatók.

```
heigth:            Integer;                    480
width:             Integer;                    640
encoding:          uncompressed;               YUV
stream_encoding:   s_mode;                     PAL
pixel_depth:       Integer;                    (8, 8, 8)
rate               signed Integer;             25
colormap_size:     Integer
colormap:          Array;                      (...)
pixel:             Structure (Y, U, V) of Bit
image:             Array (,) of Pixel
video:             timed_List of Image
```

A multimédia-adatbank realizálására a legegyszerűbb módszer a *relációs modell*be való beágyazás. Ennek legfőbb előnye az eddigi adatbank alkalmazásokkal való kompatibililitás. A különböző médiumokhoz az értéktartomány *(domains)* a legelőször definiált adattípus. Ily módon az attribútumok lehetnek pl. *video* vagy *audio* típusok.

1. relációs sématípus

A következő példában szemléltetésként egy sportiskola tanulóinak szabadon választott tárgya, a szertorna szerepel mint reláció. Az egyes relációk itt többféle médiumot tartalmazhatnak. Ebben az 1. relációs séma típusban a mozgóképek száma megfelelő attribútumok mindenkori mennyiségével van megadva.

```
tanuló    (anyakönyvi szám   Integer,
           név                String,
           kép                Image,
           ...                ...
           gyűrűk             Video,
           ...                ...
           rudak              Video)
```

2. relációs sématípus

A 2. relációs sémával változtatható számú bevitel definiálható. A következő példában minden tanulónál az anyakönyvi számon keresztül vannak az egyes szakok (pl. könnyűatlétika és torna) azonosítva. Mindazonáltal bizonyos lekérdezéseknél fáradságos összekapcsolási műveletek szükségesek.

```
tanuló          (anyakönyvi szám  Integer,
                 név               String,
                 tantárgy          String,
                 ...               ...)

könnyűatlétika  (anyakönyvi szám  Integer,
                 gerelyhajítás     Video,
                 távolugrás        Video,
                 ...               ...)

torna           (anyakönyvi szám  Integer,
                 rudak             Video,
                 gyűrűk            Video,
                 ...               ...)
```


3. relációs sématípus

A relációként rögzített számú attribútum és a változó számú bevitel mellett egy bevitel egyidejűleg több relációhoz tartozhat. Ezt nevezzük 3. típusú relációs sémának. Elképzelhető olyan eset, amikor egy tanuló szertorna-videofelvétele hozzárendelhető mind az illető tanulóhoz, mind a tanintézeti alkalmazáshoz a sportágak tipikus hibáinak analízisére.

```
tanuló          (anyakönyvi szám  Integer,
                 név               String,
                 tantárgy          String,
                 ...               ...)

könnyűatlétika  (anyakönyvi szám  Integer,
                 minősítés         Integer,
                 gerelyhajítás     Video,
                 távolugrás        Video,
                 ...               ...)

analízis        (minősítés        Integer,
                 hibaminta         String,
                 kommentár         Audio,
                 ...               ...)
```

A mai MDBMS-ben gyakran *rejtőznek* az audio- és videobevitelek mögött filenevek. A prezentációs komponens aztán ezekhez a külső adatcsoportokhoz fordul.

Az objektumorientált adatbankban a relációk helyett az innen levezetett objektumok osztályai vannak. Ezek egy hierarchiában egymással kapcsolatba hozhatók, és így egy specializáció jöhet létre. Egy objektum a relációs modellben egy értéknek felel meg. Ezt némely MDBMS-ek igen különböző osztályhierarchia fajtákkal alkalmazzák. E sokféleségből ma még nem alakult ki általánosan érvényes osztályhierarchia. Ezekben a rendszerekben igen jól lehet az adatok között *navigálni,* és az ábrázolási lehetőségek igen flexibilisek. Ellentétben a relációs modellel azonban az igénylők számára fontos mennyiségi műveletek nincsenek kielégítően támogatva.

10.6 Zárómegjegyzések

Mai ismereteink szerint az MDBMS fejlesztése egy közbenső stádiumban van, egyes területek még jelentős kidolgozásra várnak.

