

# **Il livello “collegamento dati”**

**Gruppo Reti TLC**

**[giancarlo.pirani@telecomitalia.it](mailto:giancarlo.pirani@telecomitalia.it)**

**<http://www.telematica.polito.it/>**

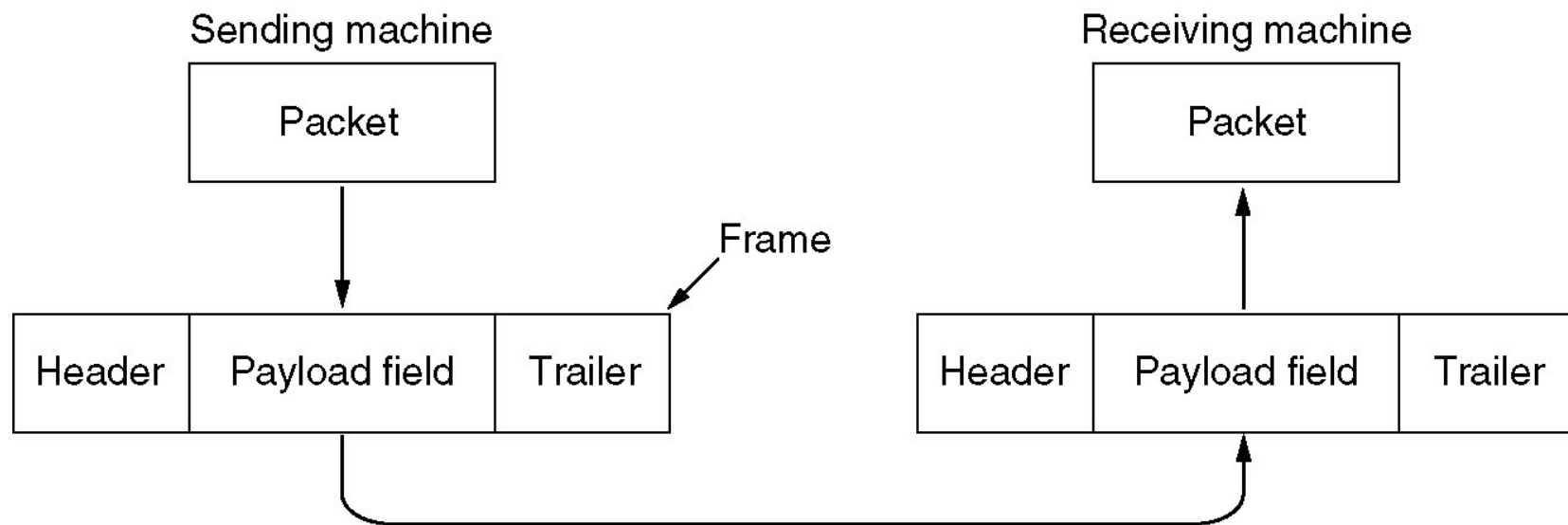


## **Funzioni dello Strato 2 – Collegamento dati (Data Link Layer)**

- **Obiettivo principale: elevare le prestazioni di una linea fisica con un dato tasso di errore offrendo allo strato di rete un servizio di collegamento dati privo di errori**
- **Fornire un'interfaccia di servizi allo strato superiore (strato di rete)**
- **Controllare gli errori di trasmissione**
- **Regolare il flusso di dati**
  - **Evitare che ricevitori più lenti siano “ingolfati” da trasmettitori veloci**

## Funzioni dello Strato 2 – Collegamento dati (2)

### Relazioni tra pacchetti e trame



# Servizi offerti allo strato “Rete”

- **Servizio connectionless senza notifica di ricezione**

Non e' stabilita nessuna connessione prima dell'invio di unità informative. Non si fa nessun tentativo per rilevare o recuperare la perdita a livello di questo strato. "Late data are worse than bad data" (traffico real-time). La maggior parte delle LAN utilizzano questo tipo di servizio nello strato 2.

- **Servizio connectionless con notifica di ricezione**

Non ci sono connessioni logiche stabilite prima dell'invio, ma viene mandata una notifica di ricezione per ogni frame inviato. E' utile per canali inaffidabili, come ad esempio nei sistemi wireless. Possibili time-out se il frame non arriva entro un certo tempo. Se un pacchetto e' suddiviso in tanti frame e' piu' veloce notificare i singoli frame che non l'intero pacchetto.

