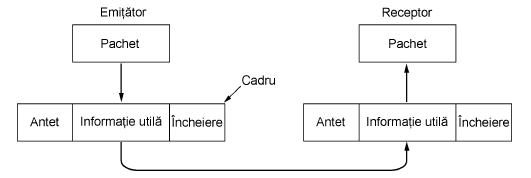
2. Nivelul legatura de date

2.a. Caracteristici ale proiectarii nivelului legaturii de date

Nivelul legatura de date are un numar de functii specifice pe care trebuie sa le indeplineasca. Aceste functii includ:

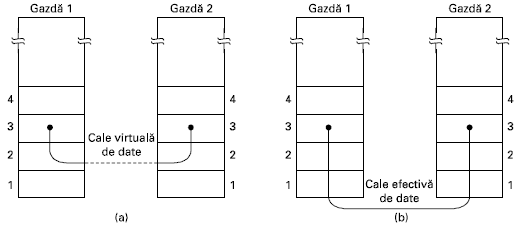
1. Furnizarea unei interfete bine-definite catre nivelul retea
2. Tratarea erorilor de transmisie
3. Reglarea fluxului cadrelor in asa fel, incat receptorii lenti sa nu fie inundati de catre emitatori rapizi

Pentru a indeplini aceste scopuri, nivelul legatura de date primeste pachete de la nivelul retea, pe care le incapsuleaza in **cadre** in vederea transmiterii. Fiecare **cadru** contine un antet, un camp de informatie utila pentru pachet si incheiere, dupa cum se vede in figura de mai jos. Gestionarea cadrelor reprezinta esenta a ceea ce face nivelul legatura de date.



# Servicii oferite nivelului retea

Functia nivelului legatura de date este sa ofere servicii nivelului retea. Principalul serviciu este transferul datelor de la nivelul retea al masinii sursa la nivelul retea al masinii destinatie. La nivelul retea al masinii sursa exista o entitate, sa-i spunem proces, care trimite biti catre nivelul legatura de date, pentru a fi transmisi la destinatie. Funcia nivelului legatura de date este sa transmita bitii spre masina destinatie, pentru ca acolo sa fie livrati nivelului retea, asa cum se arata in figura (a) de mai jos. Transmisia efectiva urmeaza calea din figura (b).



Nivelul legatura de date poate fi proiectat sa ofere diferite servicii. Serviciile oferite pot varia de la sistem la sistem. Trei posibilitati de baza, oferite in mod curent, sunt:

1. Serviciu neconfirmat fara conexiune
2. Serviciu confirmat fara conexiune
3. Serviciu confirmat orientat-conexiune

# Incadrarea

In vederea furnizarii unui serviciu nivelului retea, nivelul legatura de date trebuie sa utilizeze serviciul furnizat de catre nivelul fizic. Sarcina nivelului fizic este sa primeasca un flux de biti si sa incerce sa-l trimita la destinatie. Nu se garanteaza ca acest flux de biti nu contine erori. Numarul de biti receptionati poate fi mai mic, egal cu, sau mai mare decat numarul de biti transmisi si pot avea valori diferite. Este la latitudinea nivelului legatura de date sa detecteze si, daca este necesar, sa corecteze erorile.

2.b. Detectarea si corectarea erorilor

Exista 2 strategii de baza pentru tratarea erorilor. O modalitate este ca pe langa fiecare bloc de date trimis sa se includa suficienta informatie redundanta astfel incat receptorul sa poata deduce care a fost caracterul trimis. O alta solutie este sa se includa suficienta redundanta pentru a permite receptorului sa constate ca a aparut o eroare, dar nu care este eroarea, si sa ceara o retransmisie. Prima strategie utilizeaza **coduri corectoare de erori**, iar cea de-a doua utilizeaza **coduri detectoare de erori**.

# Coduri corectoare de erori

In cazul canalelor de comunicatie fara fir, este indicat sa adaugam destula informatie redundanta fiecarui bloc, in loc sa ne bazam pe retransmisie, care poate sa fie la randul sau afectata de erori.