A legtöbb rendszer szoros kapcsolatban van egy multimédia-alkalmazással vagy alkalmazási területtel. Az eddigi fejlesztések sajnálatos módon a dokumentumokkal vagy hipertextekkel vannak szoros kapcsolatban, míg szinte minden konfe-

jük, amelynél a tulajdonképpeni tartalmat csak később másoljuk a helyére. A következő példában egy ékezetet mint *Entity Reference*-et definiálunk:

```
m&ee;reg ... aminek méregnek ... kell lennie
```

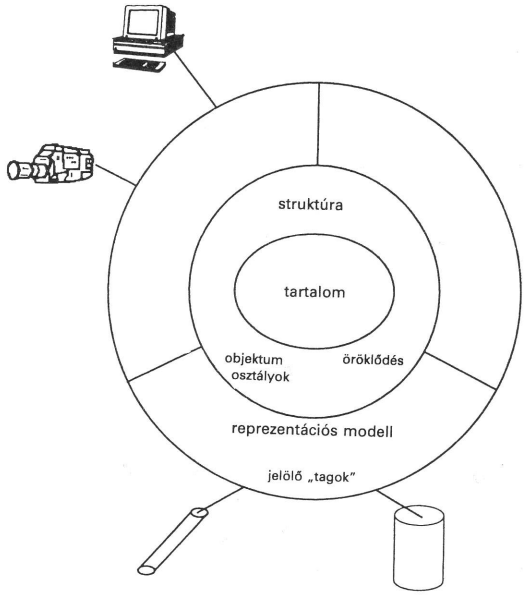
Egy ***Markup Declaration***-t (jelölő deklarálás) a struktúrák (osztályok) szabályainak definíciójához is alkalmazhatunk. A következő példa a *Papers* c. cikk felépítését rögzíti:

```
<!ELEMENT paper        (preamble, body, postamble)>
<!ELEMENT preamble     (title, author, side)>
<!ELEMENT title        (#CDATA)> -- character data
<!ELEMENT body (...)>
```

- A ***Processing Instruction***-ok segítségével írhatunk egy szövegben utasításokat más programok számára, amiket például a formázóprogramnak szánunk. Így különböző médiumokat is beiktathatunk.

Az SGML a tagok nyelvtanán keresztül egy mindig betartandó szintaxist definiált. Az SGML nem határozza meg ezeknek a tagoknak a jelentését.

Az SGML *információ- és dokumentumachitektúráját* a 293. oldal 11–15 ábrája mutatja. Az SGML a maga „tagjaival" egy reprezentációs-modellel rendelkezik. A struktúra definíciójához objektumok, osztályok és öröklődés használható.



11–15 ábra
SGML dokumentumarchitektúra. Súlypont a reprezentációs modellen

11.3.2 SGML és multimédia

Az SGML szabvány eddig a *multimédia adatokat* közvetlenül csak grafika formájában támogatta. Egy CGM *(Computer Graphics Metafile)* formátumú grafika beágyazható egy SGML dokumentumba. További médiumokról semmit sem mond a szabvány [ISO8879].

```
<!ATTLIST video  id      ID      #IMPLIED>
<!ATTLIST video  synch   synch   #IMPLIED>
<!ELEMENT video  (audio, movpic)>
<!ELEMENT audio  (#NDATA(> -- non-text media
<!ELEMENT movpic (#NDATA(> -- non-text media
...
<!ELEMENT story  (preamble, body, postamble)> :
```

Konkrét adatokra való hivatkozás csak # **NDATA** formájában történhet. Itt az adatok a legtöbbször egy külső, elkülönített file-ban vannak eltárolva. A fenti példa a video definícióját mutatja, ami hangból és mozgóképből áll.

A multimédia-információegységeket is alkalmas módon kell bemutatni. Nagyon fontos a komponensek közötti szinkronizáció. Ezen dolgoznak a HyTime-ban [Gold91]-ben és az MHEG-ben [MHEG93].

11.3.3 Befejező megjegyzések

A kommunikációhoz egy szabványosított dokumentumcsere szükséges. Feladó és címzett ekkor mind időben, mind térben szétválasztható. A dokumentumok gyakran automatikusan kerülnek továbbfeldolgozásra. Ez egy közös információs környezetet (kontextust) követel meg: a szintaxis átvitelre kerül, a szemantikáról az SGML-ben külön kell megállapodni. Ezeknek megállapodásoknak a bázisát a *dokumentum-típus definíciók* ***(Dokument Type Definitions)*** képezik.