- **Servizio connection-oriented con notifica di ricezione**

La sorgente e la destinazione stabiliscono la connessione prima di iniziare il trasferimento dei dati. Ogni frame e' numerato, in ricezione viene mantenuto l'ordine con cui viene trasmesso e ne viene notificata la ricezione.

# Posizione del protocollo “collegamento dati”





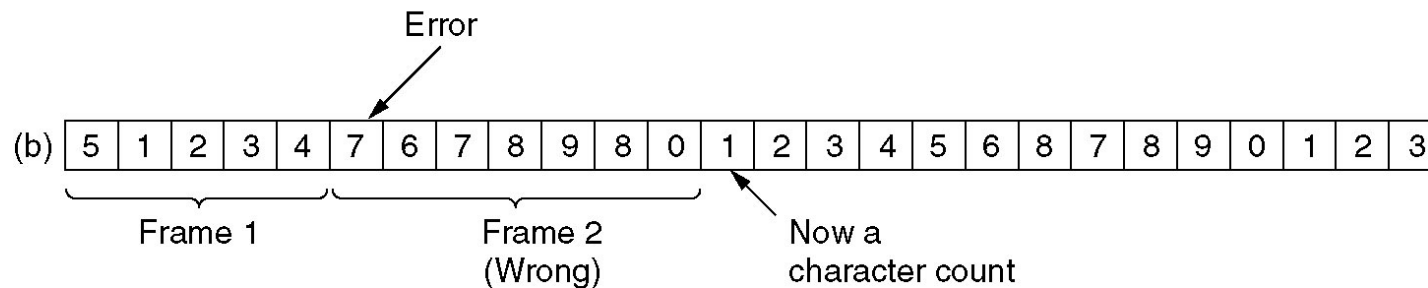
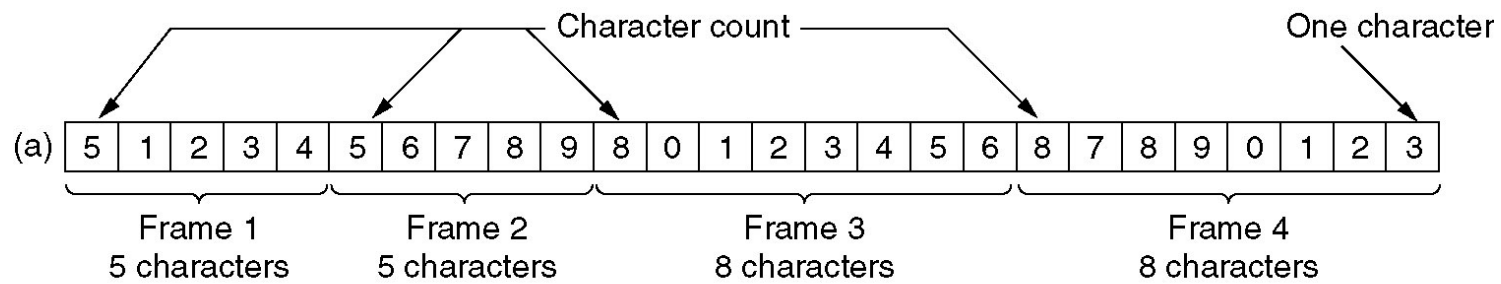
# Suddivisione di un flusso di caratteri in trame

Per suddividere un flusso di caratteri in *trame* è troppo rischioso basarsi sul timing. Ci sono diversi metodi per marcare l'inizio e la fine di una trame:

- Conto dei caratteri
- Flag byte con “byte stuffing”
- Flag di inizio e fine con “bit stuffing”
- Violazioni di codice nello strato fisico

## Suddivisione di un flusso di caratteri in frame

- Flusso di caratteri. (a) Senza errori.  
(b) Con un errore.