In mod normal, un cadru contine *m* biti de date si  *r* biti redundanti sau de control. Sa consideram lungimea totala *n* (adica *m+r*). O unitate formata din n biti, care contine date si biti de control, este numita frecvent **cuvand de cod**  de n biti.

Date fiind doua cuvinte de cod, este posibil sa determinam cati biti corespunzatori difera. Pentru a determina cati biti difera, aplicam operatorul SAU EXCLUSIV intre cele doua cuvinte de cod si numaram bitii 1 din rezultat.

**Exemplu:**

10001001

10110001

00111000

Numarul de pozitii binare in care doua cuvinte de cod difera se numeste **distanta Hamming**. Daca doua cuvinte de cod sunt despartite de o distanta Hamming *d*, sunt necesare *d* erori de un singur bit pentru a-l converti pe unul in celalalt.

Proprietatile detectoare si corectoare de erori ale unui cod depinde de distanta sa Hamming. Pentru a **detecta** *d* erori, este nevoie de un cod cu distanta *d+1*. Atunci cand receptorul vede un cuvant de cod incorect, poate spune ca s-a produs o eroare de transmisie. Pentru a **corecta** *d* erori, este nevoie de un cod cu distanta *2d+1*.

## Metoda Hamming

Bitii cuvantului de cod sunt numerotati consecutiv, incepand cu bitul 1 de la marginea din stanga. Bitii care sunt puteri ale lui 2 (1,2,4,8,16,etc) sunt biti de control. Restul sunt completati cu cei m biti de date. Fiecare bit de control forteaza ca paritatea unui grup de biti, inclusiv el insusi, sa fie para (sau impara).

Un bit poate fi inclus in mai multe calcule de paritate. Pentru a vedea la care biti de control contribuie bitul de date din pozitia k, rescriem k ca o suma de puteri ale lui 2. De exemplu, 11=1+2+8 si 29=1+4+8+16. Un bit este verificat de acei biti de control care apar in dezvoltarea sa.

Cand soseste un cuvand de cod, receptorul initializeaza un contor la 0, Acesta examineaza apoi fiecare bit *k* (*k*=1,2,4,8,...) de control pentru a vedea daca are paritatea corecta. Daca nu, adauga *k* la contor. Daca, dupa ce au fost examinati toti bitii de control, contorul este 0, cuvantul de cod este acceptat ca valid. Daca valoarea este nenula, ea reprezinta numarul bitului incorect. De exemplu, daca bitii de control 1,2 si 8 sunt eronati, atunci bitul inversat este 11, deoarece este singurul verificat de bitii 1,2 si 8.

Codurile Hamming pot corecta numai erori singulare, insa exista un artificiu care poate fi utilizat pentru a permite codurilor Hamming sa corecteze erorile in rafala.

# Coduri detectoare de erori

Uneori este mai eficient sa utilizam un cod detector de erori si sa retransmitem blocul in care s-au detectat erori. De exemplu pe canalele cu siguranta mare, cum ar fi fibra optica.

## Codul polinomial (Cyclic redundancy code “**CRC**”)

Codurile polinomiale sunt bazate pe tratarea şirurilor de biţi ca reprezentări de polinoame cu coeficienţi 0 şi 1. Un cadru de k biţi este văzut ca o listă de coeficienţi pentru un polinom cu k termeni, de la xk-1 la x0 . Se spune că un astfel de polinom este de gradul k-1. Bitul cel mai semnificativ (cel mai din stânga) este coeficientul lui xk-1 ; următorul bit este coeficientul lui xk-2 ş.a.m.d. De exemplu, 110001 are şase biţi şi ei reprezintă un polinom cu şase termeni cu coeficienţii 1, 1, 0, 0, 0 şi 1: x 5 +x 4 +x 0 . Aritmetica polinomială este de tip modulo 2, în conformitate cu regulile teoriei algebrice. Nu există transport la adunare şi nici împrumut la scădere. Atât adunările cât şi scăderile sunt identice cu SAU EXCLUSIV.

