Capitolul 4. LEGATURA DE DATE

Acest nivel realizează o comunicare sigură şi eficientă între două noduri adiacente (conectate printr-un canal fizic de comunicaţie care asigură livrarea biţilor în aceeaşi ordine în care au fost transmişi).

4.1. Servicii şi funcţii

Serviciul principal este transferul datelor de la nivelul reţea al nodului sursă la nivelul reţea al nodului destinatar. Categoriile uzuale de servicii sînt următoarele:

- neconfirmate şi neorientate pe conexiune; utilizate în medii cu rate scăzute de erori, recuperarea fiind făcută de nivelele superioare; sînt caracteristice sistemelor de timp real, unde datele întîrziate sînt mai rele ca datele eronate;

- confirmate şi neorientate pe conexiune; fiecare cadru este recepţionat individual şi confirmat; dacă nu se primeşte confirmarea într-un anumit interval de timp se face retransmisia;

- orientate pe conexiune; garantează recepţia corectă a tuturor cadrelor, în succesiunea transmisă; transferul este caracterizat de trei faze distincte: stabilirea conexiunii, transmisia datelor şi desfiinţarea conexiunii.

Realizarea serviciilor menţionate se bazează pe împărţirea şirurilor de biţi în cadre, cărora le sînt ataşate informaţii de control al corectitudinii transmisiei. În cazul unor erori, corecţia se face prin retransmisia cadrului. Funcţiile nivelului legătură de date sînt prezentate succint în cele ce urmează.

*** Controlul cadrului.**

Se referă la delimitarea începutului şi a sfîrşitului cadrelor transmise. Un cadru reprezintă un mesaj (sau pachet) sau o parte a mesajului. Realizarea acestei funcţii se bazează pe utilizarea unor caractere de control (protocolul BSC - Bynary Synchronous Communication utilizat de IBM), a unei combinaţii formate din caractere de control (pentru începutul mesajului) şi dintr-un contor de caractere (DDCMP - Digital Data Communications Message Protocol) sau a unor delimitatori constînd din secvenţe fixe de biţi (HDLC - High Level Data Link Control).

SYN SYN SOH antet STX text ETX BCC

Figura 4.1.(a)

SYN SYN SOH contor flag răsp secv adresă CRC16 info CRC16

Figura 4.1.(b)

delimitator adresă comandă info FCS delimitator

Figura 4.1.(c)

În figura 4.1, care prezintă cele trei variante, se utilizează următoarele notaţii:
BCC - bloc de control ciclic
CRC16 - bloc de control ciclic pe 16 biţi
FCS - secvenţa de control a cadrului.

Celelalte caractere utilizate aparţin grupului de caractere de control ale codului ASCII şi au următoarele semnificaţii:
SOH - start of heading - începutul antetului mesajului;
STX - start of text - începutul textului mesajului;
ETX - end of text - sfîrşitul textului mesajului;
ETB - end of transmission block - sfîrşitul unui bloc (dacă mesajul este împărţit în mai multe
blocuri;
EOT - end of transmission - sfîrşitul transmisiei;
ENQ - enquiry - cerere de identificare a staţiei corespondente;
ACK - acknowledge - confirmare pozitivă a recepţiei unui mesaj;
NAK - not acknowledge - confirmare negativă a recepţiei unui mesaj;
SYN - synchronous idle - sincronizare a receptorului;
DLE - data link escape - prefix caractere de control.

***Transmisia transparentă.**

Asigură transferul datelor binare sau codificate nestandard, care includ coduri corespunzătoare delimitatorilor de cadre sau alte caractere de control. În cazul protocoalelor orientate pe caractere de control, se utilizează caracterul DLE, ca prefix al caracterelor de control. Începutul mesajului transparent este reprezentat prin secvenţa DLE STX, iar sfîrşitul sau prin DLE ETX. La transmisia unui octet de date avînd configuraţia caracterului DLE se inserează automat un al doilea caracter DLE. La recepţie, se ignoră toate caracterele de control cu excepţia lui DLE. Dacă se recepţionează un singur caracter DLE urmat de un caracter de control, combinaţia celor două este interpretată ca secvenţă de control. Dacă DLE este urmat de un al doilea caracter DLE, unul din ele este ignorat. Orice altă combinaţie reprezintă o eroare.

La protocoalele cu delimitatori de cadre se ţine cont de configuraţia delimitatorilor. Uzual acesta ocupă un octet şi are forma 01111110. Pentru a evita apariţia printre date a şase unităţi consecutive (şi deci interpretarea lor eronată ca delimitator de cadru), transmiţătorul inserează automat un 0 după o secvenţă de cinci unităţi, iar receptorul elimină automat un zero după cinci unităţi.

***Controlul erorilor.**

Se realizează prin următoarele mecanisme:
- adăugarea unei informaţii de control a cadrului, care permite receptorului să verifice corectitudinea acestuia;
- utilizarearea unor confirmări pentru cadrele recepţionate, transmitatorul fiind informat despre succesul sau insuccesul operaţiilor sale;
- utilizarea unor ceasuri care permit tratarea situaţiilor de pierdere a cadrelor sau confirmărilor;

- utilizarea unor numere de secvenţă utilizate în eliminarea duplicatelor şi în detectarea cadrelor pierdute, permiţînd totodată refacerea secvenţei corecte a cadrelor la recepţie.

***Controlul fluxului.**

Evită transmiterea cadrelor cu viteză mai mare decît cea cu care receptorul le poate accepta. Diversele variante de control se bazează pe acelaşi principiu: receptorul trebuie să garanteze, implicit sau explicit, permisiunea de a transmite.

