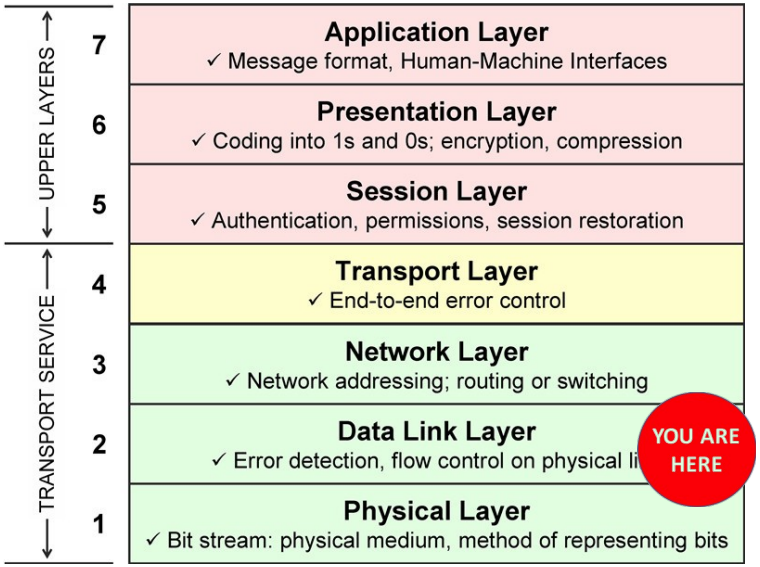
Nivelul legăturii de date

Cuprins

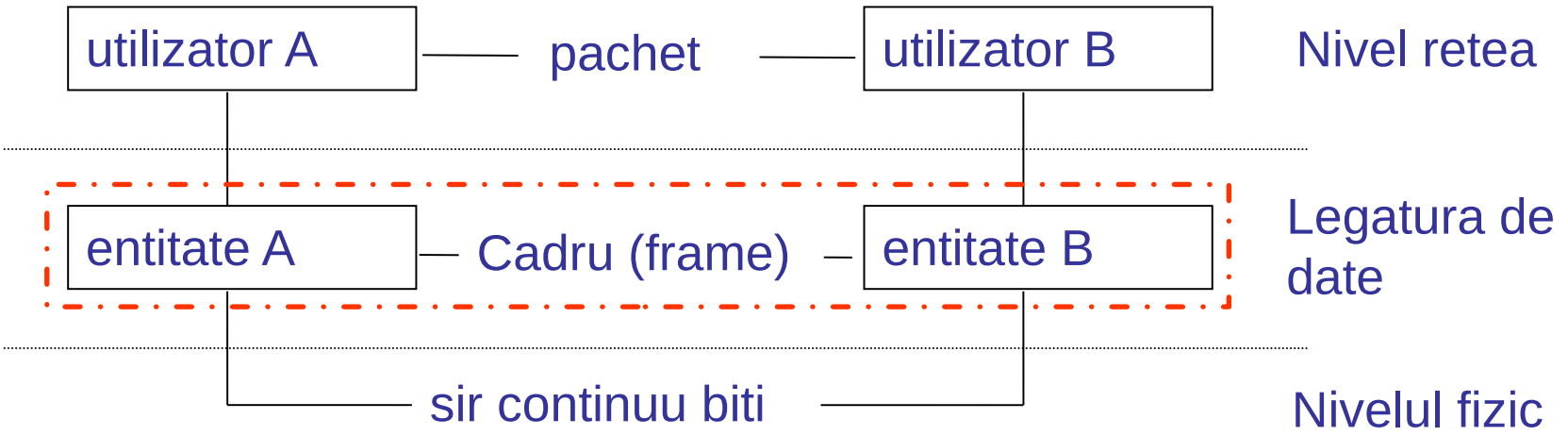
- Functiile legaturii de date
- Incadrarea datelor
- Transmisia transparenta
- Detectia erorilor de transmisie
- Protocoale start-stop
- Exemple de protocoale:
- Ethernet, PPP, PPPoE



Foarte importante!

- implemantarea protocoalelor (programarea)
- orientarea pe evenimente
- tratarea erorilor (nu exista in programarea “obisnuita”)
- mecanismul ferestrelor glisante
- performanta (retransmitre neselectiva / selectiva)

Funcţiile nivelului legăturii de date



1. Încadrarea datelor
   - nivelul fizic transmite un sir continuu de biti
   - legatura de date partitioneaza sirul in cadre
   - delimiteaza cadrele
2. Transmisia transparentă
   - permite trimiterea in cadru a oricarei combinatii de biti
     - ex. care coincid cu delimitatorii cadrului

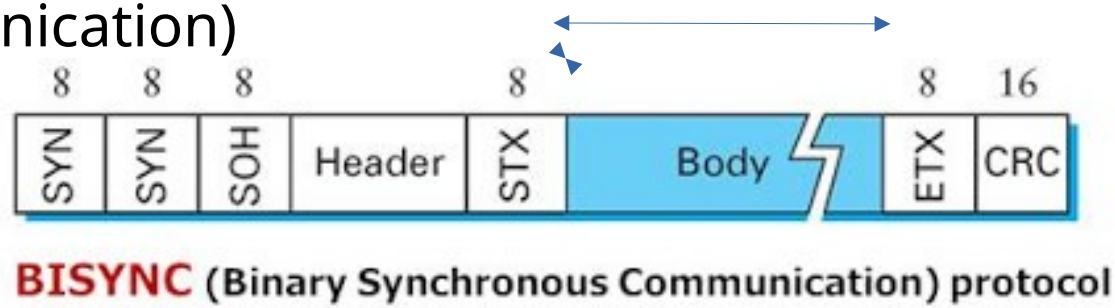
Funcţiile nivelului legăturii de date

3. Controlul erorilor
   - Coduri detectoare si corectoare de erori
   - Mecanisme de protocol
     - mesaje de confirmare
     - ceasuri
     - numere de secvenţă
4. Controlul fluxului – protocol
   - mesaje de permisiune pentru transmiţător
5. Gestiunea legăturii – protocol
   - stabilirea şi desfiinţarea legăturii
   - re-iniţializare după erori
   - configurarea legăturii (ex. legături multipunct)

- Moduri de transmisie:
  - **Full duplex** = Comunicația poate avea loc în ambele sensuri simultan
  - **Half duplex** = Comunicația poate avea loc în ambele sensuri, dar nu simultan
  - **Simplex** = O entitate poate doar transmite informații și cealaltă poate doar să primească

Metode de încadrare a datelor (Exemple)

1) Caractere de control (BISYNC - Binary Synchronous Communication)



BISYNC (Binary Synchronous Communication) protocol

SOH - start of heading
ETX - end of text
EOT - end of transmission
ACK - acknowledge
SYN - synchronous idle
CRC - cyclic redundancy check
STX - start of text
ETB - end of transmission block
ENQ - enquiry
NAK - not acknowledge
DLE - data link escape

- Începutul unui cadru este delimitat prin caracterul special **SYN.**
- Segmentul de date este încadrat de caracterele speciale **STX** și **ETX**.
- Începutul header-ului este marcat prin caracterul special **SOH.**
- **CRC** – folosit pentru detecția erorilor

Transmisia transparentă

Probleme cu abordarea bazată pe caractere de control:

- Datele trimise pot conține caractere de control în interior

În cazul transmisiei de conținut binar:

STX text… ETX …text ETX → ETX fals ?!

Soluție:

- umplere cu caractere *(**Byte/Character stuffing**)* - escaparea caracterelor de control, folosind caracterul special DLE (Data Link Escape)

Umplere cu caractere

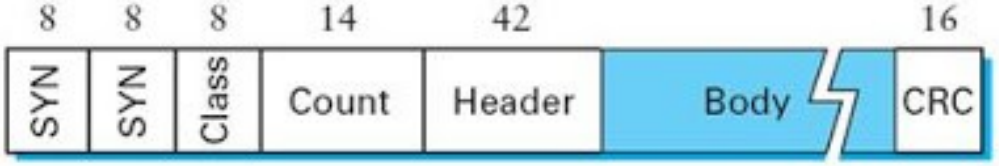
- Toate caracterele de control din corpul cadrului sunt escapate cu **DLE (data link escape)** înainte de transmisie
- Caracterul DLE este escapat și el (dublat) dacă apare în corpul cadrului.

- Sunt definite transformări admise în datele trimise
  - STX – > DLE STX
  - ETX – > DLE ETX
  - DLE – > DLE DLE

- Se consideră că a avut loc o eroare la receptie dacă se primește caracterul DLE urmat de altceva decât STX, ETX, DLE

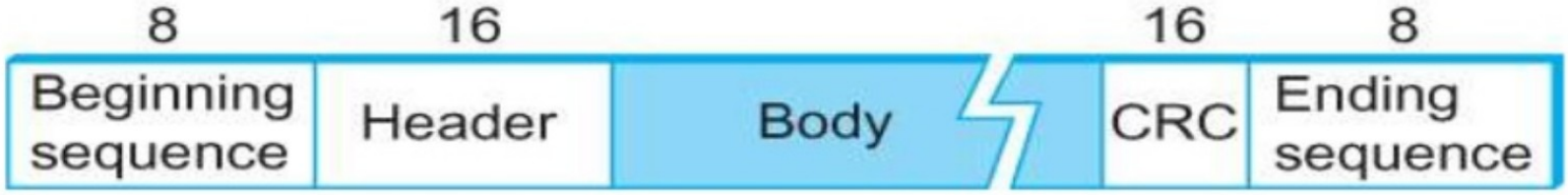
Metode de încadrare (Exemple)

2) Numărarea caracterelor (DDCMP - Digital Data Communications Message Protocol)



DDCMP (Digital Data Communications Message Protocol)

3) Indicatori de încadrare (HDLC – High Level Data Link Control)



Umplere cu biţi (Bit stuffing)

În cazul protocolului HDLC secvențele de început și de sfârșit al cadrului sunt 01111110.

Datele de transmis pot include un flag fals de terminare a cadrului

Date de trimis: 01101111110111111111010

Solutia: se adauga un zero dupa fiecare 5 unitati consecutive

01111110 01101111101011111011111010 01111110

Bitul 0 inserat se elimină la recepție

- Adaugarea se face indiferent daca dupa 5 unitati urmeaza 0 sau 1
- Simplifica regula receptorului: elimina zerouri aflate dupa 5 unitati

Date primite: 01101111101011111011111010

După eliminarea biților inserați: 01101111110111111111010

Nivelul legăturii de date

Detecţia și corectarea erorilor

Detecţia și corectarea erorilor

Descrierea problemei:

- Fie A = {0,1} un alfabet binar
- W_n multimea sirurilor w de lungime n peste A

  W= w[0] w[1]……...w[n-1]
- **Ponderea Hamming** a lui w = numarul de unitati (1) continute de w
- **Distanta Hamming**, d(u,v) dintre cuvintele u si v reprezinta numarul de simboluri diferite dintre cele doua. Este echivalent cu numarul minim de erori de 1 bit de care ar fi nevoie pentru a transforma unul din cuvinte in celalalt.