Az SGML mint szabvány mindig a mai formájában fog megmaradni [ISO8879], de kidolgozzák a kiegészítéseit:

Szükséges egy szabványosított layoutszemantika. Ez leegyszerűsíti a felhasználó csoportok megegyezéseit. A *Document Style Semantics and Specifikation Language* **(DSSSL)** egy prezentációs célú kiegészítés. A betűkészleteknél még egy információcserének kell megtörténnie a digitális írásformában.

Dolgoznak a Postscript bázisán egy *szabványos lapleíró nyelven (Standard Page Description Language* ***SPDL***).

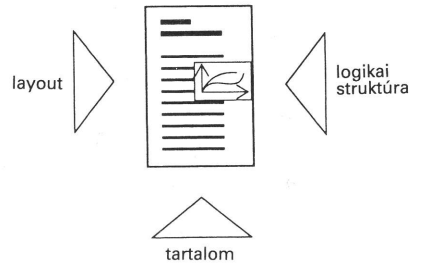
A multimédiába a hivatkozásokat mint nem látható kiterjesztéseket kell bevinni. Itt a zene leírásának *Standard Music Description Language* ***SMDL*** és a **HyTime** bővítése is használható.

11.4 Az ODA dokumentumarchitektúra

Az ***Open Dokument Architecture***-t ODA [ISO8613] néhány évvel ezelőtt még *Office Document Architecture*-nek hívták, mert leginkább az irodatechnikai alkalmazásokat támogatja. Ennek a dokumentumarchitektúrának az a célkitűzése, hogy a dokumentumok nyílt rendszereken való cseréjét, feldolgozását és bemutatását támogassa. Az ODA-t leginkább az Európa központú számítógépipar támogatja.

11.4.1 Néhány ODA részlet

Az ODA lényeges ismérve a **tartalom, logikai struktúra** és a **layout-struktúra** közötti különbségtétel. Ez ellentétben áll az SGML-lel, aminél csak egy logikai struktúrát és a tartalmat definiálják. Az ODA a szintaxis mellett tehát a szemantikát is definiálja. A 11–16 ábra az ODA ezen három vonatkozását mutatja egy dokumentumon. Ezt egyazon dokumentum 3 ortogonális nézeteként képzelhetjük jól el. Mindegyik nézet csak a maga szempontjaival foglalkozik, együtt adják ki a tulajdonképpeni dokumentumot.



11–16 ábra
ODA-, layout- és logikai nézet

Content Portions (tartalmi részek)

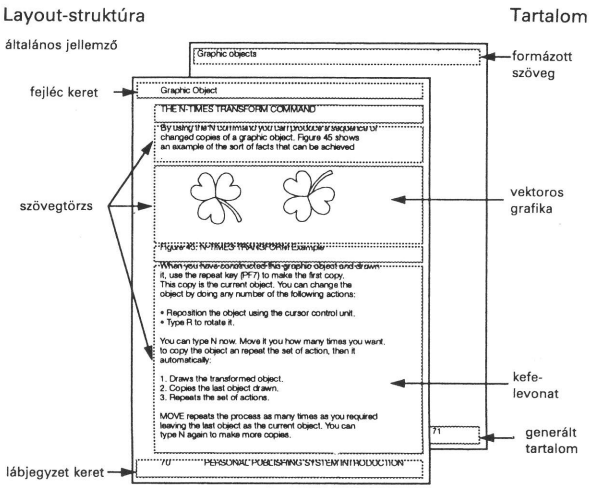
A dokumentum tartalma *Content Portionokból* áll. Ezeket a médiának megfelelően manipulálják.

Egy tartalmi architektúra minden médiára leírja az elemek specifikációját, a lehetséges hozzáférési funkciókat és az adatok kódolását. Egyes elemek a logikai adategységek (LDU-k), amelyeket minden egyes médiumra rögzíteni kell. A hozzáférési funkciók az egyes elemek manipulálására szolgálnak. Az adatok kódolása rögzíti a bitekre és byte-okra történő leképzést.