***Gestiunea legăturii.**

Se referă la stabilirea şi desfiinţarea conexiunilor, iniţializarea numărătoarelor, readucerea conexiunilor într-o stare normală de funcţionare după producerea unor erori, administrarea legăturilor multipunct etc.

4.2. Protocoalele legăturii de date.

Prezentarea ce urmează să referă la faza de transfer al datelor. Există două categorii de protocoale pentru legătura de date: start-stop şi cu fereastră glisantă. La primele, transmiţătorul trimite un nou cadru numai după recepţia confirmării pozitive a cadrului precedent. La celelalte, nu aşteaptă confirmarea cadrelor precedente pentru a transmite cadre noi.

Configuraţia entităţilor protocoalelor discutate în continuare este prezentată în figura 4.5.

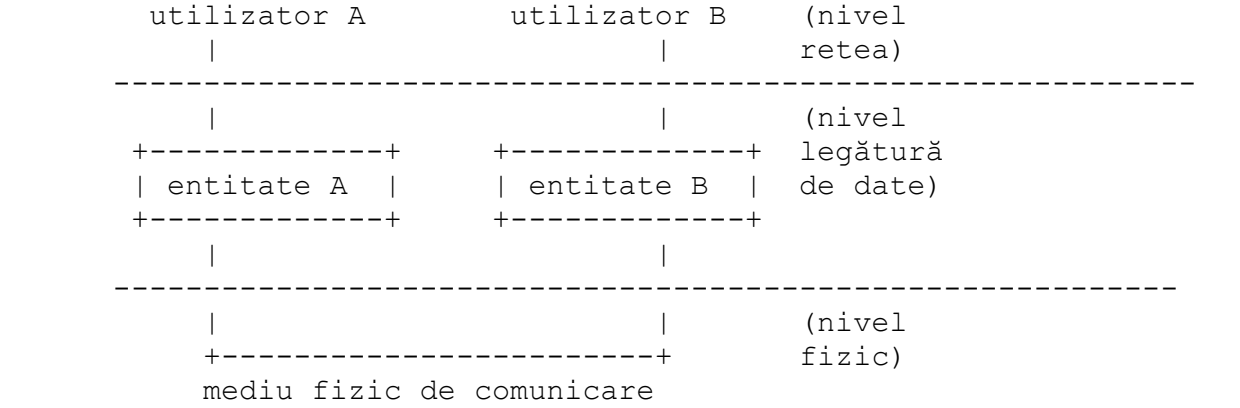


Figura 4.5.

Entităţile de protocol rezidă în sisteme diferite, avînd însă o legătură directă, printr-un mediu de comunicaţie.

Datele comunicate între utilizatori sînt grupate în **pachete**, pentru transmiterea lor între utilizatori şi entităţile legăturii de date fiind folosiţi descriptori de tipul următor (care specifică adresa datelor în memorie şi lungimea pachetului):

```
typedef struct {void far* adresa;
        word lungime;} pachet;
```

Pentru transmitere prin mediul fizic de comunicare, entitatea legătură de date încapsulează pachetul într-un cadru, prin adăugarea antetului şi a informaţiei de control. Cadrul constituie unitatea de date de protocol şi conţine cîmpuri care specifică felul cadrului, numere de secvenţă şi datele, conform următoarei descrieri:

```
enum FelCadru {data, ack, nak};
typedef unsigned char byte;
typedef unsigned int word;
typedef byte NrSecv;


typedef struct {
    FelCadru fel;
    NrSecv secv, conf;
    pachet info;
    } cadru;
```

Primitivele de serviciu la nivelul legăturii de date permit preluarea unui pachet de la reţea şi transmiterea unui pachet către reţea. În descrierea protocolului, ele se consideră implementate prin funcţiile de bibliotecă următoare:

```
- preluarea unui pachet de la reţea pentru transmitere pe canal
          pachet DeLaRetea();
- livrarea către reţea a unui pachet
          void LaRetea (pachet);
```

Similar, primitivele de serviciu la nivelul fizic permit transmiterea unui cadru şi recepţia unui cadru, funcţiile corespunzătoare fiind următoarele:

```
- trecerea unui cadru nivelului fizic pentru transmisie
          void LaFizic (cadru);
- preluarea unui cadru de la nivelul fizic
          cadru DeLaFizic();
```

Descrierea care urmează consideră că entităţile din nivelele reţea, legătură de date şi fizic sînt procese independente capabile să comunice prin mesaje. Sistemele în care ele evoluează sînt conduse prin evenimente, ceeace presupune posibilitatea ca o entitate să aştepte producerea unui eveniment, înainte de a executa anumite acţiuni. Evenimentul poate aparţine următoarelor categorii:

```
enum TipEven {SosireCadru,EroareControl,TimeOut,ReteaPregatita};
```

```
Functia:
          TipEven wait();
```

are ca efect punerea în aşteptare (blocarea) procesului apelant pînă la producerea unui eveniment semnificativ, moment în care întoarce o informaţie despre evenimentul produs.

4.2.1. Protocoale start-stop

***Protocol simplex fară restrictii.**

Cea mai simplă variantă de protocol are în vedere următoarele consideraţii:

- utilizatorul A vrea să transmită date lui B folosind o legătură sigură, simplex;
- A reprezintă o sursă inepuizabilă de date, astfel încît transmiţătorul nu trebuie să aştepte niciodată la preluarea datelor pentru transmisie;
- B reprezintă un consumator ideal, care preia imediat datele de la receptor ori de cîte ori acesta i le dă;
- canalul fizic de comunicaţie este fără erori.