Detecţia erorilor si corectarea erorilor

Se considera ca pentru transmiterea de mesaje corecte se foloseste doar un subset din W_n

Multimea cuvintelor posibile W_n se imparte in:

- S_n – multimea cuvintelor cu sens
- F_n – multimea cuvintelor fara sens
- Pentru a detecta a cel mult *k* erori se alege S_n astfel ca:

$d(u,v) >= k+1$ pentru orice u,v din S_n

- Pentru a corecta cel mult *k* erori se alege S_n astfel ca:

$d(u,v) >= 2*k+1$ pentru orice u,v din S_n

Exemplu

S_{10} ={0000000000, 0000011111, 1111100000, 1111111111}

0000000000
0000011111
1111100000
1111111111

d(u,v) = 5 ====>>> putem detecta erori de maxim 4 biti
putem corecta erori de maxim 2 biti

Sender **Receiver**

0000<u>11111</u> ======>>> 0000000000

0000011<u>111</u>1 =======>>> 0000010001

Biți de paritate

- Metodă foarte simplă.
- Se adaugă un singur bit pentru fiecare caracter
- **Paritate pară** – numărul de biți 1 este par după adaugarea bitului
- **Paritate impară** – numărul de biți 1 este impar după adaugarea bitului

- Receptorul recalculează bitul de paritate
- Se consideră eroare dacă bitul primit diferă de cel recalculat

- Nu este o metodă foarte sigură. Dacă are loc un număr par de erori, verificarea bitului de paritate nu le va detecta.

7 biți de date	nr. de biți 1	8 biți cu paritate	
		pară	impară
0000000	0	0000000**0**	0000000**1**
1010001	3	1010001**1**	1010001**0**
1101001	4	1101001**0**	1101001**1**
1111111	7	1111111**1**	1111111**0**

Suma de control (checksum)

- Se calculează suma valorilor zecimale a fiecărui caracter din mesaj
- Se împarte rezultatul la 255
- Restul împărțirii se salvează și se adaugă la mesaj

- Are o eficiență de ~ 95%

Data (15B)	CHK(1B)

Metoda Hamming

Se considera un sir de n biti numerotati de la 1 la n.
Bitii 1,2,4,8…..(puteri ale lui 2) sunt **biti de control**

Exemplu: pentru un payload de 8 biti este nevoie sa trimitem 4 biti de control (1,2,4,8)

Paritatea pentru bitii de control poate fi **pară** sau **impară**, dupa urmatoarea regula:

Bit 1 – controleaza bitii 1,3,5,7,9,11
Bit 2 – controleaza bitii 2,3,6,7,10,11
Bit 4 – controleaza bitii 4,5,6,7,12
Bit 8 – controleaza bitii 8,9,10,11,12

Regulă: fiecare bit k este controlat de bitii ale caror pozitii insumate dau k.

Metoda Hamming

Exemplu: Se dorește trimiterea șirului de biți 11001010

Se adaugă biții de paritate la trimitere:

1	2	3	4	5	6	7	8	9	10	11	12
0	0	1	1	1	0	0	X	1	0	1	0

Obs: Codul Hamming poate corecta erorile de 1 bit

Metoda Hamming-Corectarea erorilor

Codul Hamming se poate folosi pentru corectarea erorilor de 1 bit.

Se pot corecta erorile in rafala (en. burst) in situatia urmatoare:

- Matricea de biti este transmisa coloana cu coloana
- Se pot corecta erori in rafala dintr-o coloana daca exista un bit eronat pe fiecare linie.

Char.	ASCII	Check bits
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
c	1100011	11111000011
o	1101111	10101011111
d	1100100	11111001100
e	1100101	00111000101

Order of bit transmission

Detecţia erorilor cu coduri polinomiale

k biţi de informaţie (date) ~ coeficientii unui polinom **i(X)** cu k termeni

Ex. k=5 10110 $i(X) = 1*X^4+0*X^3+1*X^2+1*X^1+0*X^0$

n-k biţi de control ~ polinom **r(X)** cu n-k termeni
concatenarea de **n** biţi ~ polinom de n termeni

$$\mathbf{w(X) = X^{(n-k)}i(X) + r(X)}$$

Pentru detectie erori
se foloseste un polinom generator **g(X)**
se calculeaza **r(X)** astfel ca **w(X)** sa fie multiplu de **g(X)**
se verifica daca proprietatea este pastrata la receptia cadrului
Calcul r(X):
w(X) = g(X).q(X)
$X^{(n-k)}i(X) + r(X) = g(X).q(X)$
$X^{(n-k)}i(X) = g(X).q(X) + r(X)$
r(X) este restul împărţirii lui $\mathbf{X^{(n-k)}\ i(X)}$ la **g(X)**

Calculul sumei de control

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

Message after 4 zero bits are appended: 1 1 0 1 0 1 1 0 1 1 0 0 0 0

Calculul sumei de control pentru un cod polinomial

10 biti informatie + 4 biti control

Imparte **11010110110000**

la 10011

$X^{(n-k)}\, i(X)$

$g(X)$

```
                1 1 0 0 0 0 1 0 1 0
          ------------------------------
10011   | 1 1 0 1 0 1 1 0 1 1 0 0 0 0
          1 0 0 1 1
          -----------
            1 0 0 1 1
            1 0 0 1 1
            ---------
              0 0 0 0 1
              0 0 0 0 0
              ---------
                0 0 0 1 0
                0 0 0 0 0
                ---------
                  0 0 1 0 1
                  0 0 0 0 0
                  ---------
                    0 1 0 1 1
                    0 0 0 0 0
                    ---------
                      1 0 1 1 0
                      1 0 0 1 1
                      ---------
                        0 1 0 1 0
                        0 0 0 0 0
                        ---------
                          1 0 1 0 0
                          1 0 0 1 1
                          ---------
                            0 1 1 1 0
                            0 0 0 0 0
                            ---------
                              1 1 1 0   <- Remainder
```

r(X) = rest împărţire X(n-k) i(X) la g(X)

Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 1 0

Ce erori pot fi detectate?

- Probabilitatea de detectie depinde de lungimea Codului cu Redundanta Ciclică (CRC)
  CRC–8–Bluetooth = $x^8 + x^7 + x^5 + x^2 + 1$
  CRC-8-CCITT = x^8+x^2+x+1
  CRC-12 = $x^{12}+x^{11}+x^3+x^2+x+1$
  CRC-16 = $x^{16}+x^{15}+x^2+1$
  CRC-32 = $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$ (Ethernet)

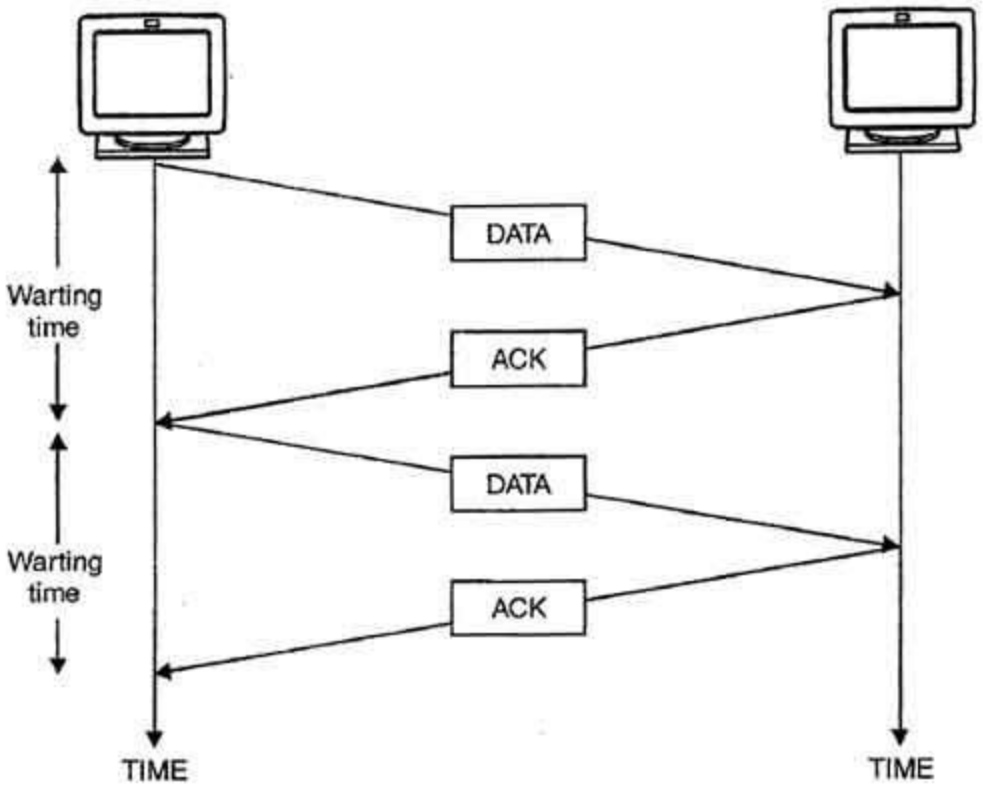
- CRC si sume de control pe
  - 8 biti detecteaza 99.6094% din erori (CRC-8-CCITT)
  - 16 biti detecteaza 99.9985% din erori (CRC-16)
  - 32 biti detecteaza 99.9999% din erori

- In plus, CRC detecteaza 100% erori de
  - 1 bit;
  - 2 biti;
  - un numar impar de biti;
  - erori in rafala de lungimea codului CRC.

Nivelul legăturii de date

Protocoale start-stop

Protocoale Stop and Wait



NACK

Stop & Wait Method.