Atunci când este utilizată metoda codului polinomial, emiţătorul şi receptorul se pun de acord în avans asupra unui polinom generator G(x). Atât bitul cel mai semnificativ cât şi cel mai puţin semnificativ trebuie să fie 1. Pentru a calcula suma de control pentru un cadru cu m biţi, corespunzător polinomului M(x), cadrul trebuie să fie mai lung decât polinomul generator. Ideea este de a adăuga o sumă de control la sfârşitul cadrului, astfel încât polinomul reprezentat de cadrul cu sumă de control să fie divizibil prin G(x). Când receptorul preia cadrul cu suma de control, încearcă să-l împartă la G(x). Dacă se obţine un rest, înseamnă că a avut loc o eroare de transmisie.

2.c. Protocoale elementare pentru legatura de date

Un cadru de date este compus din patru câmpuri: kind, seq, ack şi info, dintre care primele trei conţin informaţii de control, iar ultimul poate conţine datele efective care trebuie transferate. Ansamblul acestor câmpuri de control este numit antetul cadrului **(frame header)**.

Câmpul kind (tip) spune dacă există sau nu date în cadru, deoarece unele protocoale fac distincţie între cadrele care conţin exclusiv informaţii de control şi cele care conţin şi date. Câmpurile seq şi ack sunt utilizate pentru numere de secvenţă şi, respectiv, confirmări (acknowledgements); utilizarea lor va fi descrisă în detaliu mai târziu. Câmpul info al unui cadru de date conţine un singur pachet de date; câmpul info al unui cadru de control nu este utilizat.

# Un protocol simplex fără restricţii

Într-un protocol foarte simplu, datele sunt transmise într-o singură direcţie. Cele două niveluri reţea, de transmisie şi de recepţie, sunt considerate tot timpul pre-gătite. Timpul de prelucrare poate fi ignorat. Memoria de stocare disponibilă este infinită. Şi, cel mai bun lucru dintre toate, canalul de comunicaţie între niveluri legătură de date nu pierde şi nu alterează niciodată cadrele. Acest protocol poate fi considerat total nerealist.

Protocolul constă din două proceduri distincte, una de emisie şi cealaltă de recepţie. Emiţătorul lucrează la nivelul legătură de date al maşinii sursă, iar receptorul la nivelul legătură de date al maşinii de destinaţie.

Emiţătorul este într-un ciclu infinit care doar inserează datele pe linie cât poate de repede. Ciclul constă din trei acţiuni: preluarea unui pachet de date de la nivelul reţea (care este întotdeauna serviabil), construirea unui cadru de ieşire şi trimiterea cadrului pe drumul său. Acest protocol utilizează numai câmpul info al cadrului, deoarece celelalte câmpuri se referă la erori şi secvenţe de control, iar în acest caz nu există erori sau restricţii de control.

Receptorul este la fel de simplu. Iniţial el aşteaptă să se întâmple ceva, singura posibilitate fiind sosirea unui cadru nealterat. În cele din urmă, cadrul ajunge, iar procedura de tip wait\_for\_event se întoarce cu event setat la frame\_arrival (care este oricum ignorat). Apelul from\_physical\_layer mută cadrul nou sosit din zona tampon a echipamentului într-o anumită variabilă. În cele din urmă pachetul de date este trimis nivelului reţea şi nivelul legătură de date revine la starea de aşteptare a cadrului următor, autosuspendându-se pur şi simplu până la sosirea unui nou cadru.

# Un protocol simplu Stop-and-Wait (pas-cu-pas)

În acest caz se renunţă la cea mai nerealistă restricţie utilizată în protocolul anterior (protocolul 1): posibilitatea ca nivelul reţea receptor să prelucreze datele de intrare cu viteză infinită (sau echivalent, prezenţa în nivelul legăturii de date receptor a unui tampon infinit în care să fie memorate, cât timp îşi aşteaptă rândul, toate cadrele sosite). Totuşi, se presupune în continuare că nu se produc erori pe canalul de comunicaţie şi că traficul de date este încă simplex.