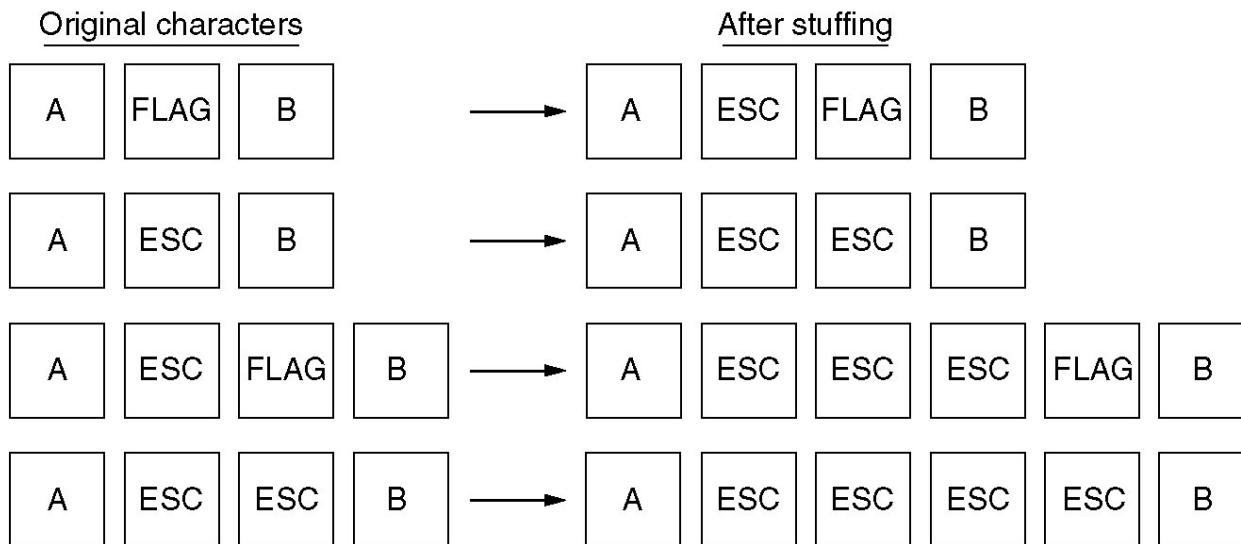


## Suddivisione di un flusso di caratteri in trame (2)

- (a) Frame delimitato da “flag bytes”.
- (b) Quattro esempi di sequenze di byte prima e dopo lo *stuffing*



(a)



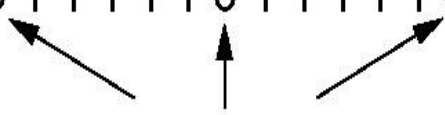
(b)



## Suddivisione di un flusso di caratteri in trame (3)

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0



Stuffed bits

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

- **Bit stuffing**
  - (a) I dati originali
  - (b) I dati come appaiono in linea
  - (c) I dati come sono memorizzati dal ricevitore dopo aver tolto i bit di stuffing.

# Codici a correzione di errore

Due strategie:

- Includere sufficiente informazione ridondante per permettere al ricevitore di dedurre quali dati sono stati trasmessi anche in presenza di errori (**codici a correzione di errore – error correcting codes**)
- Includere sufficiente ridondanza per permettere al ricevitore di capire che si è verificato un errore, senza però capire quale errore è, richiedendo una ritrasmissione (**codici a rivelazione di errore – error detecting codes**)

*Quale tecnica usare dipende dalle prestazioni del livello fisico.*

**m** dati + **r** ridonanza ->  **$N=m+r$**  lunghezza della parola di codice

Numero di posizioni di bit in cui due parole di codice differiscono: **distanza di Hamming** -> ci vogliono **d** errori singoli per modificare una parola in un'altra

La distanza di Hamming di un codice è la minima tra le distanze di Hamming delle singole coppie di parole.

Per rivelare **d** errori è necessario un codice a distanza  **$d+1$** . Per correggere **d** errori è necessario un codice di distanza  **$2d+1$**

Un semplice esempio di codice a correzione di errore è quello in cui è appeso un singolo **bit di parità** (scelto in modo che il numero di bit totale della parola sia pari).