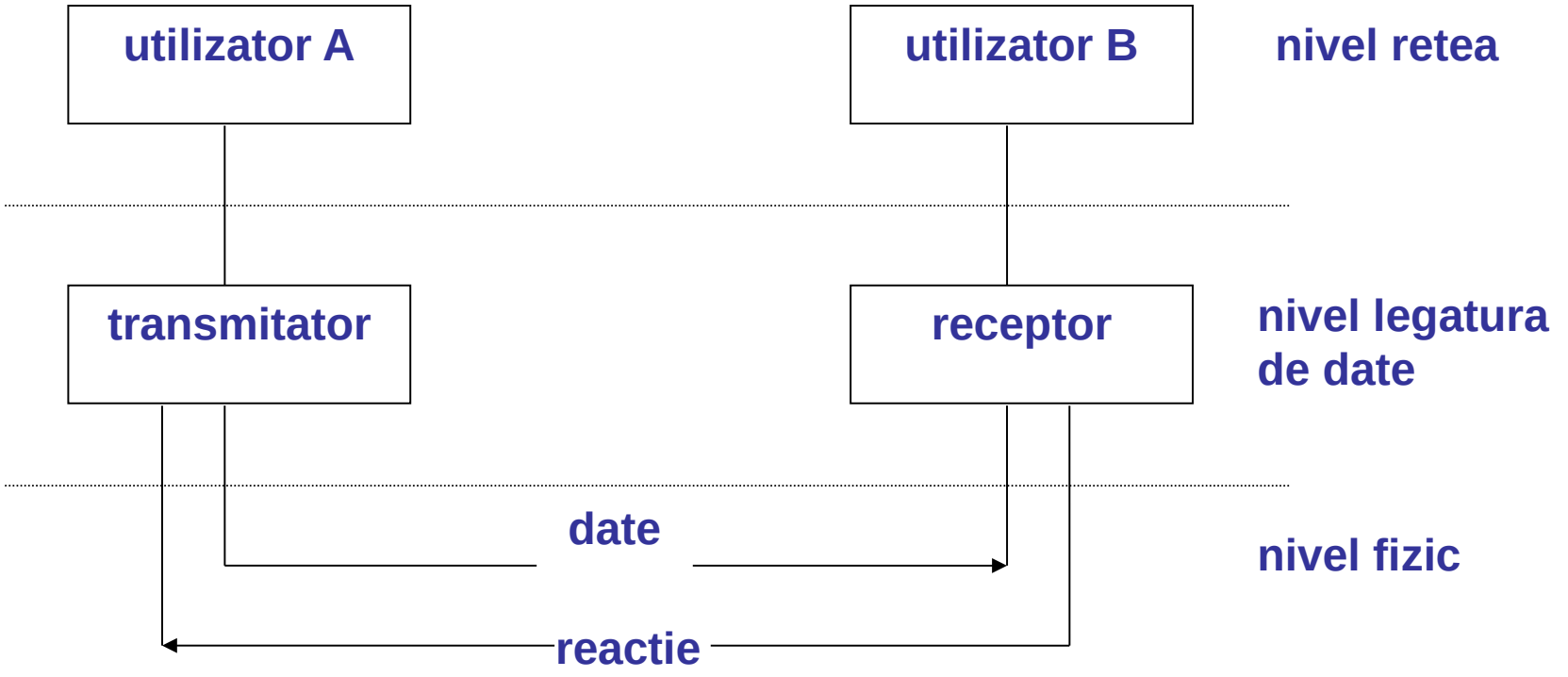
Protocol Stop and wait

- Sender-ul trimite un singur cadru si asteapta primirea unei confirmari (ACK)

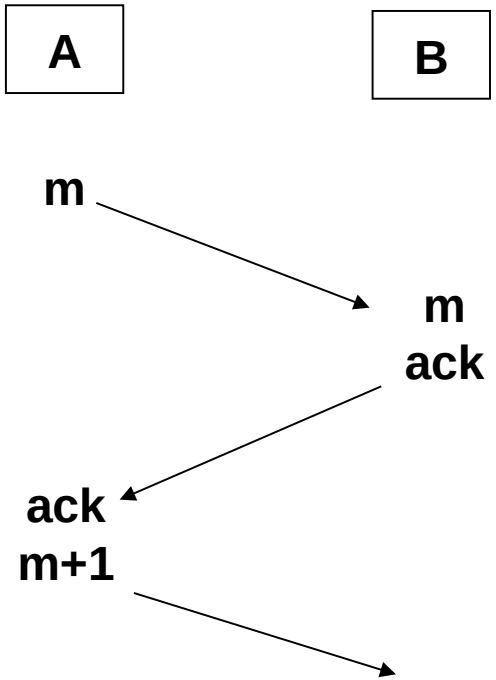
- Transfer bidirectional de informatie, se poate realiza printr-un canal half-duplex.

- Se doreste impartirea informatiei trimise in cadre de dimensiune mai mica deoarece:
    - Buffer-ul receiver-ului ar putea fi limitat
    - Cadrele mai mici au o probabilitate mai mica de eroare
    - In cazul unei erori retransmisia unui cadru mic va consuma mai putine resurse.
    - In cazul folosirii unui mediu partajat de transmisie (LAN), un singur sender nu va ocupa mediul pentru un timp prea lung.

Protocol start-stop

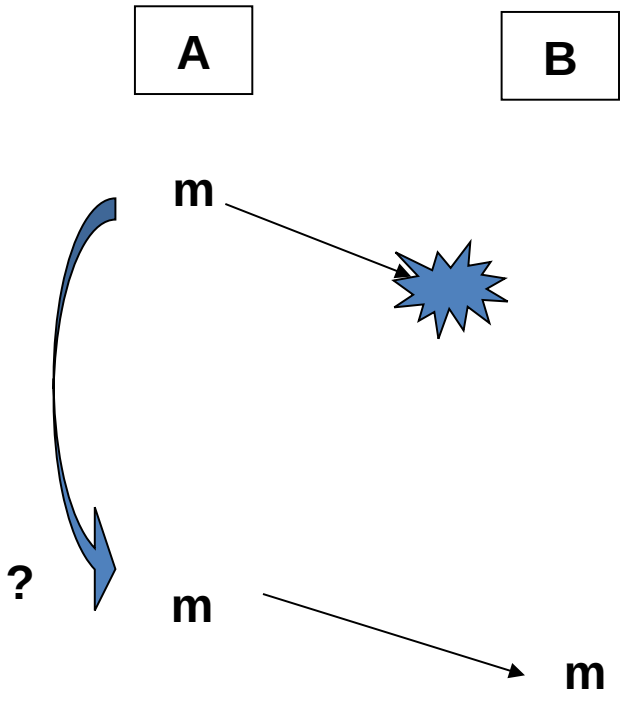


Protocol simplex pentru un canal cu erori

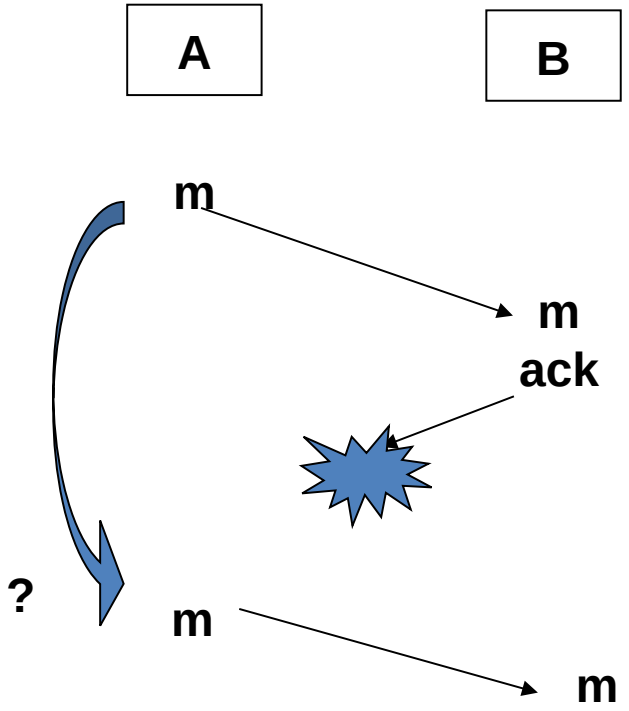


Entitaţi legătură de date

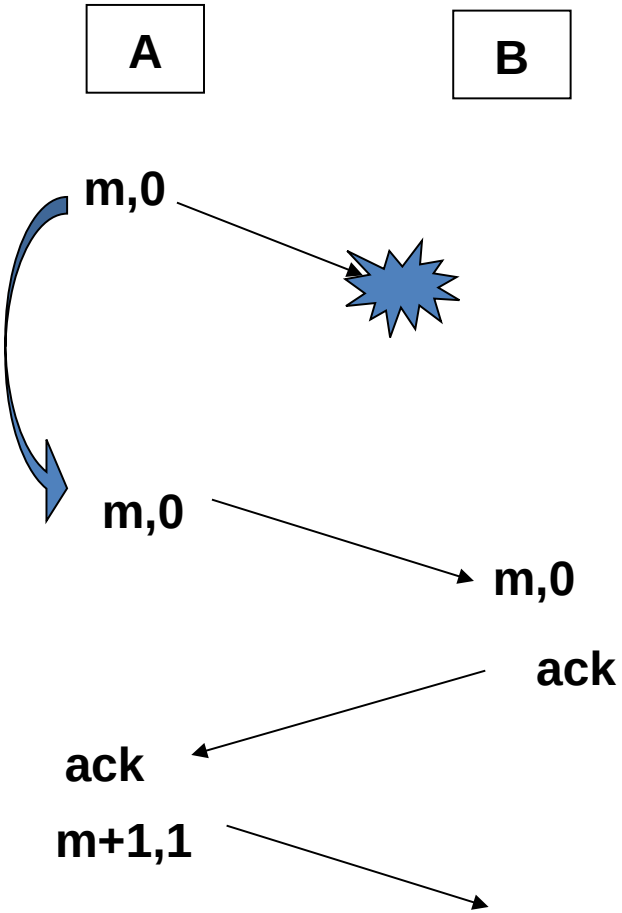
Transmisie corecta

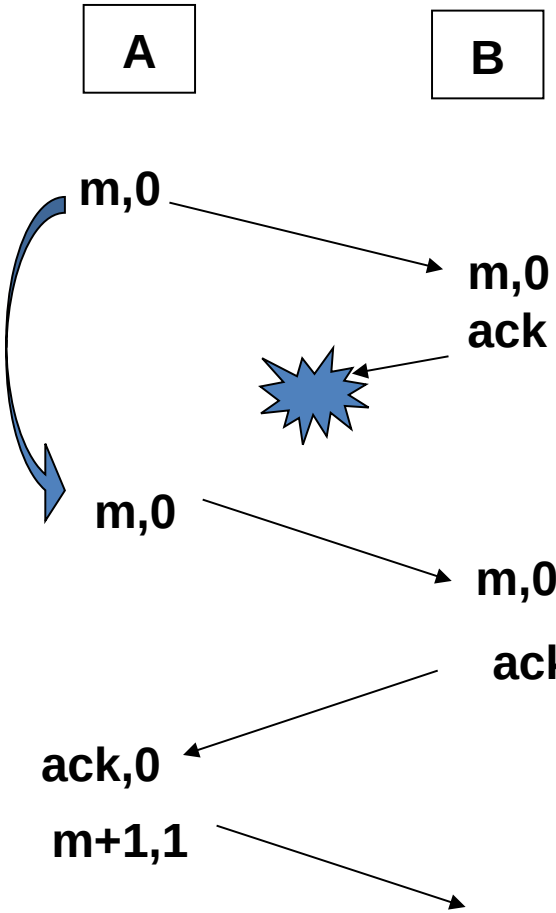


Pierdere m
La time-out A retransmite m
Care este acceptat corect de B



Pierdere ack
La time-out se retrimite m
Care este acceptat incorect, ca mesaj nou de B !