Funcţionarea entităţilor transmiţătoare şi receptoare este descrisă de următoarele funcţii:

```
# define forever while(1)
// entitatea din sistemul transmitatorului
void transmit1()
   {
   cadru s;
   do
    {s.info=DeLaRetea();  //preia pachet
    LaFizic(s);      //transmite cadru
    }
   forever;
   }

// entitatea din sistemul receptorului
void recept1()
  {
   cadru r;
   TipEven even;
   do{
     even=wait();     //asteapta cadru
     r=DeLaFizic();    //primeste cadru
     LaRetea(r.info);   //preda pachet
    }
   forever;
  }
```

***Protocol simplex start-stop.**

În a doua variantă, considerăm în continuare canalul fără erori, dar utilizatorul B nu poate accepta date în orice ritm. Este necesar **controlul fluxului**, care în acest caz se realizează printr-o reacţie a receptorului către transmiţător, permiţîndu-i acestuia din urmă să transmită următorul cadru. Reacţia are forma unui cadru fictiv.

Configuraţia entităţilor utilizate în acest protocol este prezentată în figura 4.6.

```
     utilizator A       utilizator B
          |                  |
```

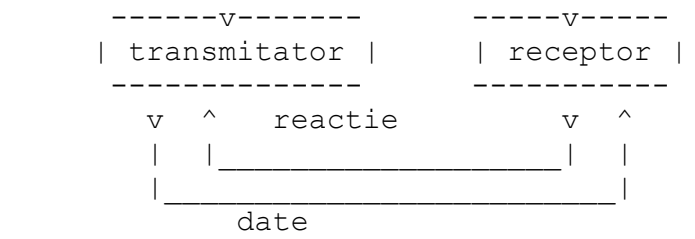


Figura 4.6.

Descrierea protocolului este următoarea:

```
void transmit2(){
   cadru s;
   TipEven even;
   do{
    s.info=DeLaRetea();
    LaFizic(s);
    even=wait();  //asteapta confirmarea
    }forever;
}


void recept2(){
   cadru s,r;
   TipEven even;
   do{
    even=wait();   //poate fi doar SosireCadru
    r=DeLaFizic();
    LaRetea(r.info);
    LaFizic(s);   //transmite confirmarea
    }forever;
}
```

***Protocol simplex pentru un canal cu erori.**

S-ar părea că în cazul unui canal cu erori o uşoară modificare a protocolului anterior este suficientă. Receptorul transmite un cadru de confirmare doar pentru cadre corecte. În caz de eroare, transmiţătorul nu primeşte confirmarea într-un timp prestabilit şi ca urmare retransmite cadrul. Funcţia de armare a ceasului şi de permisiune a evenimentului TimeOut este:

```
          void StartCeas(NrSecv);
```

iar funcţia de dezarmare a ceasului este:

```
          void StopCeas (NrSecv);
```

Parametrul lor permite asocierea unui eveniment TimeOut cu cadrul avînd un număr de secvenţă specificat.

Această variantă prezintă pericolul duplicării cadrelor, în cazul în care se pierde confirmarea mesajului şi nu mesajul

(transmiţătorul retransmite cadrul precedent, pe care receptorul îl ia drept cadru nou).

Evitarea acestei erori se face prin includerea unui număr de secvenţă în antetul cadrului. Deoarece în acest caz, numărul de secvenţă trebuie să diferenţieze doar două cadre succesive, este suficient un contor de un bit, numerele de secvenţa 0 şi 1 ale cadrelor succesive alternînd.

Pentru justificare, fie cadrele succesive m, m+1, m+2 cu numerele de secvenţă respectiv 0, 1 şi 0. După transmisia cadrului m (0), poate urma m (0), sau m+1 (1), după cum s-a primit sau nu confirmarea. În nici un caz nu urmează m+2 (0), care presupune confirmarea corectă a lui m (0). Deci dacă receptorul a primit corect m (0) şi primeşte un nou cadru cu număr de secvenţă 0, el nu poate fi decît m (0) şi în consecinţă îl ignoră.

În descrierea care urmează folosim o funcţie care incrementează circular un număr de secvenţă:

```
          void inc (NrSecv&);
```

valoarea maximă a numerelor de secvenţă fiind, în acest caz:

```
#define MaxSecv 1
```

Protocolul are următoarea descriere:

```
void inc(NrSecv& k){k==MaxSecv ? k=0 : k++;}

void transmit3(){
   NrSecv CadruUrmator=0;
   cadru s;
   TipEven even;

   s.info=DeLaRetea();
   do{
    s.secv=CadruUrmator;
    LaFizic(s);
    StartCeas(s.secv);
    even=wait();       //poate fi SosireCadru,
                 //     TimeOut sau
                 //     Eroarecontrol
    if(even==SosireCadru){  //confirmare intacta
     StopCeas(s.secv);
     s.info=DeLaRetea();
     inc(CadruUrmator);
    }
   }forever;
}

void recept3(){
   NrSecv CadruAsteptat=0;
   cadru r,s;
   TipEven even;

   do{
    even=wait();    //poate fi SosireCadru sau EroareControl
```

```
    if(even==SosireCadru){
     r=DeLaFizic();
     if(r.secv==CadruAsteptat){
      LaRetea(r.info);   //cind cadrul este în secventa
      inc(CadruAsteptat);
     }
     LaFizic(s);      //transmite oricum confirmarea
    }
   }forever;
}
```

4.2.2. Protocoale cu fereastră glisantă

Protocoalele anterioare sînt simplex. Pentru o transmisie duplex ar fi necesare două legături distincte, una pentru fiecare sens. Ca alternativă, putem utiliza cadrele de confirmare pentru transmiterea datelor în sens opus. Dacă nu sînt date, se transmite doar confirmarea. Apar deci mai multe feluri de cadre, diferențiabile printr-un cîmp din antet (fel).