Léteznek tartalmi architektúrák *szöveg, vektor-* és *rasztergrafikai* médiumokra. A *szöveges* médiumok tartalmát a *Character Content Architecture* definiálja. A *Geometric Graphics Content Architecture* lehetővé teszi az olyan egyesképek tartalmának leírását, amelyek egyedi grafikus objektumokból állnak. Ez hasonló, mint a CGM. A *Raster Graphics Content Architecture* segítségével az egyesképeket képpontonként írják le. Ez egy fakszimileszerű bitmap lehet.

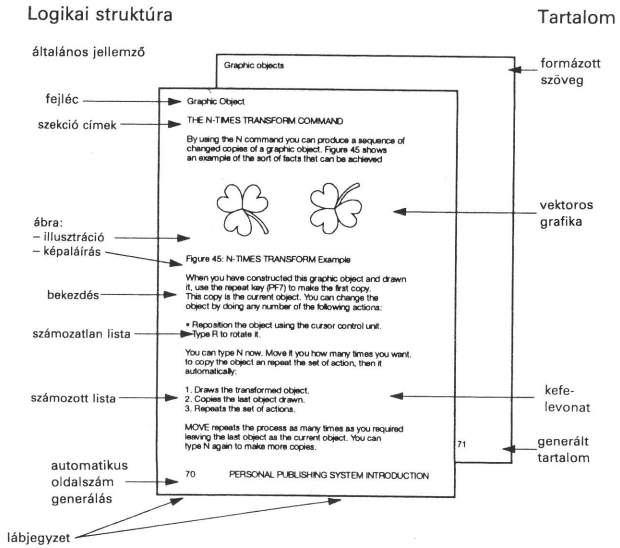
Layout-struktúra és logikai struktúra

A struktúra- és a bemutatásimodellek az információarchitektúrának megfelelően az információegységek együttműködését írják le: egy fajta metainformáció, ami megkülönbözteti a layoutot és a logikai struktúrát.



11–17 ábra
Layout nézet és egy ODA dokumentum tartalma. Példa a [Fis$89]-ből az ott megadott fogalmakkal

A *layout-struktúrát* lényegében a dokumentum kialakítása rögzíti. Ez a képernyőn vagy a papíron való kétdimenziós megjelenésre vonatkozik. A prezentációs modell egy fa. A 11–17 ábra együtt mutatja egy dokumentum tartalmát a layout struktúrájával. Itt keretek *(Frames)* segítségével rögzítik a layout elemek helyzetét és nagyságát, például meghatározzák az oldalméretet és a bekezdések írásmódját.

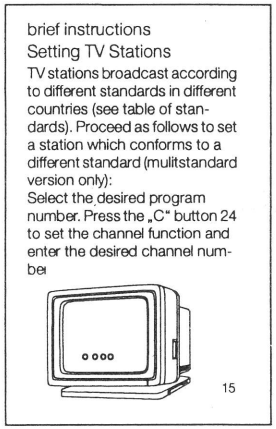


11–18 ábra
Logikai nézet és egy ODA dokumentum tartalma

A *logikai struktúra* a tartalom tagolását tartalmazza. Itt rögzítik a szakaszokat és az egyes címeket egy faszerű strukturális modellnek megfelelően. A listákat bejegyzéseikkel együtt definiálják.

```
paper    = preamble body postamble
body     = chapter1 chapter2
chapter1 = heading paragraph ... picture ..
chapter2 = heading paragraph picture paragraph
...
```

A fenti példa egy cikk logikai struktúráját írja le. Minden cikk egy *Preamble*-ből, egy törzsből *(Body)* és egy *Postamble*-ből áll. A törzs két fejezetet tartalmaz, amelyek mindegyike címmel kezdődik. Ennek a logikai struktúrának minden eleméhez tartozik tartalom.