## Codici a correzione di errore (2)

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

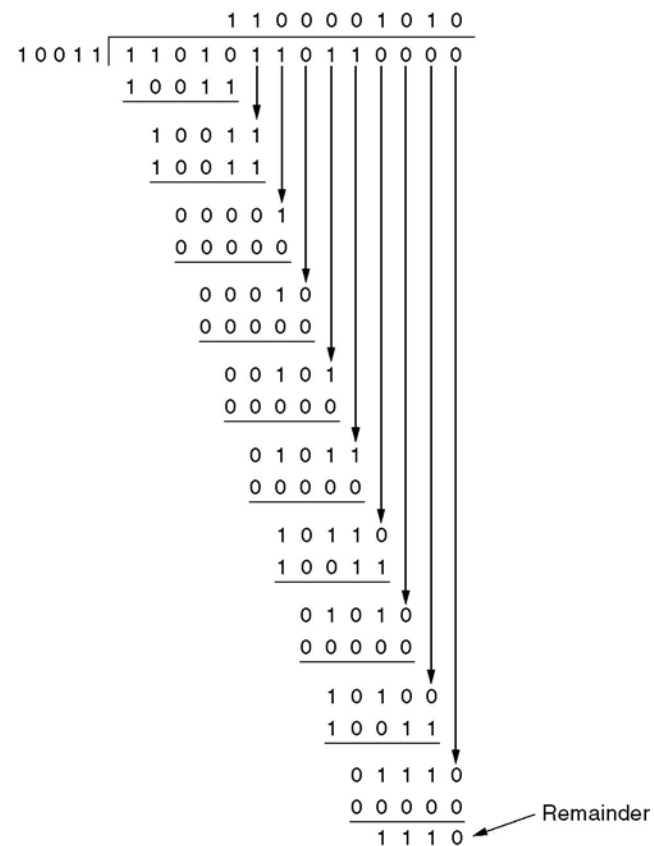
Order of bit transmission

## Codici a rivelazione di errore

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



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

## Codici a rivelazione di errore (2)

### Codici polinomiali o CRC (Cyclic Redundancy Check).

Un frame di  $k$  bit e' visto come un polinomio di grado  $k-1$  con  $k$  termini che vanno da  $x^{k-1}$  a  $x^0$ .

Esempio: 110001  $\rightarrow x^5 + x^4 + x^0$ .

10011011	00110011	11110000	01010101
+11001010	+11001101	- 10100110	- 10101111
-----	-----	-----	-----
01010001	11111110	01010110	11111010

Le divisioni vengono fatte come nel caso binario ad eccezione del fatto che la sottrazione è fatta modulo 2.

Polinomio generatore  $G(x)$  di grado  $r$  condiviso da trasmettitore e ricevitore.

### Algoritmo per calcolare la “checksum”:

- Appendere  $r$  bit “0” all'estremo destro del frame  $\rightarrow x^r M(x)$
- Dividere  $x^r M(x)$  per  $G(x)$  usando la divisione modulo 2
- Sottrarre il resto (che e' sempre di  $r$  o meno bit) da  $x^r M(x)$  usando la sottrazione modulo 2. Il risultato è la trama “checksummed” che deve essere trasmessa  $T(X)$ .

## Codici a rivelazione di errore (3)

Stringa ricevuta:  $T(x) + E(x)$       *ogni “1” in  $E(x)$  corrisponde a un bit errato*

*un singolo burst di errori e' caratterizzato da un “1” iniziale un insieme di “0” e “1” e da un “1” finale*

Il ricevitore esegue  $[T(x) + E(x)] / G(x)$ ; il resto di  $T(x)/G(x) = 0 \rightarrow E(x)/G(x)$

- gli errori che corrispondono a polinomi che contengono  $G(x)$  come fattore, verranno saltati mentre tutti gli altri verranno rivelati.*
- Se c'e' stato un singolo errore:  $E(x) = x^i \rightarrow$  se  $G(x)$  contiene almeno due termini non dividerà mai esattamente  $E(x) \rightarrow$  il singolo errore viene rivelato*
- Se ci sono stati due errori  $\rightarrow E(x) = x^i + x^j = x^i (x^{i-j} + 1) \rightarrow$  se prendiamo  $G(x)$  non divisibile per  $x$  e' sufficiente che  $G(x)$  non divida  $x^k + 1$  per ogni  $k$  fino al valore massimo di  $i-j$  per rilevare anche tutti gli errori doppi:  
ad esempio:  $x^{15} + x^{14} + 1$  non divide  $x^k + 1$  almeno fino a  $k=32768$ .*

Un codice polinomiale con  $r$  bit di check rivela tutti i burst di errore di lunghezza  $\leq r$ .  
Se la lunghezza del burst è  $r+1$  la probabilità che la trama sbagliata venga accettata è  $(0.5)^{r-1}$



## Codici a rivelazione di errore (4)

Alcuni polinomi sono diventati standard internazionali.