Aceste elemente sînt cuprinse în protocoalele cu fereastră glisantă. Esența lor este prezentată în continuare.

Transmițătorul menține o listă cu numerele de secvență ale cadrelor transmise dar neconfirmate, alcătuind fereastra transmițătorului. Cînd trebuie transmis un cadru, i se ataşează numărul de secvență următor (fereastra se măreşte). Cînd se recepționează o confirmare, marginea inferioară a ferestrei se deplasează cu o unitate. Cadrele se păstrează de transmițător pînă la confirmarea lor.

Receptorul păstrează o listă cu numerele de secvență ale cadrelor pe care le poate accepta (fereastra receptorului). Cadrele din fereastră sînt acceptate, celelalte sînt ignorate. Cînd soseşte un cadru cu număr de secvență egal cu marginea inferioară, fereastra este deplasată cu o poziție (prin modificarea ambelor margini), păstrîndu-se dimensiunea constantă.

*** Protocol cu fereastră de dimensiune unu.**

Configurația entităților acestui protocol este cea din figura 4.7.

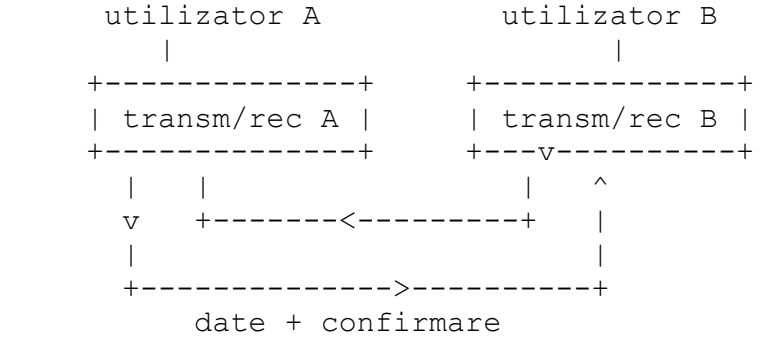


Figura 4.7.

Ambele ferestre fiind unitare, transmiţătorul trebuie să aştepte confirmarea înainte de a transmite un nou cadru (la fel ca în cazul precedent). Deoarece algoritmii transmiţătorului şi receptorului sînt aproape identici, dăm o singură descriere pentru ambele staţii. Fiecare staţie realizează ciclic următoarele operaţii:

- recepţia unui cadru,
- prelucrarea şirului de cadre recepţionate,
- prelucrarea şirului de cadre transmise,
- transmiterea sau retransmiterea unui cadru impreună cu confirmarea cadrului recepţionat corect.

```
void protocol4(){
   NrSecv CadruUrmator=0;
   NrSecv CadruAsteptat=0;
   cadru r,s;
   TipEven even;      //SosireCadru, TimeOut sau
                      //EroareControl
   s.info=DeLaRetea();
   s.secv=CadruUrmator;
   s.conf=1-CadruAsteptat;
   LaFizic(s);
   StartCeas(s.secv);
   do{
    even=wait();
    if(even==SosireCadru){
     r=DeLaFizic();
     if(r.secv==CadruAsteptat){ //prel. şir cadre receptionate
      LaRetea(r.info);
      inc(CadruAsteptat);
     }
     if(r.conf==CadruUrmator){  //prel. sir cadre transmisie
      StopCeas(r.conf);
      s.info=DeLaRetea();
      inc(CadruUrmator);
     }
    }
    s.secv=CadruUrmator;
    s.conf=1-CadruAsteptat;
    LaFizic(s);
    StartCeas(s.secv);
   }forever;
}
```

O situaţie necorespunzătoare apare dacă ambele staţii transmit un cadru iniţial. Figura 4.8. prezintă comparativ o secvenţă normală de pachete şi una anormală, datorată transmiterii simultane a cadrelor iniţiale. Notaţia utilizată pentru conţinutul cadrelor este (secv, conf, info).

Pentru a corecta acest neajuns, o staţie trebuie să o preceadă pe cealaltă prin acţiunile sale. Pentru aceasta, doar una trebuie să execute transmisia unui cadru în afara buclei principale.

Cîmpul conf conţine numărul ultimului cadru recepţionat corect. Dacă el corespunde cadrului aflat în transfer, transmiţătorul preia alt pachet pentru transmisie. Altfel, se transmite acelaşi pachet.

```
statia A       statia B       statia A          statia B
------------------------      ------------------------
 (0, 1, A0) -> acceptat       (0, 1, A0) -> <- (0, 1, B0)
 acceptat  <- (0, 0, B0)      aceptat           aceptat
 (1, 0, A1) -> acceptat       (0, 0, A0) -> <- (0, 0, B0)
 acceptat  <- (1, 1, B1)      rejectat          rejectat
 (0, 1, A2) -> acceptat       (1, 0, A1) -> <- (1, 0, B1)
 ...                          acceptat         acceptat
                              (1, 1, A1) -> <- (1, 1, B1)
                              rejectat          rejectat
```

Figura 4.8 (a).