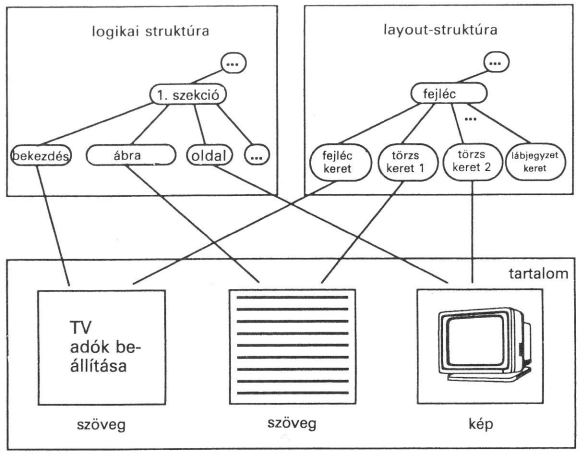


11–19 ábra
Egy példa az ODA dokumentumra

A 11–19 ábra példája konkrétan magyarázza a tartalom, a logikai és a layout-struktúra közötti kapcsolatot. A 299. oldal 11–20 ábrája a 11–19 ábra dokumentumának tartalmát, logikai és layout-struktúráját mutatja.

A 11–19 ábra egy címből, leíró szövegből és egy ábrából áll. A cím és a leíró szöveg a *Character Content Architecture* tartalmi architektúrához sorolandó. Az ábra a *Raster Graphics Content Architekture* tartalmi architektúrához tartozik (l. a fenti ábra alját).

A fenti példa logikai struktúrája minden szekcióhoz legalább egy címet, egy szakaszt és egy ábrát rendel. Ez a 299. oldal 11–20 ábráján jobbra fönt látható.



11–20 ábra
A tartalom, a logikai és a layout-struktúra közötti kapcsolat a 298. oldal 11–19 ábrájának példájára

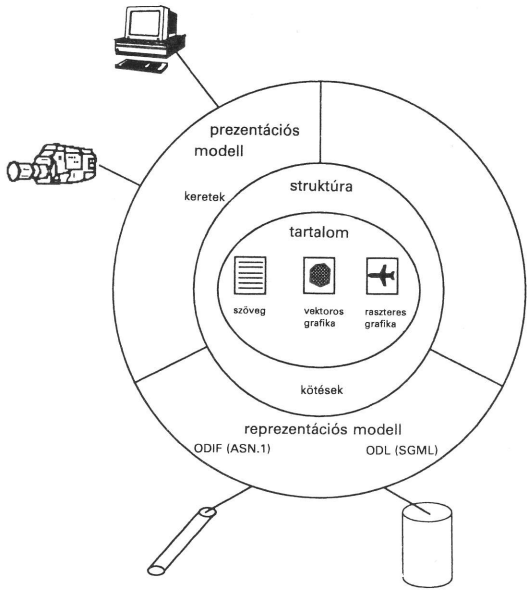
A 11–20 ábra jobb felső részén a dokumentum layout-struktúráját ábrázoltuk. Ez többek között több *keretből (Frame)* áll, amelyek egy kétdimenziós felületen egy előre megadott módon vannak elrendezve.

A különböző területek elemei között húzódó egyes vonalak a hozzárendeléseket mutatják. A logikai fa (bázisobjektum) és a layout fa minden egyes „leveléhez" egy *Content Portion* van hozzárendelve.

Az ODA a következő layout- és logikai struktúrákat különbözteti meg:

- Az *általános logikai* és az *általános layout-struktúra* az ODA-ban egy készlet általánosan érvényes szabályt tartalmaz *(Alapértékek)*. Itt például megadhatjuk egy bekezdésre a Left Hand Offset = 0 értéket.
- A *specifikus logikai* és *specifikus layout-struktúra* egy konkrét dokumentumot ír le, hivatkozva az általános struktúrára. Például egy konkrét bekezdést Left Hand Offset = 1 cm értékkel definiálhatunk. A következő példa egy specifikus layout-objektumot mutat ODA-ban:

```
object type:                block
object identifier:          title
position:                   (x=111, y=222)
dimensions:                 (height=333, width=444)
content architecture class: formatted character
                            content architecture
content portions:           (HIVATKOZÁS A SZÖVEGRE)
```



11–21 ábra
ODA információarchitektúra. Struktúra, tartalom, prezentációs és reprezentációs modell

Az ODA információsarchitektúra tartalmazza a 11–21 ábrán bemutatott együttműködő modelleket.