- accepta

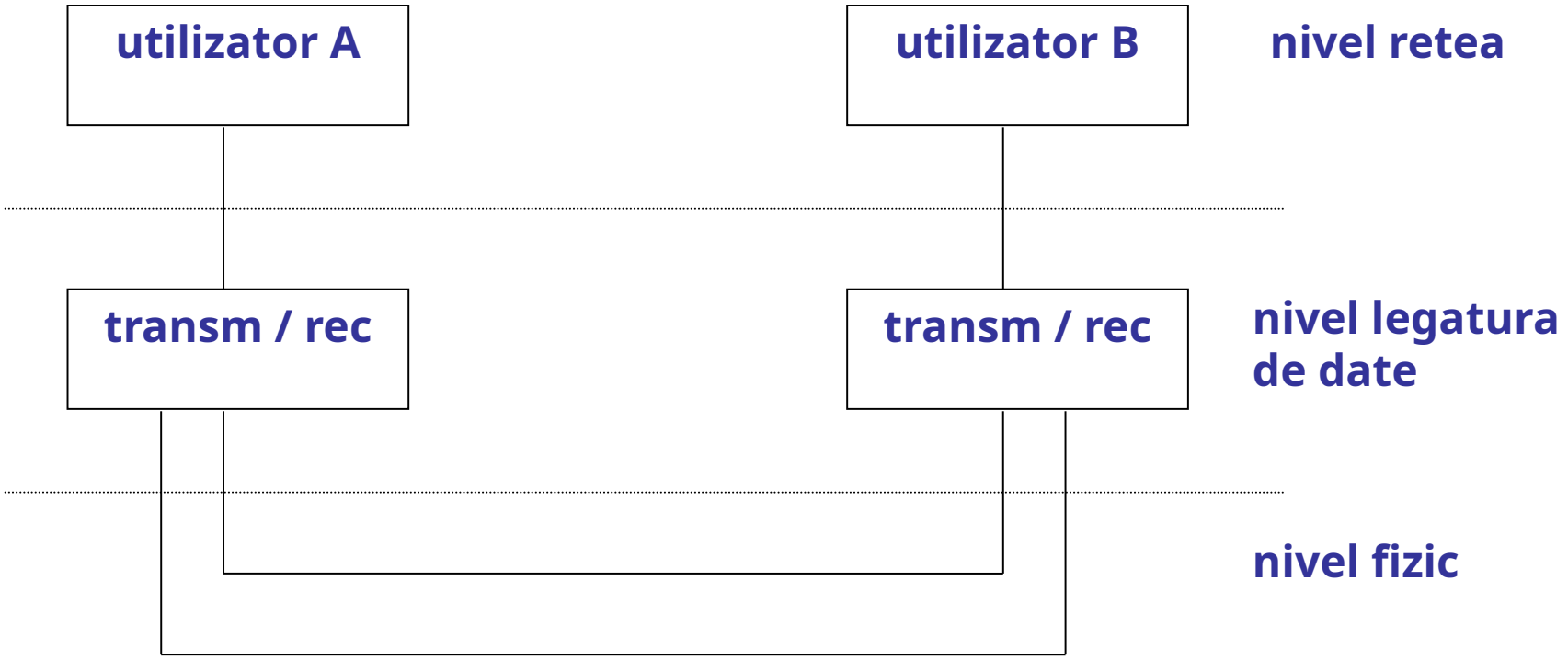
- ignora

se adauga un numar de secventa
la time-out se re-transmite ultimul cadru
B accepta daca este corect

B ignora daca este dublura
dar re-trimite ack !

Protocoale cu fereastră glisantă

Configuraţia generala



2 legaturi pentru date+confirmare

Protocoale cu fereastra glisanta

- Daca este nevoie de comunicatie full duplex se folosesc 2 canale separate de comunicatie: unul pentru trimiterea datelor si al doilea pentru confirmari ACK/NACK.

- Sender-ul pastreaza o lista de cadre ce trebuie sa fie trimise. Receiver-ul pastreaza o lista de cadre pe care le poate accepta.

- Numerele de secventa pentru cadre trimise dar pentru care inca nu s-a primit un ACK sunt pastrate in fereastra glisanta.

- **Numerele de secventa** din ferestrele glisante sunt consecutive.

Fereastra glisanta de 1 bit

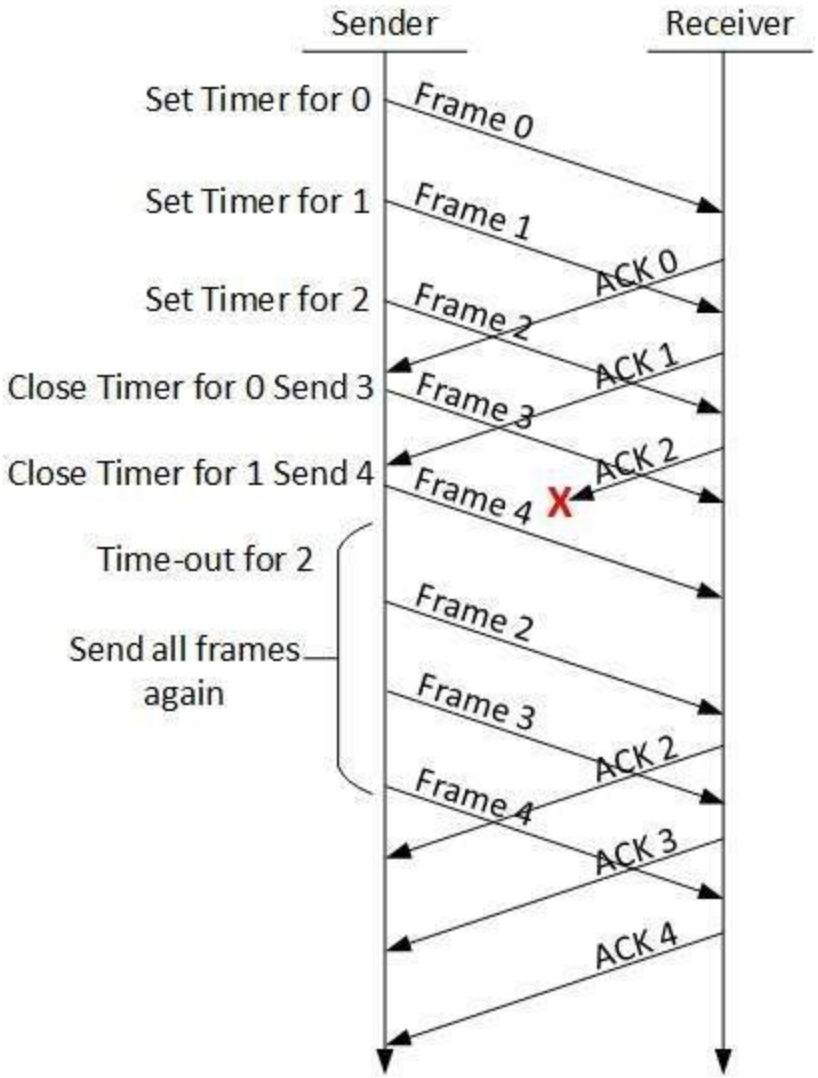
- **Fiecare entitate a**re o secventa de initializare in care trimite un prim cadru

- **si realizează ciclic următoarele operaţii:**

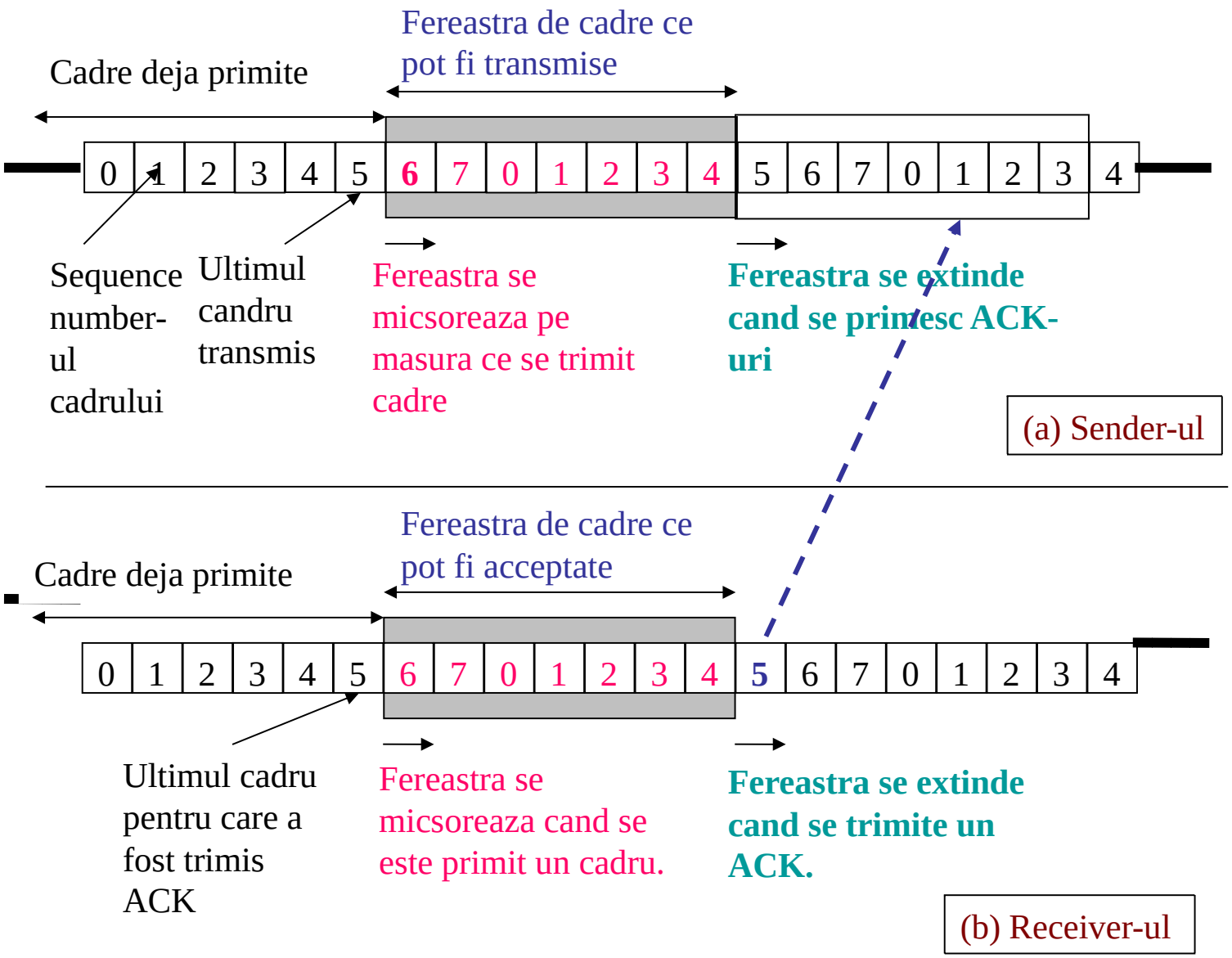
  - receptia unui cadru,
  - prelucrarea sirului de cadre receptionate,
  - prelucrarea sirului de cadre transmise,
- transmiterea sau retransmiterea unui cadru impreuna cu confirmarea cadrului receptionat corect.

- Se ajunge la modul de transmisie Stop and wait, in care sender-ul trimite un cadru si asteapta primirea unui ACK inainte sa il trimita pe urmatorul.