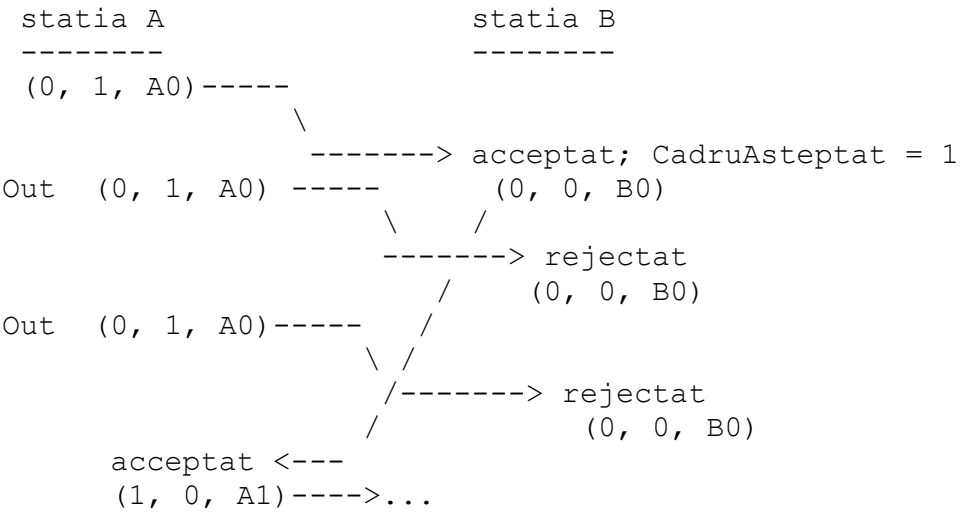


Figura 4.8 (b).

Cîmpul secv conţine numărul de secvenţă al cadrului. Dacă el coincide cu cel aşteptat, se transmite un pachet nivelului reţea.

Staţiile nu se blochează şi nu acceptă duplicate. Să presupunem de exemplu că A încearcă să trimită cadrul 0 lui B, iar B încearcă să trimită cadrul 0 lui A. Presupunem că timpul de aşteptare (time-out) al lui A este prea scurt, deci A trimite repetat cadre (0, 1, A0). Cînd primul cadru valid ajunge la B, va fi acceptat, punîndu-se CadruAsteptat pe 1. Celelalte cadre sînt ignorate, avînd secv=0. Mai mult, duplicatele au conf = 1, în timp ce B aşteaptă conf = 0 pentru a scoate un nou pachet de la reţea. Ca urmare, B va continua să transmită repetat un cadru cu (0, 0, ...). Unul din cadre ajunge la A, care începe să transmită un nou pachet (figura 4.8.b).

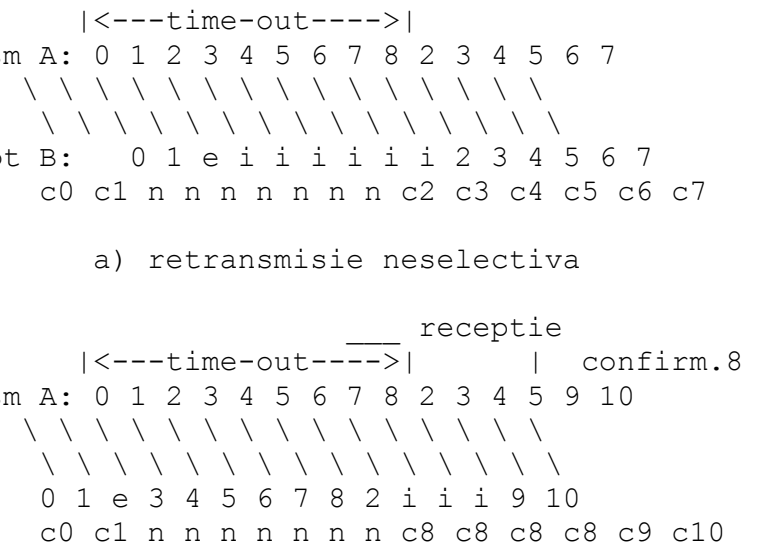
*** Protocoale cu fereastră supraunitară de transmisie.**

În cazul unor canale cu întîrzieri mari de transmisie, aşteptarea confirmării fiecărui cadru înainte de transmiterea următorului

poate conduce la o eficienţă scăzută. Astfel, la un canal cu o întîrziere de propagare dus-întors de 500 msec şi o viteza de 50 Kbps, transmiterea unui cadru de 1000 de biţi durează efectiv doar 20 de milisecunde din cele 520 necesare ciclului complet de transmitere şi recepţie a confirmării.

Performanţa se poate ameliora permiţînd transmiterea unui cadru înainte de a se primi confirmarea cadrelor anterioare. În exemplul precedent, dacă transmiţătorul trimite cadrele unul după altul, el va recepţiona confirmarea primului cadru după transmiterea celui de al 26-lea.

Erorile nu pot fi sesizate şi corectate imediat de transmiţător. De obicei, el află despre producerea unei erori după ce a transmis şi alte cadre. Receptorul, care trebuie să păstreze secvenţialitatea cadrelor pe care le livrează utilizatorului, are două posibilităţi de tratare a cadrelor primite după eroare. O cale este de ignorare a acestora, fără transmitere de confirmări. Ea corespunde unei ferestre de recepţie unitare. În acest caz, transmiţătorul va fi obligat să retransmită toate cadrele începînd cu cel eronat. A doua cale este memorarea cadrelor recepţionate corect, urmînd a fi retransmise selectiv doar cadrele eronate. Ea corespunde unei ferestre a receptorului mai mare ca unu. Cele două variante sînt schematizate în figura 4.9.



a) retransmisie neselectiva

b) retransmisie selectivă

Figura 4.9.
(c0-confirmă cadru 0; e-cadru eronat; i-cadru ignorat; n-cadru neconfirmat)

Prima variantă este mai puţin eficientă pentru canalul de transmisie. A doua necesită spaţiu de memorare la receptor a cadrelor sosite după producerea erorii.