Esempio in IEEE 802:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$$

Rivela:

- tutti i burst di lunghezza 32 o meno
- tutti i burst che danneggiano un numero dispari di bit



## Protocollo stop-and-wait

- Il trasmettitore attende un ack esplicito a valle di ogni trasmissione
- Non richiede numerazione delle trasmissioni
- Utilizza il canale full-duplex come fosse half-duplex (inefficiente)



# Protocollo S&W in assenza di errore

Parametri riferiti a un generico collegamento A-B:

$T_f$  : tempo di trasmissione di una trama (s)  
 $T_a$  : tempo di trasmissione di un riscontro (s)  
 $T_p$  : tempo di elaborazione di un'unità informativa (s)  
 $\tau$  : tempo di propagazione (s)  
 $C$  : frequenza di cifra sul collegamento (bit/s)  
 $L_f$  : lunghezza di una trama (bit)  
 $L_a$  : lunghezza di un riscontro (bit)  
 $d$  : estensione del collegamento (m)  
 $v$  : velocità di propagazione sul collegamento (m/s)

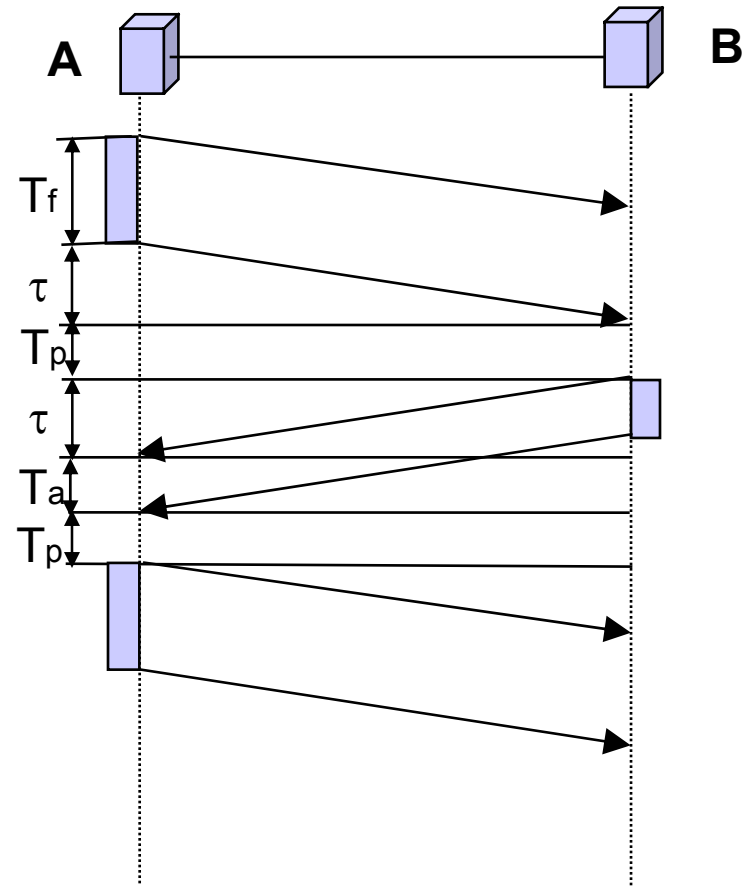
Tempo di propagazione normalizzato:

$$a = \frac{\tau}{T_f} = \frac{d/v}{L_f/c} = \frac{dc}{L_f v}$$

Efficienza  $\eta$ : quota parte di tempo in cui il collegamento da A a B e' impegnato in trasmissioni.

$$\eta = \frac{T_f}{T_f + T_a + 2T_p + 2\tau} \cong \frac{T_f}{T_f + 2\tau} \cong \frac{1}{1 + 2a}$$

A parità di frequenza di cifra l'efficienza aumenta aumentando la lunghezza di trama