Protocoale cu fereastra glisanta



Protocoale cu fereastra glisanta



Protocoale cu fereastră supraunitară de transmisie

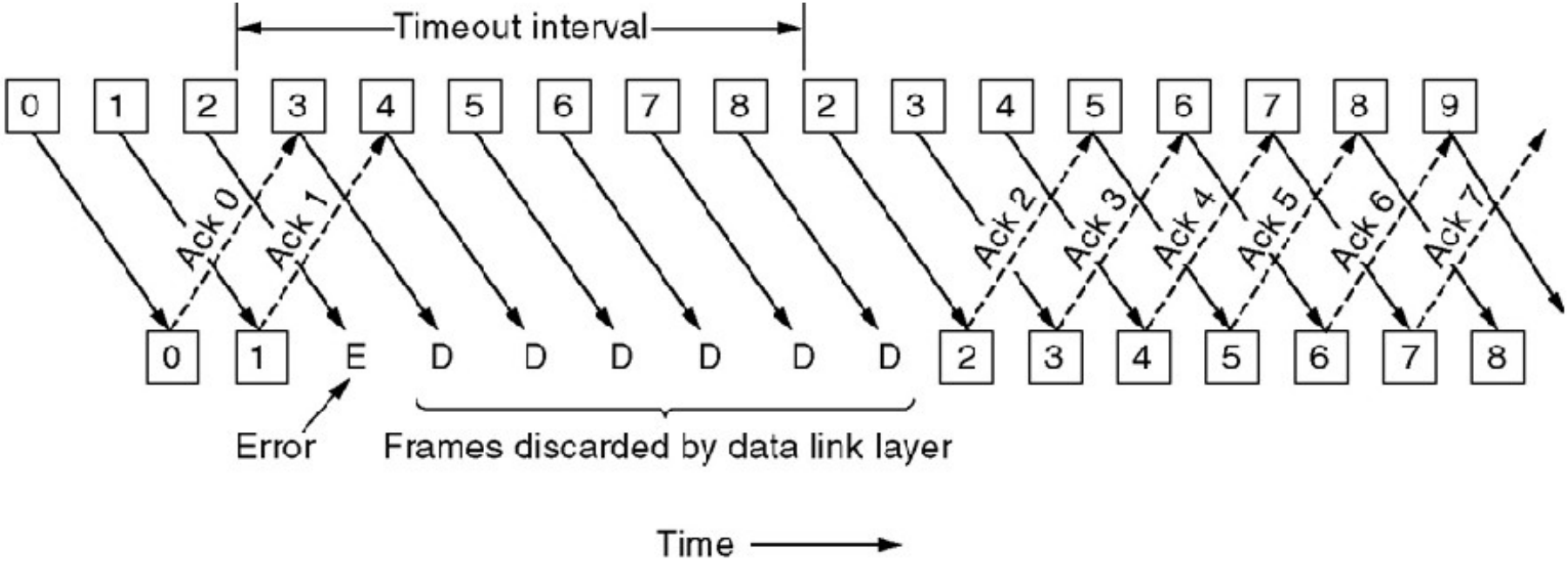
Protocol cu retransmitere neselectivă (Go back n)

Sunt **`MaxSecv + 1`** numere de secventa diferite
Fereastra maxima a transmitatorului poate fi de **`MaxSecv`** cadre
Demonstratie pe scenariu cu **`MaxSecv = 7`** si fereastra de 8

1. Transmitatorul trimite cadrele 0..7;
2. Toate cadrele sunt receptionate si confirmate;
3. Toate confirmarile sunt pierdute;
4. Transmitatorul retrimite la time-out toate cadrele;
5. Receptorul accepta duplicatele.

Un protocol cu retransmitere neselectivă “Go Back N”

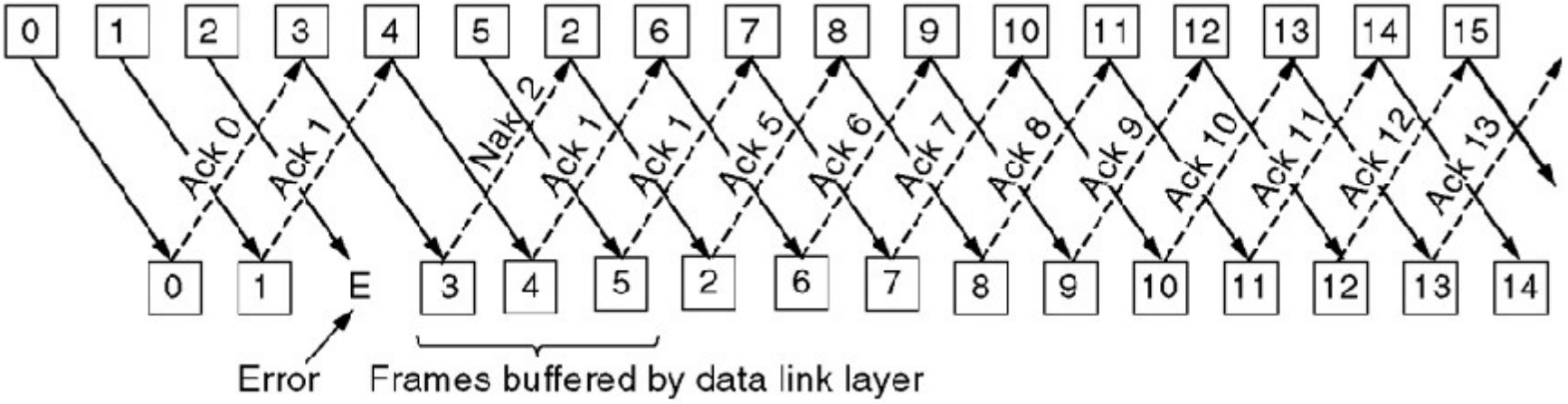
Efectul erorii cand fereastra receptorului este 1.



Protocol cu retransmitere selectiva "selective repeat"

Fereastra receptorului este supraunitara

- transmite Nak cu numarul 2 pentru cadru eronat
- apoi reconfirma ultimul cadru corect receptionat (Ack 1)
- dupa re-primirea cadrului eronat, Ack 5 confirma toate cadrele pastrate in buffer



Dimensiunea ferestrei de receptie

Fereastra receptorului nu poate fi egală cu cea a transmiţătorului

1. Transmitatorul trimite cadrele 0..6
2. Cadrele sunt receptionate si confirmate. Fereastra receptorului devine 7, 0, 1, 2, 3, 4, 5
3. Toate confirmarile sunt pierdute (se strica sincronizarea intre transmitator si receptor)
4. Transmitatorul retrimite cadrul 0 la time-out
5. Receptorul accepta drept cadru nou aceasta copie (cadrul 0) care se potriveste in fereastra sa actuala (7,0,1,2,3,4,5); cere cadrul 7 (dinaintea lui 0) care lipseste
6. Transmitatorul interpreteaza ca a trimis corect cadrele 0..6 si trimite 7, 0, 1, 2, 3, 4, 5
7. Receptorul accepta cadrele, cu exceptia lui 0, pentru care are deja un cadru receptionat. Ignora acest cadru 0 (a luat în loc duplicatul 0 primit anterior).

Optimizari protocoale cu fereastra glisanta

- Receiver-ul poate confirma un cadru si sa anunte ca asteapta urmatoarele (**Receive Ready**) sau poate sa nu permita trimiterea de alte cadre (**Receive Not Ready**). Este necesara trimiterea unui Acknowledgement normal pentru a relua trimiterea datelor dupa un cadru RNR.
- Este ineficienta trimiterea unui cadru ACK separat – se foloseste **piggybacking**. ACK-ul este atasat intr-un cadru de date (camp din header pentru ACK)

Nivelul legăturii de date

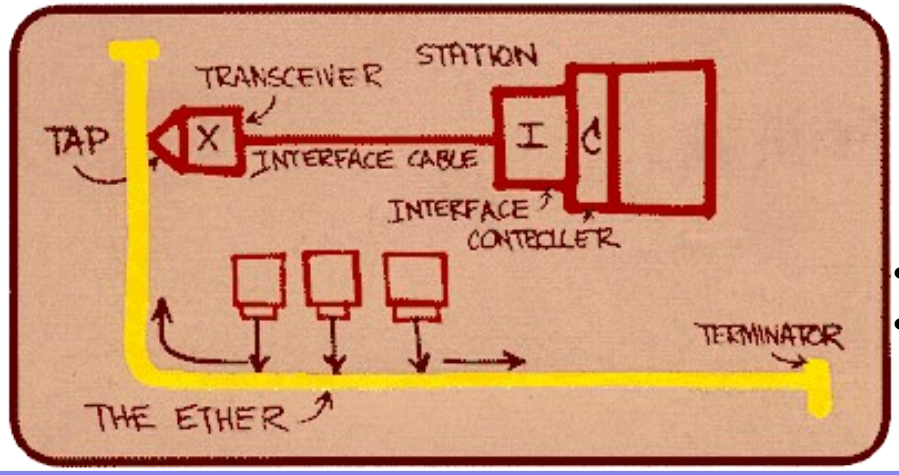
Exemple de protocoale

Exemple Protocoale Legatura de date

- Ethernet
- PPP
- PPPoE

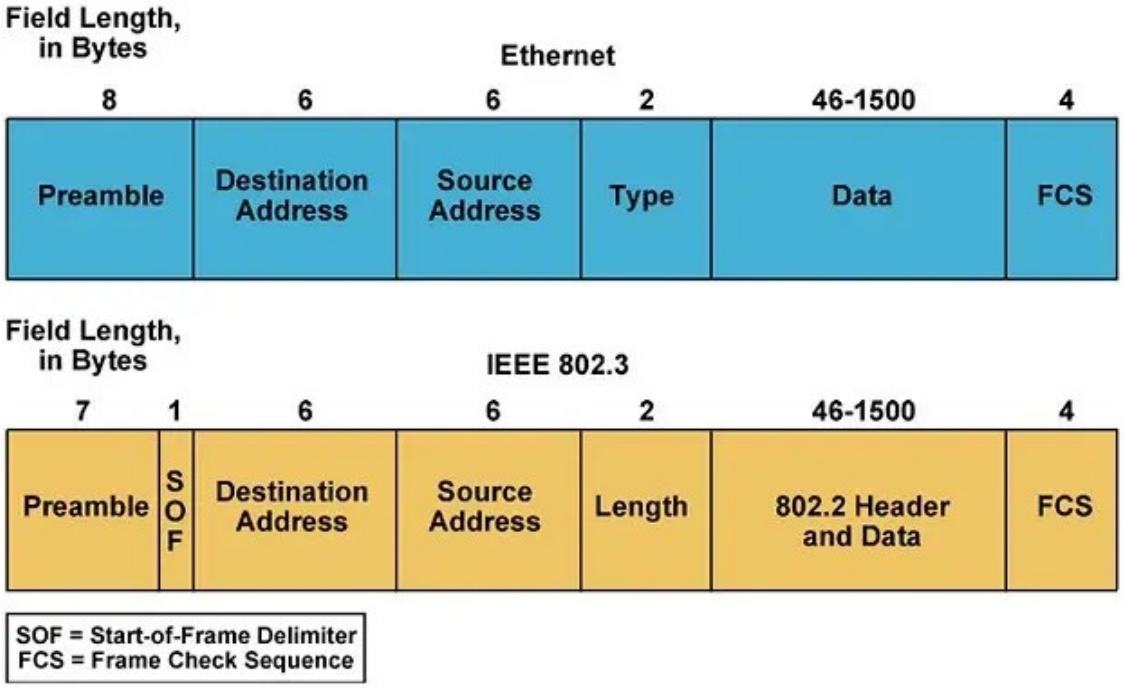
Protocolul Ethernet (IEEE 802.3)