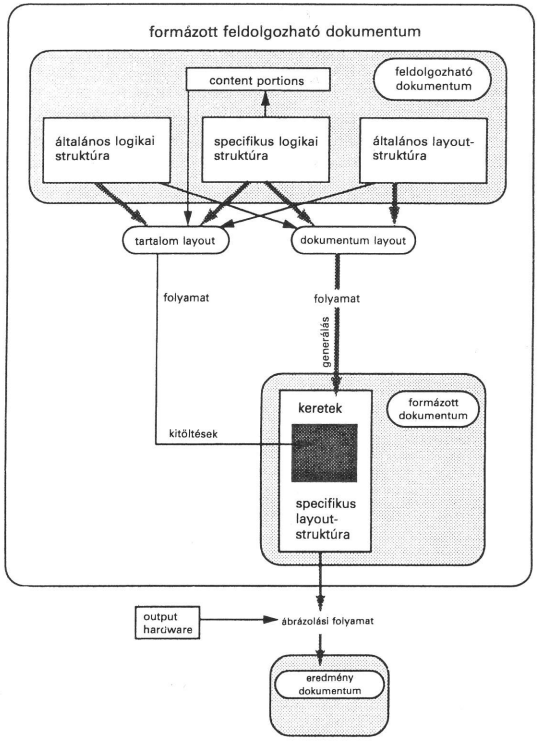
A struktúra- és prezentációsmodellek alapvető leíró eszköze az egyes csomópontokra vonatkozik, amelyek egy egészként képezik a dokumentumot. A dokumentum faként látható. Minden csomópont (és a dokumentum mint egész is) egy *Constituent* vagy objektum. A csomópont sok attribútumot tartalmaz, amelyek a csomópont tulajdonságait jellemzik.

Emellett a csomópont maga egy konkrét értéket is tartalmaz vagy más csomópontok kapcsolatát definiálja. A következő kapcsolódások és műveletek megengedettek:

11–1 táblázat
Kapcsolatok a csomópontok között

Sequence	minden fiókcsomópont sorrendbe rendezett
Aggregate	a fiókcsomópontok nincsenek sorrendbe rendezve
Choice	egy fiókcsomópont követi
Optional	egy vagy egy sem (művelet)
Repeat	egyszer... tetszőleges gyakran (művelet)
Optional Repeat	0... tetszőleges gyakran (művelet)

Ha egy, a 268. oldal 11–3 ábrájának megfelelő feldolgozási folyamatot veszünk, akkor ezt ODA-ban úgy írhatjuk le, mint ahogy azt a 11–22 ábra mutatja.



11–22 ábra
ODA dokumentumfeldolgozási folyamat az információtól a prezentációig

Itt leegyszerűsítve a szerkesztés, formázás *(Document Layout Process* és *Content Layout Process)* és a tulajdonképpeni megjelenítés *(Imaging Process)* között teszünk különbséget. A mai WYSIWYG szerkesztők ezt egy lépésben tartalmazzák. Emellett fontos, hogy a feldolgozási folyamat egy lineáris reprodukciós folyamatból indul ki. Ez eddig csak nagyon feltételesen volt egy hypertext rendszer dokumentumarchitektúrájaként alkalmas. Ezért ma egy *Hyper-ODA*-n is dolgoznak.

Dokumentumcserére különböző, a 301. oldal 11–22 ábráján bemutatott ***Dokumentumarchitektúra-osztályok*** alkalmazhatók:

- Egy *formázott dokumentum (Formatted Dokument)* tartalmazza a specifikus layout-struktúrát és adott esetben az általános layout-struktúrát is. A címzett közvetlenül ki tudja nyomtatni, illetve meg tudja jeleníteni. Emellett viszont a címzett nem tudja megváltoztatni.
- Egy *feldolgozható dokumentum (Processable Dokument)* a specifikus logikai struktúrából, adott esetben az általános struktúrából és adott esetben az általános layout-struktúrából áll. A címzett nem tudja közvetlenül kinyomtatni illetve megjeleníteni. Lehetséges a tartalom változtatása.
- Egy *formázott feldolgozható dokumentum (Formatted Processable Dokument)* egy kevert forma. Közvetlenül kinyomtatható illetve megjeleníthető és a címzett a tartalmat is meg tudja változtatni.