## Protocollo S&W in presenza di errori

- Positive Acknowledge with Retransmission
- Basta un contatore di un solo bit, per distinguere un messaggio da quelli a lui adiacenti
- Ogni messaggio ha il suo riscontro positivo
- E' delicato dimensionare lungo il time-out

$$T > T_a + 2T_p + 2\tau \approx 2\tau$$

- se troppo breve posso incorrere nell'errore di ritrasmettere mentre l'ack è ancora in viaggio; tale riscontro verrebbe interpretato come quello alla seconda trasmissione, si procederebbe con un nuovo messaggio che potrebbe scomparire senza che nessuno se ne accorga



## sui canali full-duplex...

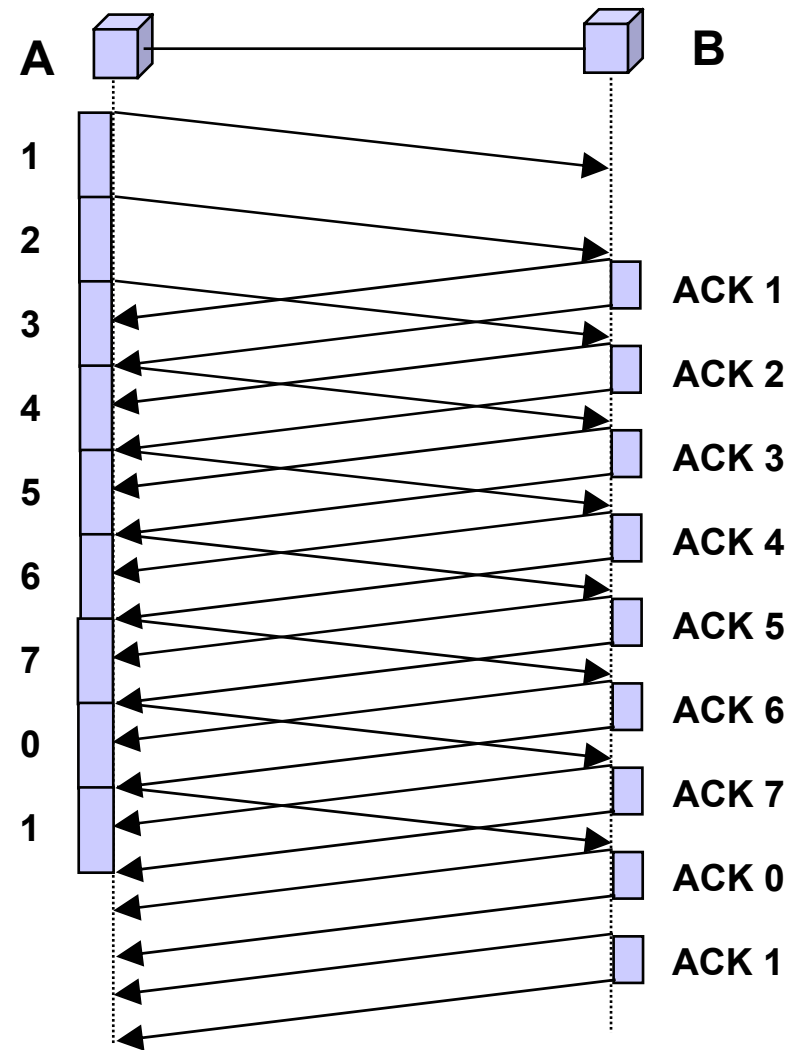
- Ack può essere inserito in una trama di dati che viaggia al contrario
  - tecnica di piggybacking
- Quanto è necessario aspettare?
  - traffico imprevedibile → serve un time-out (millisec.) che forza ack a partire comunque

# Perchè spedire più trame?

Il meccanismo di numerazione delle trame deve fare sì che tutte le trame inviate ma non ancora riscontrate abbiano un identificativo distinto.

La numerazione è di tipo ciclico modulo-N con  $N=2^b$ : se  $b=3 \rightarrow 0,1,2,3,4,5,6,7,0,1,2,\dots$

Spedire più trame consecutive permette di sfruttare meglio la capacità della linea, ma introduce delle complicazioni in caso di errori su una singola trama. E bisogna numerare anche i riscontri.