- Cel mai folosit protocol LAN, dezvoltat de catre Digital, Intel si Xerox in anii 1970 de catre Bob Metcalfe si altii.
- Acum este un standard IEEE
- Protocol de nivel legatura de date, proiectat ca protocol broadcast
- Nu are probleme de transmisie transparenta a datelor
  - Foloseste un camp care contine numarul de bytes pentru a delimita cadrele
- Corectia erorilor se realizeaza folosind CRC-32



- Bob Metcalfe –
- schita initiala pentru Ethernet

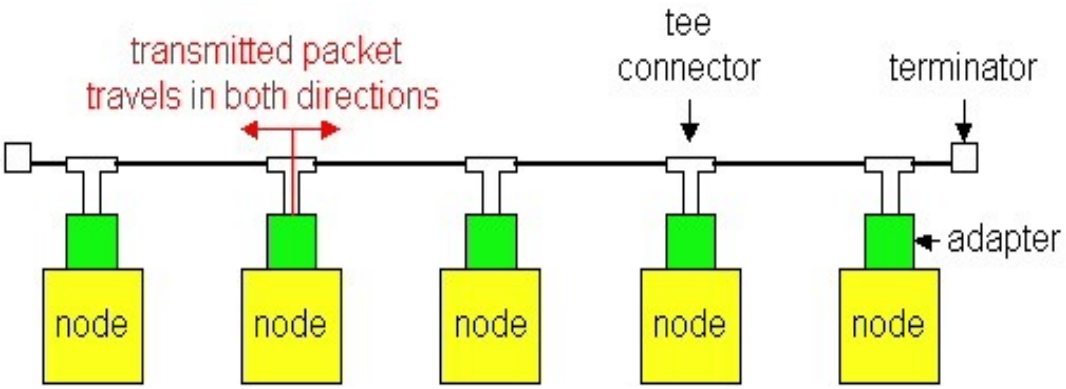
Protocolul Ethernet (IEEE 802.3)

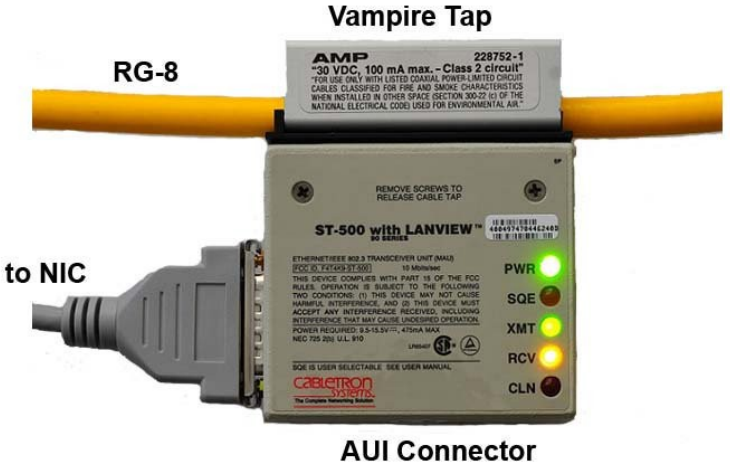


- **Preamble** – un pattern de biti 0 si 1 care alterneaza (01010101….)
- **SOF** – delimiteaza inceputul cadrului (10101011)
- **FCS** – verificarea corectitudinii transmisiei folosind CRC-32

Protocolul Ethernet (IEEE 802.3)

- Cadrul poate sa aiba minim 64 bytes si maxim 1516 bytes (fara preambul) – 12 bytes adrese, 2 bytes lungimea cadrului, 46 bytes date, 4 bytes CRC
- Daca un host are de trimis mai putin de 46 de bytes, transmitatorul va adauga padding.

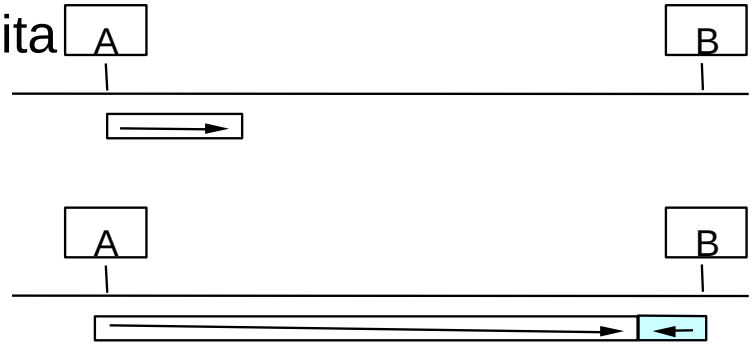




Coliziuni

- Coliziunile apar cand 2 adaptoare incearca sa trimita in acelasi timp
- Daca este detectata o coliziune se asteapta si se incearca din nou dupa un timp random
- Modul de trimitere se bazeaza pe Carrier-sense multiple access with collision detection (CSMA/CD).
  - MA = multiple access
  - CS = carrier sense
  - CD = collision detection

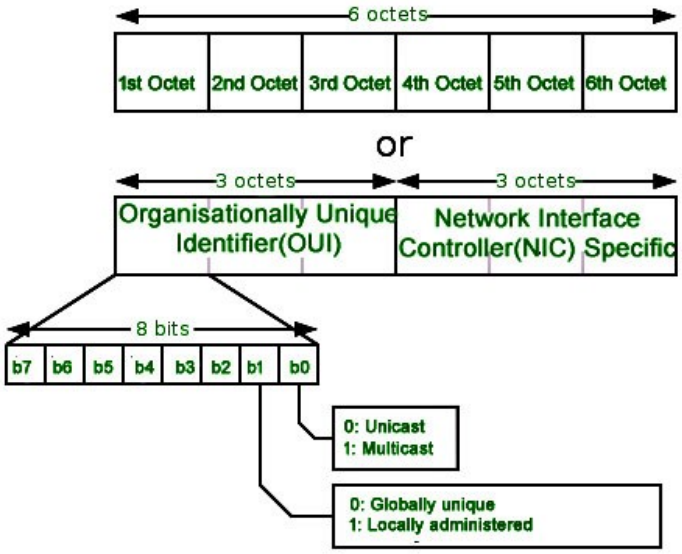
- A incepe sa trimita
- la timpul 0



- Mesajul aproape a ajuns
- la destinatie cand B incepe
- sa trimita --- > Coliziune!

Protocolul Ethernet (IEEE 802.3)

- **Foloseste adrese MAC de 6 bytes** (adrese de nivel Legatura de Date). Exemplu de adresa MAC: B0:60:88:AE:6E:15
- **Dezavantaj**: A fost proiectat ca un protocol de tip broadcast, ceea ce ii limiteaza performanta.
- Nu are o negociere a parametrilor de comunicatie.

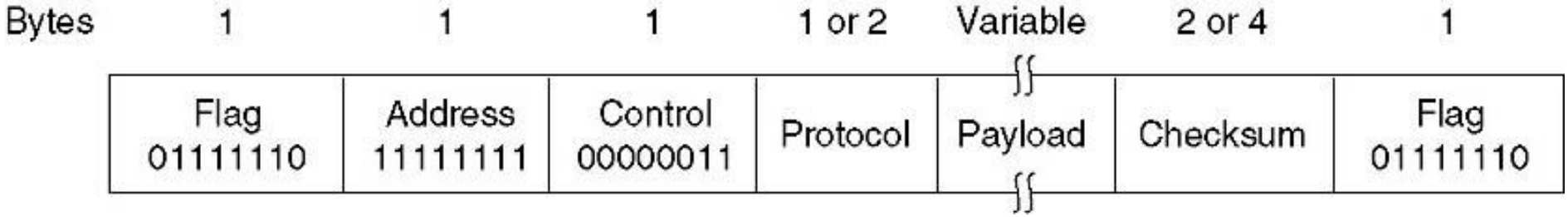


Caracteristici Ethernet

- Protocolul functioneaza cel mai bine pentru incarcare redusa (o incarcare de 30% este considerata foarte mult)
- Este un protocol simplu, usor de administrat.
- Foloseste algoritmul CDMA/CD pentru managementul coliziunilor
- Tipuri de Ethernet
  - Fast Ethernet 100 Mbps
  - Gigabit Ethernet 1Gbps
  - 10Gbps

PPP – Point to Point Protocol

Ofera incadrare
Link Control Protocol, LCP
Network Control Protocol, NCP

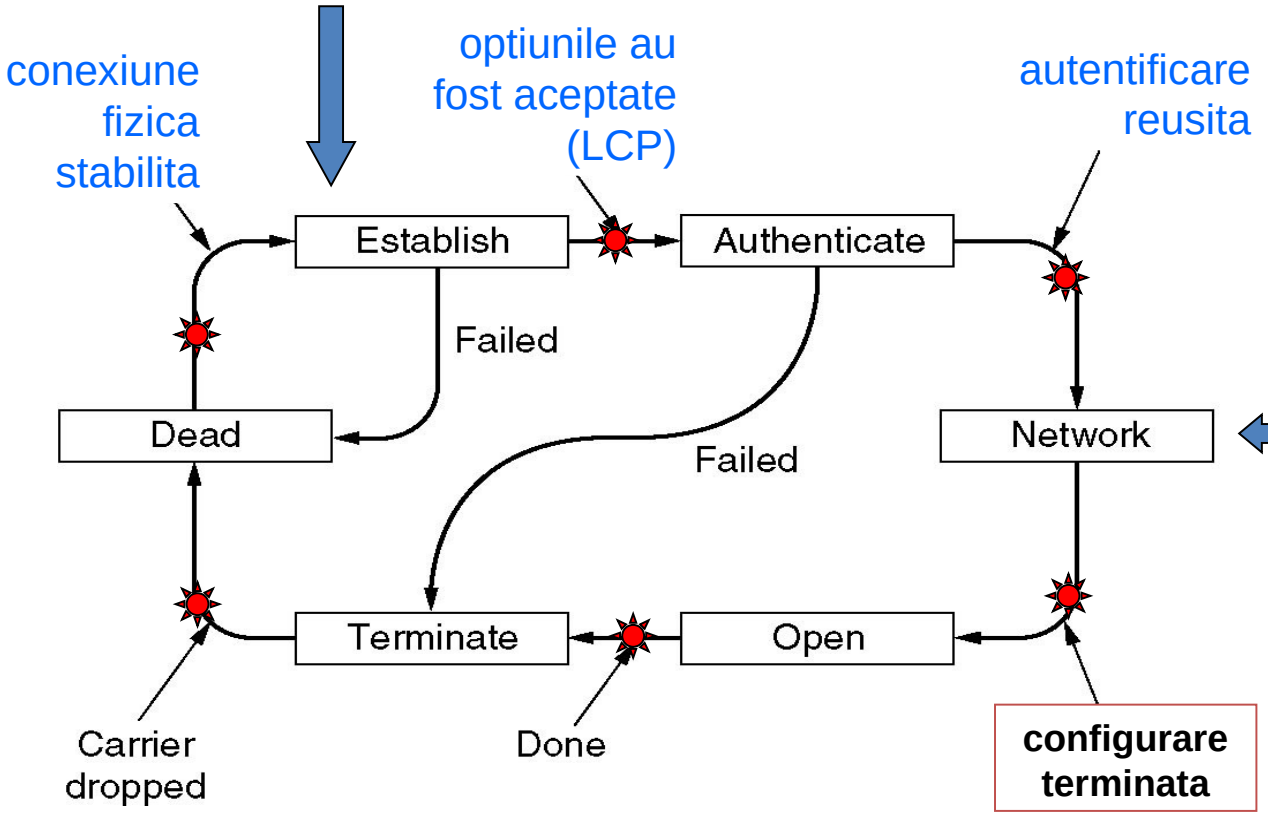


Format de cadru PPP pentru modul nenumerotat

Addresa 11111111 = toate statiile accepta cadrul
Control 00000011 = nenumerotat
Lungimea default a datelor este 1500 bytes
Protocol = selecteaza dintre
LCP, NCP
IP, IPX (Internetwork Packet eXchange), OSI CLNP, XNS (Xerox Network Services)