*** Protocol cu retransmitere neselectivă.**

Fereastra maximă a transmiţătorului poate fi de MaxSecv cadre, deşi există MaxSecv+1 numere de secvenţă distincte. Justificăm această restricţie prin următorul scenariu, în care presupunem că MaxSecv=7:

1. Transmiţătorul trimite cadrele 0..7;
2. Toate cadrele sînt recepţionate şi confirmate;
3. Toate confirmările sînt pierdute;
4. Transmiţătorul retrimite la time-out toate cadrele;
5. Receptorul acceptă duplicatele.

În acest protocol renunţăm la presupunerea că utilizatorul este o sursă ideala de pachete şi includem mecanismul de reglare a fluxului de date: cînd utilizatorul are un pachet de transmis el provoacă un eveniment "ReteaPregatita"; cînd o staţie are MaxSecv cadre neconfirmate, ea interzice reţelei să-i transmită alte pachete prin procedura "DezactivRetea"; cînd numărul de pachete scade, ea anulează restricţia prin funcţia "ActivRetea".

Transmiţătorul păstrează cadrele neconfirmate pentru o eventuală retransmisie. Confirmarea cadrului n provoacă automat confirmarea cadrelor n-1, n-2,... anterioare, ceeace compensează eventualele pierderi ale confirmărilor acestor cadre.

Deoarece mai multe cadre îşi asteaptă confirmarea, se utilizează ceasuri separate pentru diferite mesaje. Fiecare ceas este pornit la transmiterea unui cadru şi este oprit la recepţia confirmării sale. La producerea unui eveniment TimeOut, toate cadrele aflate în memoria tampon a transmiţătorului sînt retrimise.

```
#define MaxSecv 7
void ActivRetea();
void DezactivRetea();

NrSecv CadruUrmator,   //urmatorul cadru de transmis
    CadruAsteptat,  //urmatorul cadru asteptat
    ConfAsteptata;  //cel mai vechi cadru neconfirmat
cadru r,s;
pachet tampon[MaxSecv+1];
NrSecv ntampon,i;
TipEven even;

short intre(NrSecv a, NrSecv b, NrSecv c){
  //intoarce 1 daca a<=b<c circular
  return a<=b && b<c || c<a && a<=b || b<c && c<a;
}

void transmite(NrSecv nrcadru){
   //construieste şi transmite un cadru de date
   s.info=tampon[nrcadru];
   s.secv=nrcadru;
   s.conf=(CadruAsteptat+MaxSecv)%(MaxSecv+1);
   LaFizic(s);
   StartCeas(nrcadru);
}
```

```
void protocol5(){
   ActivRetea();
   CadruUrmator=0;
   CadruAsteptat=0;
   ConfAsteptata=0;
   ntampon=0;
   do{
    even=wait();
    switch(even){
     case ReteaPregatita: tampon[CadruUrmator]=DeLaRetea();
                ntampon++;
                transmite(CadruUrmator);
                inc(CadruUrmator);
                break;
     case SosireCadru:  r=DeLaFizic();
                if(r.secv==CadruAsteptat){
                 LaRetea(r.info);
                 inc(CadruAsteptat);
                }
                while(intre(ConfAsteptata,r.conf,
                      CadruUrmator)){
                 ntampon--;
                 StopCeas(ConfAsteptata);
                 inc(ConfAsteptata);
                }
                break;
     case EroareControl: break;
     case TimeOut:    CadruUrmator=ConfAsteptata;
                for(i=1;i<=ntampon;i++){
                 transmite(CadruUrmator);
                 inc(CadruUrmator);
                }
     }
     if(ntampon<MaxSecv)
          ActivRetea();
     else   DezactivRetea();
   }forever;
}
```

*** Protocol cu retransmitere selectiva.**

În acest caz, fereastra receptorului este mai mare ca unu. Cînd se primeşte un cadru avînd număr de secvenţă în fereastră, el este acceptat şi memorat de nivelul legătură de date, nefiind comunicat utilizatorului decît după livrarea cadrelor dinaintea sa.

Fereastra receptorului nu poate fi egală cu cea a transmiţătorului aşa cum rezultă din următorul scenariu, pentru MaxSecv = 7 :

1. Transmiţătorul trimite cadrele 0..6
2. Cadrele sînt recepţionate şi fereastra receptorului devine 7, 0, 1, 2, 3, 4, 5
3. Toate confirmările sînt pierdute
4. Transmiţătorul retrimite cadrul 0 la time-out
5. Receptorul acceptă cadrul 0 aflat în fereastra şi cere cadrul 7

6. Transmiţătorul află că toate cadrele sale au fost transmise corect şi trimite
cadrele 7, 0, 1, 2, 3, 4, 5
7. Receptorul acceptă cadrele, cu excepţia lui 0, pentru care are deja un cadru recepţionat. Ca urmare, ignoră acest cadru, luînd în locul lui dulpicatul cadrului 0 anterior.

Eroarea se produce deoarece ferestrele succesive ale receptorului au numere de secvenţă comune, neputîndu-se face diferenţa între cadrele noi şi duplicatele celor vechi. Pentru a elimina aceste suprapuneri, fereastra receptorului trebuie să fie cel mult jumătate din gama numerelor de secvenţă.