Egy ODA dokumentum kommunikációjára a 300. oldal 11–21 ábráján bemutatott reprezentációs modellt használják. Ez vagy az *Open Document Interchange Format* **ODIF** (az ASN. 1 bázisán) vagy az *Open Document Language* **ODL** (az SGML bázisán) lehet.

A 300. oldal 11–21 ábrája szerint az ODA-ban még hiányzó modellen, a manipulációs modellen, a *Document Appliction Profiles*-on **(DAP)** már dolgoznak. Ez egy, a dokumentum manipulálására szolgáló ODA kiterjesztés. Az ODA terjedelme miatt erre 3 szintet definiáltak, amelyek egyenként az ODA egy részhalmazát írja le *(Text Only, Text + Raster Graphics+ Geometric Graphics, Advanced Level)*.

11.4.2 Az ODA és a multimédia

A multimédia a térbeli bemutatási dimenziók mellett az *időt* is mint egy dokumentum lényeges alkotórészét feltételezi. Ha az ODA-nak folytonos médiumokat kell tartalmaznia, akkor a meglévő szabványok kiterjesztése szükséges, azaz a *multimédia* még nem része a szabványnak. Az összes következő megvalósítás csak utal

a lehetséges bővítésekre, amelyek az ODA-ban nem feltétlenül ezen a módon fognak jelentkezni.

Tartalmak

A *Content Portions*-ból *Timed Portions* lesz [Herr90h]. Sőt emellett az időtartam a priori módon nem is rögzíthető. Ekkor *Open* ***Timed Content Portions***-ról beszélünk. Vegyük egy animáció példáját, ami a prezentáció időpontjában külső eseményektől függően generálódik. Ehhez számíthatók a prezentáció ideje alatt kamerával felvett mozgóképek is, amennyire ezeket még dokumentumnak nevezzük. Egy *Closed Timed Content Portions* esetén az időtartam fix. Egy előzetesen eltárolt zenedarab példa lehet erre.

Struktúra

Az objektumok közötti műveleteket ki kell bővíteni egy időbeli dimenzióval. Ezért a *v* apacsomópontban a k1, k2 fiókcsomópontokhoz az időkapcsolatot is megadják. Példák erre a 13–10 ábrán részben, és a 13.8 fejezetben a 354. oldalon leírt kapcsolatok:

- előtt, *before*
  k1 k2 előtt, k1 + k2 > v
- egymásra következő, *meets*
  (k1, k2) k1 pontosan k2 előtt, k1 + k2 = v
- átlapoló, *overlaps*
- alatt (időben), *during*
- induló, *start*
- befejeződő, *end*
- közbenső, *middle*
- időpontjában, *at*
- egyenlő, *equal*

Tartalomarchitektúrák

Az audióra és a videóra járulékos tartalomarchitektúrákat kell definiálni. Rögzíteni kell az ennek megfelelő elemeket, az LDU-kat. A hozzáférési funkcióknál specifikálni kell a médiafolyamok *vezérlésére* szolgáló általános érvényű funkciók tömegét. Ilyen funkciók például a *Start* és a *Stop*. Sok funkció gyakran nagyon esz-

közfüggő. Az egyik legfontosabb szempont az adott ODA-implementáció különböző rendszerek közötti kompatibilitásának biztosítása.

Az adatok kódolásánál figyelembe kell venni a létező de jure és de facto szabványokat. Külön megemlítjük az ECMA, CCIR, CCITT és ISO szabványokat (például a JPEG-et az egyesképekre, az MPEGet a videóra és audióra, a H.261-et videóra és a CD-technológiát). Pontosabban figyelembe kellene venni egy *nyitott* architektúra lehetőségeit, amivel további fejlesztéseket már ma figyelembe lehetne venni, és nem kellene a szabványokat ismét változtatni, illetve bővíteni.

Logikai struktúrák

A logikai struktúráknál is terveznek a multimédia érdekében bővítéseket. Így egy film is kaphat logikai struktúrát. Ez például egy fa lehet a következő komponensekkel:

1. Előkészítés
   - Bevezető filmrészlet
   - A második filmrészlet szereplői
2. 1. jelenet
3. 2. jelenet
4. ...
5. Lecsengés

Egy ilyen struktúra gyakran nagyon kívánatos volna a felhasználó számára. Így célzatosan lehetne tartományokat átugrani és másokat pedig megjeleníteni. A kínálati oldalról ez például a hirdetések célzott kivágása miatt nem feltétlenül kívánatos. Egy ilyen logikai struktúra az ODA koncepcióival minden további nélkül definiálható.