PPP – Point to Point Protocol (2)

Tipuri de cadre LCP
Configure-request **Configure-reject**
Configure-ack **Code-reject**
Configure-nak **Protocol-reject**



protocolul de retea este configurat folosind NCP (ex. ptr IP – alocare adresa IP)

Inchide conexiune
Terminate-request
Terminate-ack

PPP transporta pachete (de ex. IP)

Tipuri de cadre LCP

Name	Direction	Description
Configure-request	I → R	List of proposed options and values
Configure-ack	I ← R	All options are accepted
Configure-nak	I ← R	Some options are not accepted
Configure-reject	I ← R	Some options are not negotiable
Terminate-request	I → R	Request to shut the line down
Terminate-ack	I ← R	OK, line shut down
Code-reject	I ← R	Unknown request received
Protocol-reject	I ← R	Unknown protocol requested
Echo-request	I → R	Please send this frame back
Echo-reply	I ← R	Here is the frame back
Discard-request	I → R	Just discard this frame (for testing)

I - Initiator **R - Responder**

Point To Point Protocol over Ethernet (PPPoE)

- Publicat in RFC 2516 (1999)

- Extinde capabilitatile protocolului PPP pentru a putea crea o conexiune punct-la-punct peste o retea Ethernet

- Este folosit de catre ISP (Internet Service Provider) pentru a gestiona accesul la retea pentru clienti. Companiile care ofera servicii de Internet pot folosi aceleasi mecanisme de autentificare pentru sesiunile PPPoE si PPP.

- Protocolul este configurat ca o legatura de date punct-la-punct intre doua porturi Ethernet.

Point To Point Protocol over Ethernet (PPPoE)

Pasii protocolului:

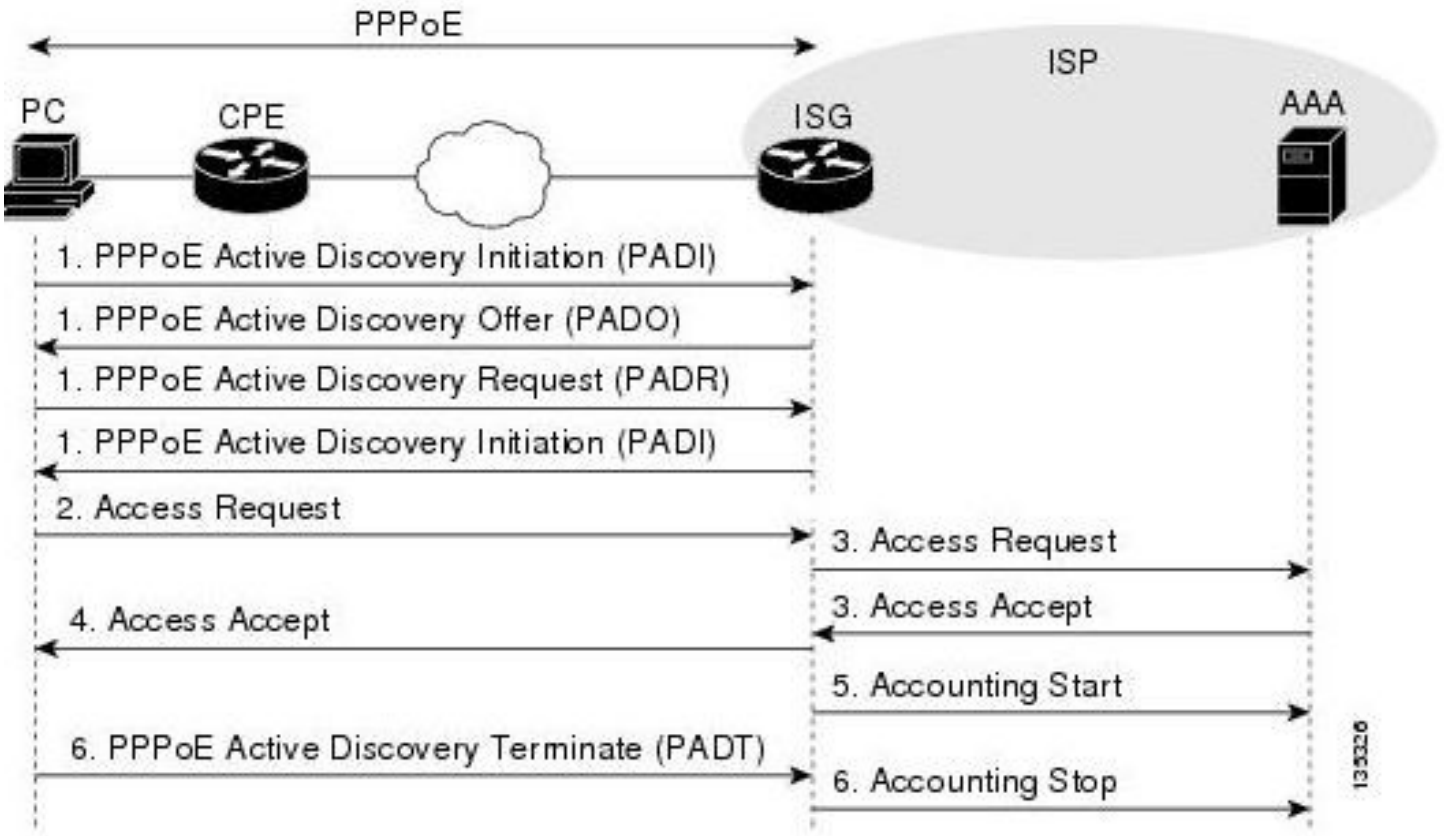
1) Initiation – Clientul trimite un pachet **PPPoE Active Discovery Initiation (PADI)** pentru a initia sesiunea.

2) Offer – Serverul raspunde cu un pachet **PPPoE Active Discovery Offer (PADO).**

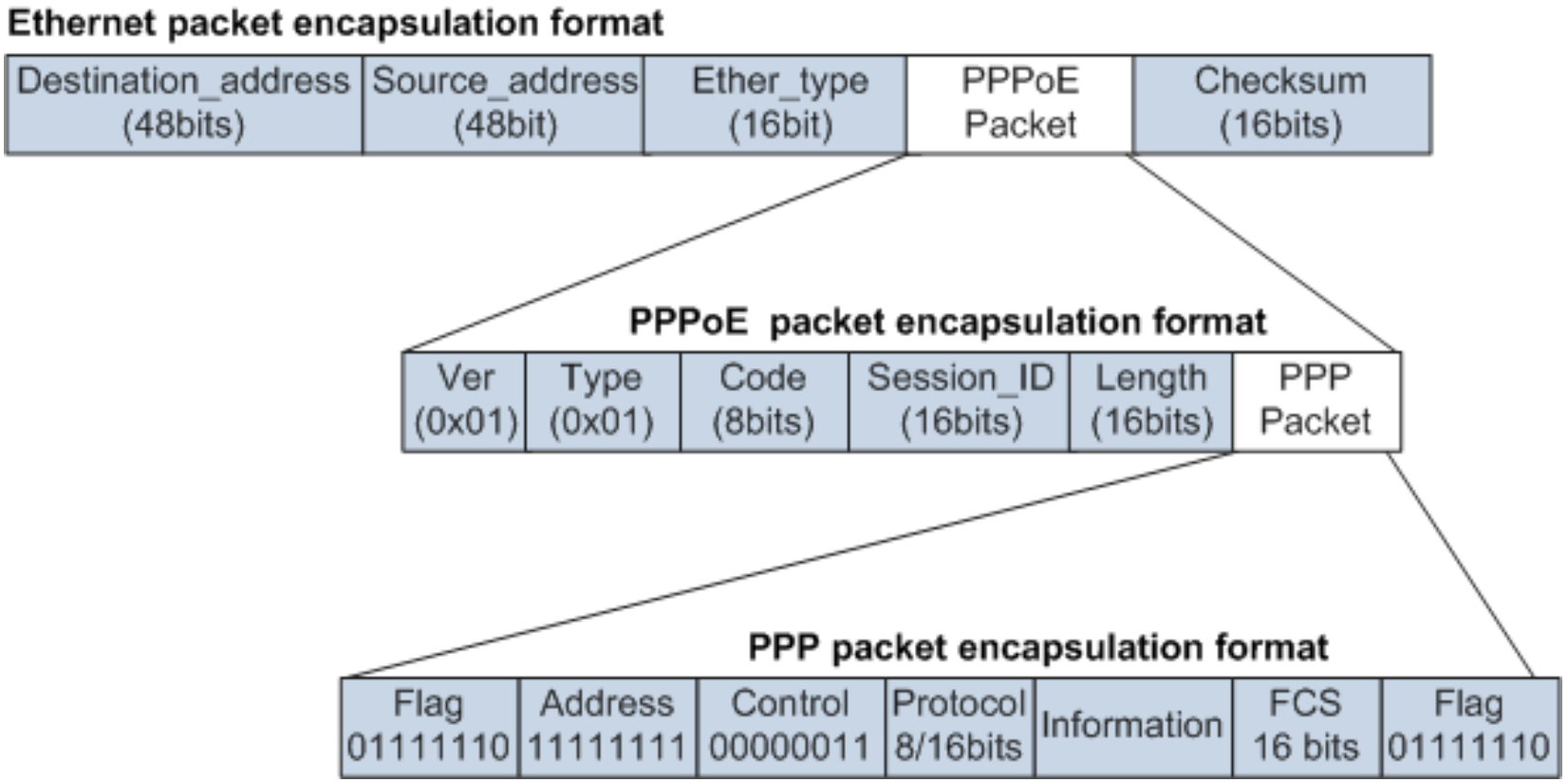
3) Request – Dupa primirea unui pachet PADO, clientul raspunde prin trimiterea unui pachet **PPPoE Active Discovery Request (PADR)** catre server.

4) Confirmation – Dupa primirea unui pachet PADR, serverul raspunde prin generarea unui ID unic pentru sesiunea PPP si trimite un pachet **PPPoE Active Discovery Session (PADS)** de confirmare catre client.