Fereastra receptorului determină şi numărul de tampoane de recepţie, ca de altfel şi numărul de ceasuri necesare.

Cînd canalul este puternic încărcat, confirmările cadrelor transmise de la staţia A la staţia B pot fi comunicate odată cu datele vehiculate de la B la A. Dacă traficul de la B la A este scăzut, se recurge la transmiterea unor cadre speciale de confirmare. Transmiterea confirmărilor nu se face imediat după recepţia unui cadru corect, ci după un interval de timp convenabil stabilit şi numai dacă în acest interval nu a apărut posibilitatea transmiterii confirmării într-un cadru de date de la B la A. Terminarea intervalului de timp are drept corespondent evenimentul "ReteaLibera".

Pentru a îmbunătăţi eficienţa transmisiei, dacă receptorul detectează un cadru eronat, el poate transmite o confirmare negativă "nak", care reprezintă o cerere explicită de retransmitere a cadrului specificat în "nak". Receptorul trebuie să evite transmiterea mai multor cadre "nak" pentru acelaşi cadru aşteptat.

```
void protocol6(){

 initializari_contoare;
 do{
   even=wait();
   switch (even)
    case ReteaPregatita:
     accepta_salveaza_si_transmite_un_cadru;
     break;

    case SosireCadru:
     r=DeLaFizic();
     if (r.fel == data){
        transmite_nak_daca_r_diferit_de_cadru_asteptat;
        accepta_cadru_daca_in_fereastra_receptie;
        livreaza_pachetele_sosite;
        actualizeaza_fereastra_receptie;
      }
     if (r.kind == nak) retransmite_cadru_cerut;
     trateaza_confirmare_cadre_eliberind_buffere;
     break;
```

```
    case EroareControl: transmite_nak;break;

    case TimeOut: retransmite_cadrul_corespunzator;break;

    case ReteaLibera: transmite_confirmare_ack; break;
   }
   activeaza_sau_dezactiveaza_nivel_retea;
 }forever;
}
```

4.3. Protocolul HDLC procedura LAPB

HDLC reprezintă o mulţime de protocoale, care diferă între ele prin tipurile staţiilor (entităţilor legătură de date), configuraţiile legăturii şi modurile de operare.

Sînt astfel acceptate următoarele tipuri de staţii:
- primare care controlează legătura, cadrele generate de ele fiind denumite comenzi;
- secundare care lucrează sub controlul staţiilor primare, cadrele generate de ele fiind denumite răspunsuri;
- combinate, care pot genera atît comenzi cît şi răspunsuri.

Legătura poate fi balansată sau nebalansată. O configuraţie nebalansată constă dintr-o staţie primară şi una sau mai multe staţii secundare (legătură punct la punct, respectiv legătură multi-punct). O configuraţie balansată cuprinde două staţii combinate. În ambele situaţii, transmisia poate fi duplex sau semiduplex.

În ceeace priveşte modurile de transfer, ele pot fi:
- modul de răspuns normal (NRM - Normal Response Mode), pe o legătura nebalansată, în care staţia secundară poate transmite date doar ca urmare a unei invitaţii primite de la staţia primară;
- modul balansat asincron (ABM - Asynchronous Balanced Mode), pe o legătură balansată, în care fiecare staţie poate transmite fară a cere permisiunea celeilalte;
- modul de răspuns asincron (ARM - Asynchronous Response Mode), pe o legătură nebalansată, în care o staţie secundară poate transmite un răspuns, fară a aştepta o comandă de la staţia primară.

Procedura LAPB (Link Access Protocol Balanced) corespunde unei legături balansate cu staţii combinate. Formatul cadrului HDLC a fost stabilit în ideea folosirii eficiente a legăturii şi a independenţei totale a codului folosit de utilizator pentru date. Din acest motiv, cîmpurile de control au poziţii, dimensiuni şi structuri predefinite.

Formatul general al cadrului este următorul:

delimitator adresă control informaţie SCC delimitator

unde **delimitator** este o secvenţă fixă de biti, 01111110;

adresa este utilizată pe liniile multipunct pentru a identifica una din staţiile secundare;
pentru uniformitate, se foloseşte şi pe staţiile punct la punct;

control utilizat pentru numere de secvenţă, confirmări; are o configuraţie care diferă de la
un tip de cadru la altul;

SCC secvenţa de control a cadrului este o variantă a codului de control ciclic CRC CCITT.

Există trei tipuri de cadre: de informaţie, de supervizare a transmisiei şi de gestiune a legăturii (nenumerotate). Cîmpurile de control corespunzatoare sînt prezentate în figura 4.10.

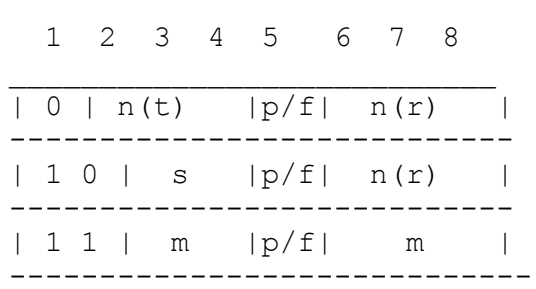


Figura 4.10.

Notaţiile au următoarele semnificaţii:
n(t) număr de secvenţă al cadrului transmis,
n(r) număr de secvenţă al cadrului confirmat, ambele mod 8 (restrîns) sau mod 128 (extins)
s biţi supervizor, codifică diferite funcţii de control
m biţi modificatori, codifică funcţiile de gestiune
p/f poll/final, are semnificaţia de invitaţie la emisie (în cadru de comandă), respectiv de sfîrşit de
transmisie (în cadru de răspuns).