Principala problemă care trebuie rezolvată aici este cum să se evite ca emiţătorul să inunde receptorul cu date care sosesc mai rapid decât poate acesta să prelucreze. În esenţă, dacă receptorul are nevoie de un timp ∆t ca să execute from\_physical\_layer şi to\_network\_layer, atunci emiţătorul trebuie să transmită la o viteză medie mai mică de un cadru la fiecare interval de timp de ∆t. Mai mult, dacă se presupune că echipamentul receptor nu realizează automat memorarea în zona tampon şi gestiunea cozii de aşteptare, atunci emiţătorul nu trebuie să transmită niciodată un nou cadru până când cel vechi nu a fost preluat de rutina from\_physical\_layer, ca nu cumva cel nou să se scrie peste cel vechi.

O soluţie mult mai generală a acestei dileme este ca receptorul să furnizeze o reacţie către emiţător. După trimiterea unui pachet către nivelul său reţea, receptorul trimite un mic cadru fictiv către emiţător care, de fapt, îi dă emiţătorului permisiunea să transmită următorul cadru. După ce a transmis un cadru, emiţătorul este obligat de protocol să intre în aşteptare un timp, până când soseşte micul cadru fictiv (deci confirmarea). Utilizarea reacţiei de la receptor pentru a anunţa emiţătorul că poate trimite date este un exemplu de control al fluxului menţionat anterior.

2.d.1. PROTOCOALE CU FEREASTRĂ GLISANTĂ

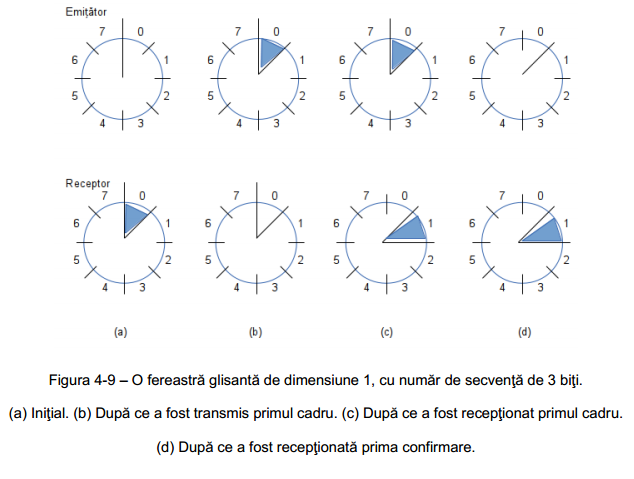
În protocoalele anterioare, cadrele cu date erau transmise într-o singură direcţie. În cele mai multe situaţii practice, este necesar să se transmită date în ambele direcţii. În acest model, cadrele cu date de la A la B sunt amestecate cu cadrele de confirmare de la A la B. Uitându-se la câmpul kind din antetul cadrului ce a sosit, receptorul poate spune dacă este vorba de un cadru de date sau de confirmare.

Cu toate că întrepătrunderea cadrelor de date şi control pe acelaşi circuit constituie o îmbunătă- ţire faţă de cazul utilizării a două circuite fizice separate, mai este posibilă încă o îmbunătăţire. Atunci când soseşte un cadru cu date, în locul emiterii imediate a unui cadru de control separat, receptorul stă şi aşteaptă până când nivelul reţea îi dă următorul pachet. Confirmarea este ataşată cadrului cu date de ieşire (utilizând câmpul ack din antetul cadrului). De fapt confirmarea este transportată pe gratis de către următorul cadru cu date de ieşire. Tehnica întârzierii confirmării, astfel încât să poată fi agăţată de următorul cadru de date, este cunoscută ca **ataşare (piggybacking)**.

Următoarele protocoale sunt protocoale bidirecţionale care aparţin unei clase de protocoale numite protocoale cu **fereastră glisantă (sliding window)**.

Esenţa protocoalelor cu fereastră glisantă este aceea că, la orice moment de timp, emiţătorul menţine o mulţime de numere de secvenţă care corespund cadrelor pe care are permisiunea să le trimită. Se spune că aceste cadre aparţin **ferestrei de transmisie (sending window)**. Similar, receptorul menţine de asemenea o **fereastră de recepţie (receiving window)**, ce corespunde mulţimii de cadre care pot fi acceptate. Fereastra emiţătorului şi fereastra receptorului nu trebuie să aibă aceleaşi limite minime şi maxime şi nici măcar aceeaşi dimensiune. În unele protocoale ele au dimensiune fixă, dar în altele ele pot creşte sau scădea pe măsură ce cadrele sunt emise sau recepţionate.