Point To Point Protocol over Ethernet (PPPoE)



Point To Point Protocol over Ethernet (PPPoE)



Point To Point Protocol over Ethernet (PPPoE)

Field	Description
Destination_address	Indicates either a unicast Ethernet destination address or the Ethernet broadcast address (0xffffffffffff). For Discovery packets, the value is either a unicast or broadcast address. A PPPoE client uses a broadcast address to search for a PPPoE server and uses a unicast address after choosing a PPPoE server. For PPP session traffic, this field must contain the peer's unicast address as determined from the Discovery stage.
Source_address	Indicates the Ethernet MAC address of a source device.
Ether_type	Set to either 0x8863 (Discovery stage) or 0x8864 (Session stage).
Ver	Indicates a PPPoE version number. The **Ver** field is four bits and must be set to 0x1.
Type	Indicates a PPPoE type. The **Type** field is four bits and must be set to 0x1.
Code	Indicates a PPPoE packet type. The **Code** field is eight bits and has different values: • 0x00: session data • 0x09: PADI packet • 0x07: PADO or PADT packet • 0x19: PADR packet • 0x65: PADS packet
Session_ID	The **SESSION_ID** field is sixteen bits. The value is fixed for a given PPP session and defines a PPP session along with Ethernet source and destination addresses. A value of 0xffff is reserved for future use and must not be used.
Length	Indicates the length of the PPPoE payload. The **Length** field is 16 bits, excluding the length of the Ethernet or PPPoE header.
PPP Packet	For PPP packet format, see PPP Basic Concepts-PPP Packet Format.

Nivelul legăturii de date

Metrici de performanta

Conventii

1 B = 8 biti

In transmisii de date

1 kb = 10^3 b = 1 000 b

1 Mb = 10^6 b = 1 000 000 b

1 Gb = 10^9 b = 1 000 000 000 b

In calculatoare, volumul datelor

1 kb = 2^{10} b = 1 024 b

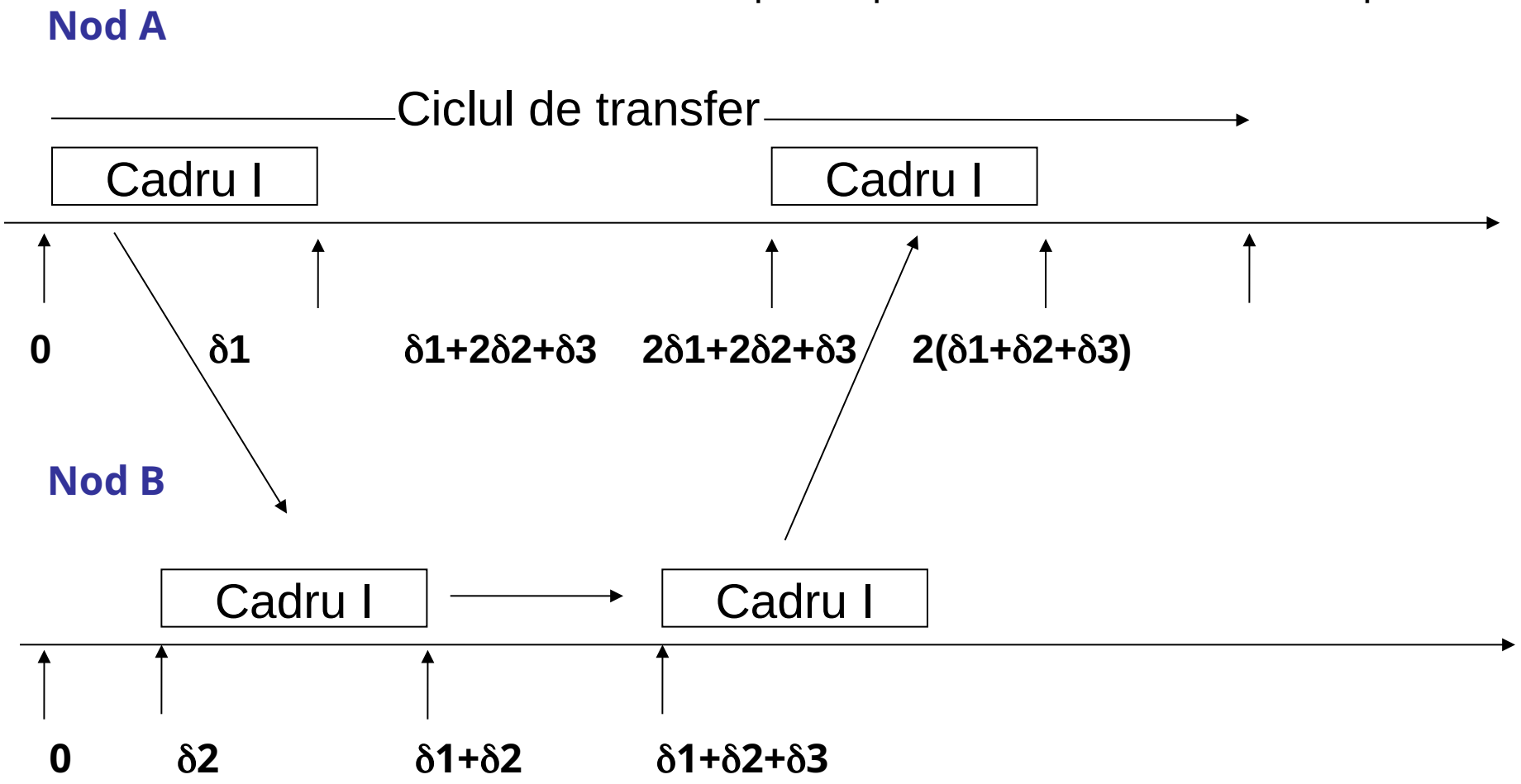
1 Mb = 2^{20} b = 1 048 576 b

1 Gb = 2^{30} b = 1 073 741 824 b

- Analiza performanţelor protocoalelor start-stop

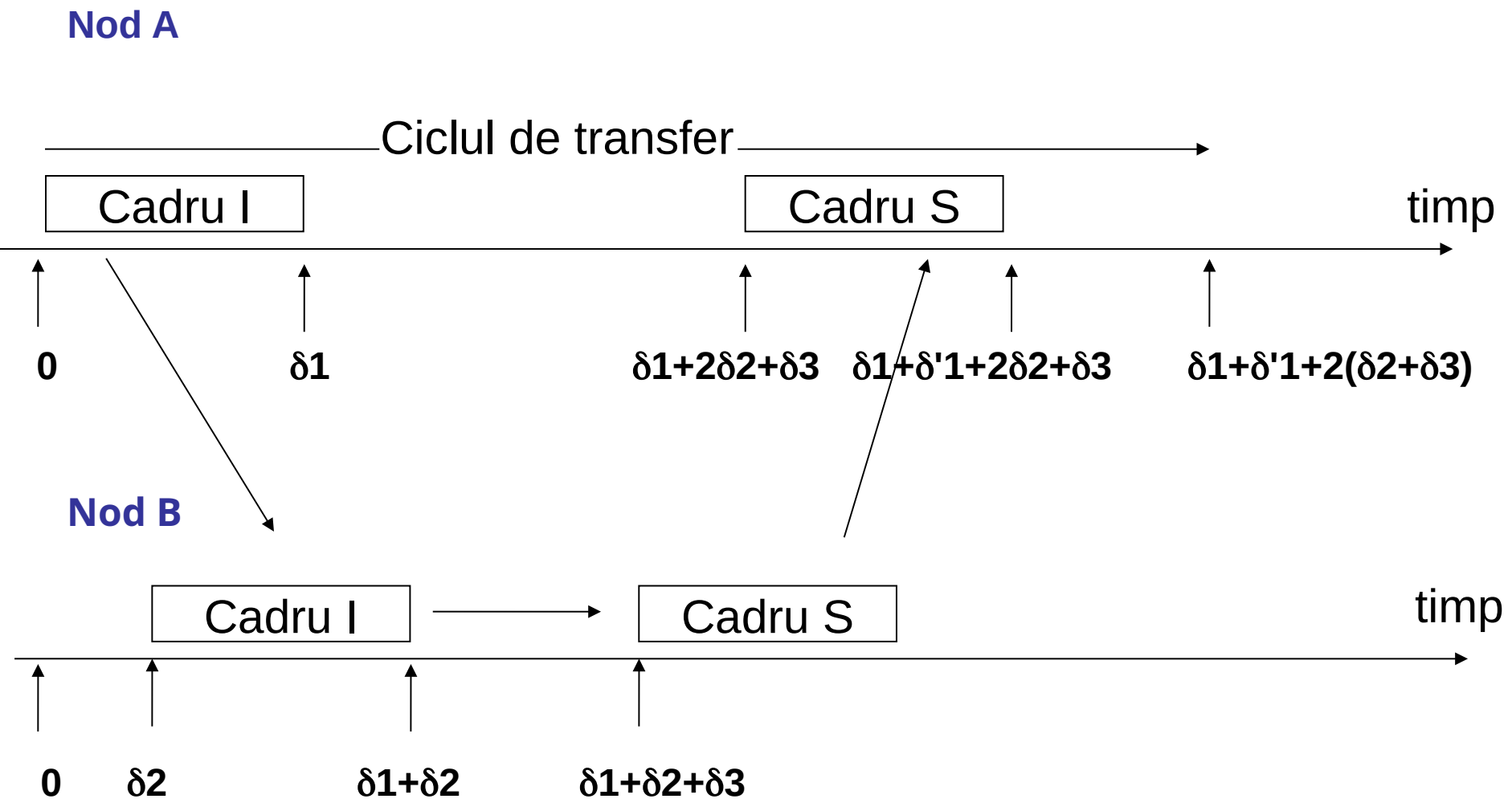
Transmitere cu confirmare în cadre de informatie I

∂1 – durata de transmitere a unui cadru I (sec)
∂2 – intarzierea de transmisie
∂3 – timpul de prelucrare a cadrului la receptor.



Transmitere cu confirmare în cadru supervizor S

∂'1 – durata de transmitere a unui cadru S (sec)



Throughput

- O metoda mai buna decat eficienta
- Numarul total de biti transmisi, tine cont de
  - Bitii de overhead
  - Nevoia de a retransmite unele cadre
- Complex de calculat, depinde de
  - Eficienta transmisiei
  - Rata de erori
  - Numarul de retransmiteri
- Transmission Rate of Information Bits (TRIB)
  - Folosit ca o metoda de a masura throughput-ul

• TRIB

= Numarul de biti de informatie primiti / timpul total necesar

(numarul de biti de informatie) (Probabilitatea de transmisie corecta)
timpul necesar de transmisie + intarzierea de propagare

Biti de informatie per caracter

Media numarului de biti care nu sunt biti de informatie

Probabilitatea ca un cadru va avea nevoie de retransmisie

$$TRIB = \frac{K\,(M - C)\,(1 - P)}{(M / R) + T}$$

Lungimea unui cadru

Rata de transmisie pe secunda

Timpul intre cadre (intarziere, timp de procesare, etc.)

Ex:
K=7 bits/character
M = 400 char/cadru
R= 4.8 Kb/s
C = 10 char/block
P = 1%
T = 25 ms

$$TRIB = \frac{7(400-10)(1-0.01)}{(400/600)+0.025}$$

= 3.908 Kb/s

• Dimensiunea unui cadru