A különböző objektumok közötti idő **időbeli hivatkozás** a logikai struktúrák definíciójánál is fontos. Ezt az időbeli hivatkozást a logikai struktúra kontextusában csak annyira kell specifikálni, amennyire ez a tartalomra nézve fontos, hogy ne keletkezhessen hiba. Így a felirat és a jelenet mint egész közötti kapcsolat definiálható, de a logikai struktúrában nem rögzíthető a pontos időpont. Ezt számítsuk inkább a layout-struktúrához.

Egy különböző objektumok közötti **helyhivatkozás**t úgy definiálhatunk, mint más diszkrét médiumok között.

Egy ODA multimédia dokumentumban lehetségesnek kell lennie, hogy egy fastruktúra helyett egy gráfot definiáljunk, amit egy általános hypermédia technika megengedne.

Layout-struktúra

A layout-struktúrának szüksége van a multimédia bővítésekre. Itt egy mozgókép és egy hang időbeli hivatkozását konkretizálni kell és meg kell választani a „Mikor játsszuk le?" „Melyik helyen?" és „Milyen attribútumokkal és minek a függvényében?" típusú kérdéseket. A helyhivatkozás például az audio médiumok objektumainál relatív és abszolút pozíciókat is rögzíthet. Azonkívül a hangerőt és az összes további attribútumot és függést meg kell határozni. Különösen a folytonos médiumoknál figyelembe kell venni az *interaktivitást*. A dokumentum nemcsak egy papír, a *lineáris* feldolgozási folyamat mind inkább *interaktív* feldolgozási folyamattá alakul. Emellett nem szabad az ODA-nál az *Imaging Process*-t szem elől téveszteni.

Ha az ODA minden folytonos médiumot integráló kibővítését összefoglaljuk, akkor a 267. oldal 11–2 ábráján bemutatott multimédia-dokumentumarchitektúrát kapjuk.

11.4.3 Zárómegjegyzések az ODA-hoz és az SGML-hez

Ha összehasonlítjuk az ODA-t az SGML-lel, akkor feltűnik, hogy az SGML csak a szövegmegjelölés szintaxisát definiálja, aminek nem meghatározott a jelentése. Az ODA egy rögzített szemantikát tartalmaz a dokumentumok leírására.

Az ODA lehetőséget nyújt egy, a folytonos médiumokat integráló, nyílt szabvány realizálására. Ezzel lehetővé válna, hogy multimédia-dokumentumokat cseréljünk, s így, mint ma világszerte, a szövegeket az elektronikus postával elküldjük. Ehhez azonban még az előző szakaszban leírtakat át kell ültetni a gyakorlatba. Az ODA-ban eddig sok más szempontot is csak elégtelenül vettek figyelembe.

Az ODA figyelembe veszi a *biztonságot* és a *színeket a dokumentumokban*. Ehhez biztosítani kell a felfelé kompatibilitást. A szöveg és grafika mellett a jövőben táblázatokat és adatokat is támogatni fognak a dokumentumokban. Emellett ezeket az adatokat ki kell tudni cserélni a dokumentum és egy táblázatkezelő között, valamint át kell tudni alakítani az adatokat szöveggé.

A most következő megjegyzések az ODA szabványosításánál a jövőben figyelembe veendő szempontokra vonatkoznak. *(Ezek a gondolatok lényegében a heidelbergi belső vitákból származnak.)* Részleges dokumentumok nem teljes dokumentumok, amelyek *külső hivatkozásokkal* rendelkezhetnek. Ezzel a dokumentumokat számítógéphatárokon keresztül definiálhatunk. A képleteket fel kellene venni az ODA-ba, mint annak egy részét. Be kellene vezetni egy verziókezelést; további tartalmi struktúrákat kellene definiálni. Külön figyelembe kellene venni más szabványosító grémiumoknak az SGML-re (mint az SMDL és HyTime) és az MHEG-re vonatkozó munkáit és figyelemre méltatni mindenekelőtt a tömörítési eljárásokat (mint pl. az MPEG) is.