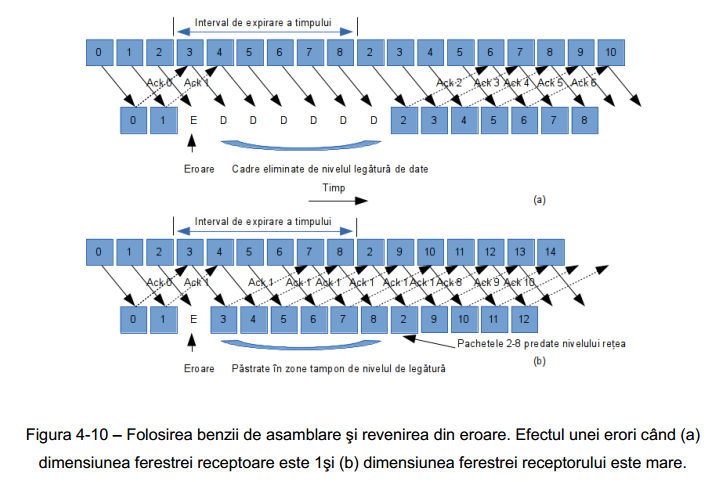
Numerele de secvenţă din cadrul ferestrei emiţătorului reprezintă cadre transmise sau cadre ce pot fi transmise, dar încă neconfirmate. De fiecare dată când de la nivelul reţea soseşte un nou pachet, acestuia îi este atribuit următorul număr de secvenţă, iar marginea superioară a ferestrei este avansată cu unu. Atunci când soseşte o confirmare, creşte cu unu limita inferioară a ferestrei. În acest mod, fereastra menţine continuu o listă de cadre neconfirmate. Un exemplu este prezentat în figura de mai jos.



2.d.2. Protocoale de revenire cu n paşi (Go Back n)

Utilizarea benzii de asamblare în cazul unui canal de comunicaţie nesigur ridică probleme serioase. Mai întâi să vedem ce se întâmplă dacă un cadru din mijlocul unui şir lung este modificat sau pierdut. Multe cadre succesive vor ajunge la receptor înainte ca emiţătorul să observe că ceva este greşit. Atunci când un cadru modificat ajunge la receptor este evident că el trebuie eliminat, dar ce trebuie să facă receptorul cu toate cadrele corecte care urmează? Nivelul legătură de date receptor este obligat să livreze pachete către nivelul reţea în secvenţă. În Figura 4-10, se prezintă efectele utilizării benzii de asamblare asupra revenirii în caz de eroare.

Există două moduri de bază de tratare a erorilor în prezenţa benzii de asamblare. Un mod, numit **revenire cu n paşi (go back n)**, este ca receptorul să elimine pur şi simplu cadrele care urmează, netrimiţând confirmări pentru cadrele eliminate. Această strategie corespunde unei ferestre de recepţie de dimensiune 1. Cu alte cuvinte, nivelul legătură de date refuză să accepte orice cadru exceptându-l pe următorul care trebuie livrat către nivelul reţea. Dacă fereastra emiţătorului se umple înaintea expirării contorului de timp, banda de asamblare va începe să se golească. În cele din urmă, timpul emiţătorului va expira şi se vor retransmite toate cadrele neconfirmate, în ordine, începând cu cadrul pierdut sau modificat. Dacă rata erorilor este mare, această abordare poate risipi o mare parte din lărgimea de bandă.



În Figura 4-10 (a) este prezentat protocolul de revenire cu n paşi pentru cazul în care fereastra receptorului are dimensiune unu. Cadrele 0 şi 1 sunt primite şi confirmate corect. Cadrul 2, totuşi, este alterat sau pierdut. Emiţătorul, care nu ştie de această problemă, continuă să trimită cadre până când timpul pentru cadrul 2 expiră. Apoi se întoarce la cadrul 2 şi o ia de la început cu el, trimiţând din nou cadrele 2, 3, 4 etc.