*** Comenzi şi răspunsuri LAPB**

Comenzile au adresa staţiei îndepărtate, iar răspunsurile au adresa staţiei locale. Pentru LAPB, acestea sînt prezentate în tabelul 4.1 şi descrise succint în cele ce urmează.

Tabel 4.1.

	Comenzi	Răspunsuri
cadre informatie	I = information	suprimat)
cadre supervizor	RR = receive ready	RR
	RNR = receive not ready	RNR
	REJ = reject	REJ
cadre nenumerotate	SABM = set asynchronous balanced mode	UA = unnumbered acknowledge
	DISC = disconnect	DM = disconnected mode
		FRMR = frame reject

RR confirmare pozitivă; n(r) indică următorul cadru aşteptat;

REJ confirmare negativă, datorată detecţiei unei erori; se indică numărul cadrului eronat, care trebuie deci retransmis; retransmiterea se face neselectiv;

RNR confirmă toate cadrele cu numere de secvenţă pînă la n(r) exclusiv, dar determină transmiţătorul să suspende temporar transmisia; reluarea se face cu RR, REJ sau alt cadru de control;

SABM iniţializează o legătură de date pentru două staţii cu drepturi egale (legătură balansată; în legăturile nebalansate, definite de alte proceduri HDLC, o staţie este master, cealaltă este slave);

DISC desfiinţează o legătură;

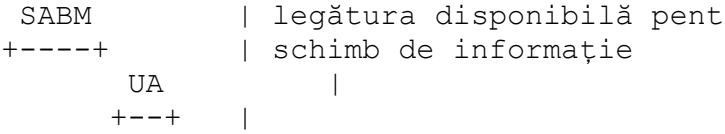
FRMR indică recepţia unui cadru cu suma de control corectă dar care nu respectă protocolul;

UA confirmă cadrele de gestiune a legăturii; deoarece un singur cadru de gestiune poate fi în transfer la un moment dat, nu este necesar un număr de secvenţă.

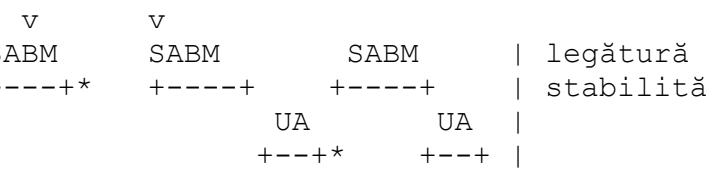
LAPB are următoarele caracteristici:
- semnalare rapidă a erorilor prin REJ;
- absenţa cadrelor I de răspuns;
- posibilitatea numerotării cadrelor modulo 128.

Prezentăm în continuare cîteva exemple de funcţionare a unei legături balansate. Se folosesc următoarele notaţii:
pentru cadre de informaţie I n, m P/F
pentru cadre supervizor S m P/F
pentru cadre nenumerotate N P/F
unde **n** este numărul de secvenţă la transmisie
m este numărul de secvenţă la recepţie
P/F dacă apar, sînt considerate active.
Cadrele afectate de linia de transmisie sînt marcate prin *.

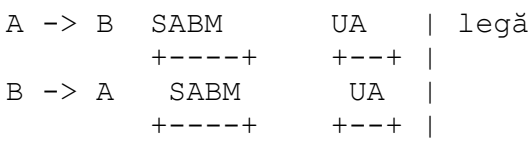
1) Stabilirea legăturii fară erori



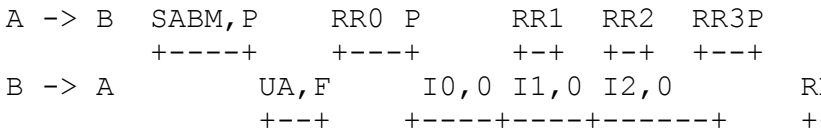
2) Producerea unor erori la stabilirea legăturii



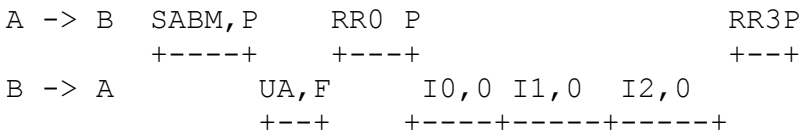
3) Conflict între staţii la stabilirea legăturii



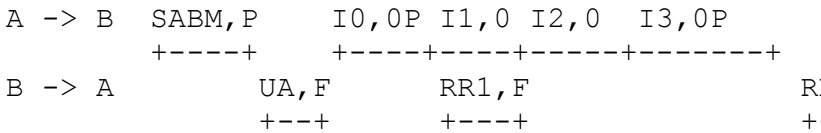
4) Stabilirea legăturii şi transfer de informaţii de la B, cu confirmarea fiecărui cadru



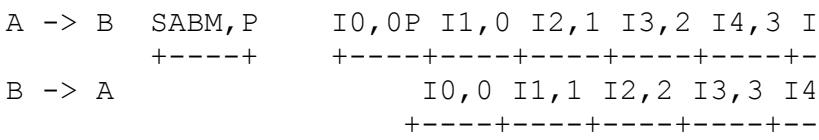
5) Confirmarea simultană a mai multor cadre



6) Stabilirea legăturii şi transferul informaţiei de la A la B, cu confirmarea mai multor cadre



7) Stabilirea legăturii şi transferul simultan între staţii



8) Deconectarea legăturii