# Controllo di flusso a finestra

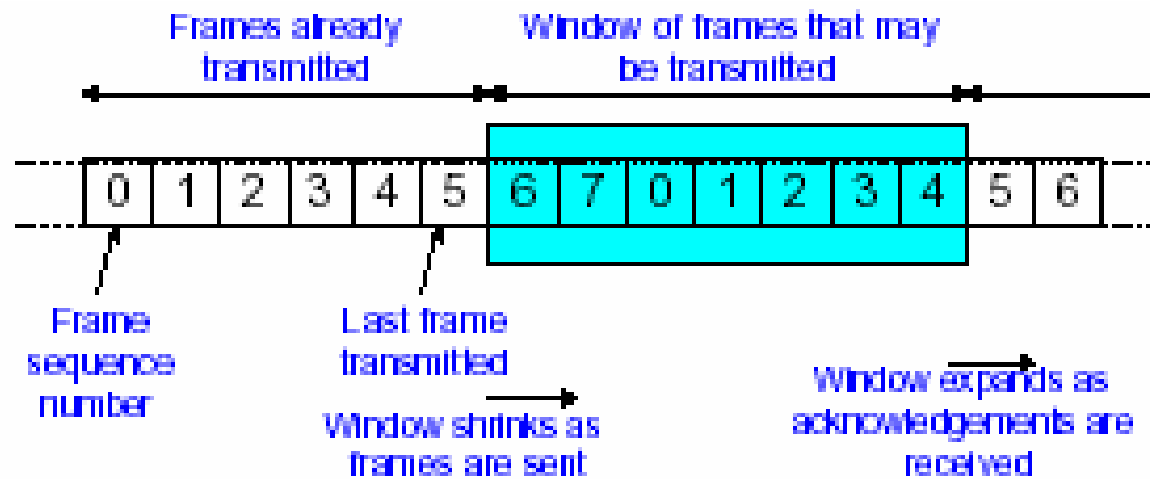
- Evita i problemi di saturazione di buffer in ricezione
- Permette la trasmissione di trame multiple
- Assegna ad ogni trama un numero di sequenza  
contato a modulo-k (con un range  $[0, 1, \dots, 2^k - 1]$ )

## Controllo di flusso a finestra (2)

- Finestra di trasmissione
  - Insieme dei numeri di sequenza progressivi che il mittente può spedire prima di ricevere un ack
- Finestra di ricezione
  - Insieme delle trame consecutive che il ricevente può accettare nel suo buffer di ingresso

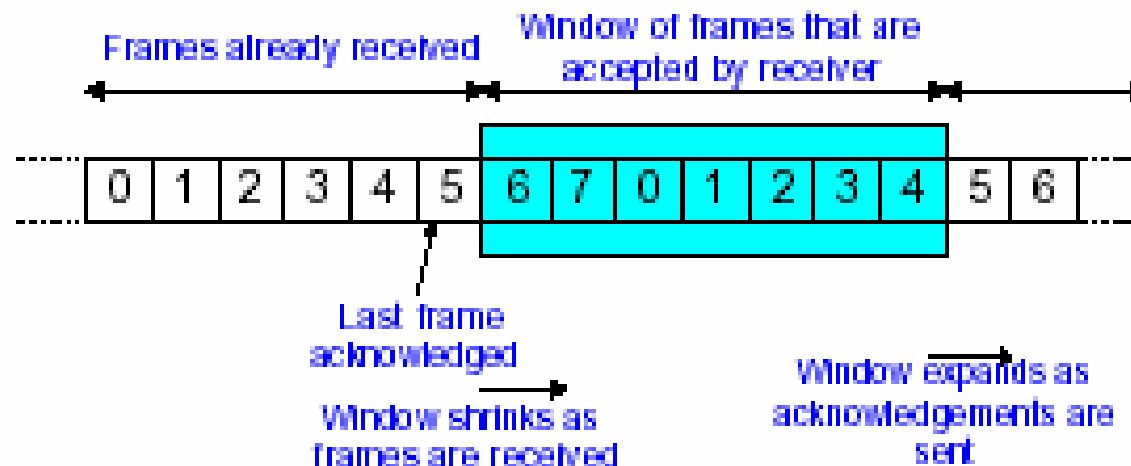
# Controllo di flusso a finestra (2)

- Finestra di trasmissione
  - Ad ogni istante il trasmettitore può inviare frame con numero di sequenza all'interno di un certo *range* (la finestra di trasmissione)



# Controllo di flusso a finestra (3)

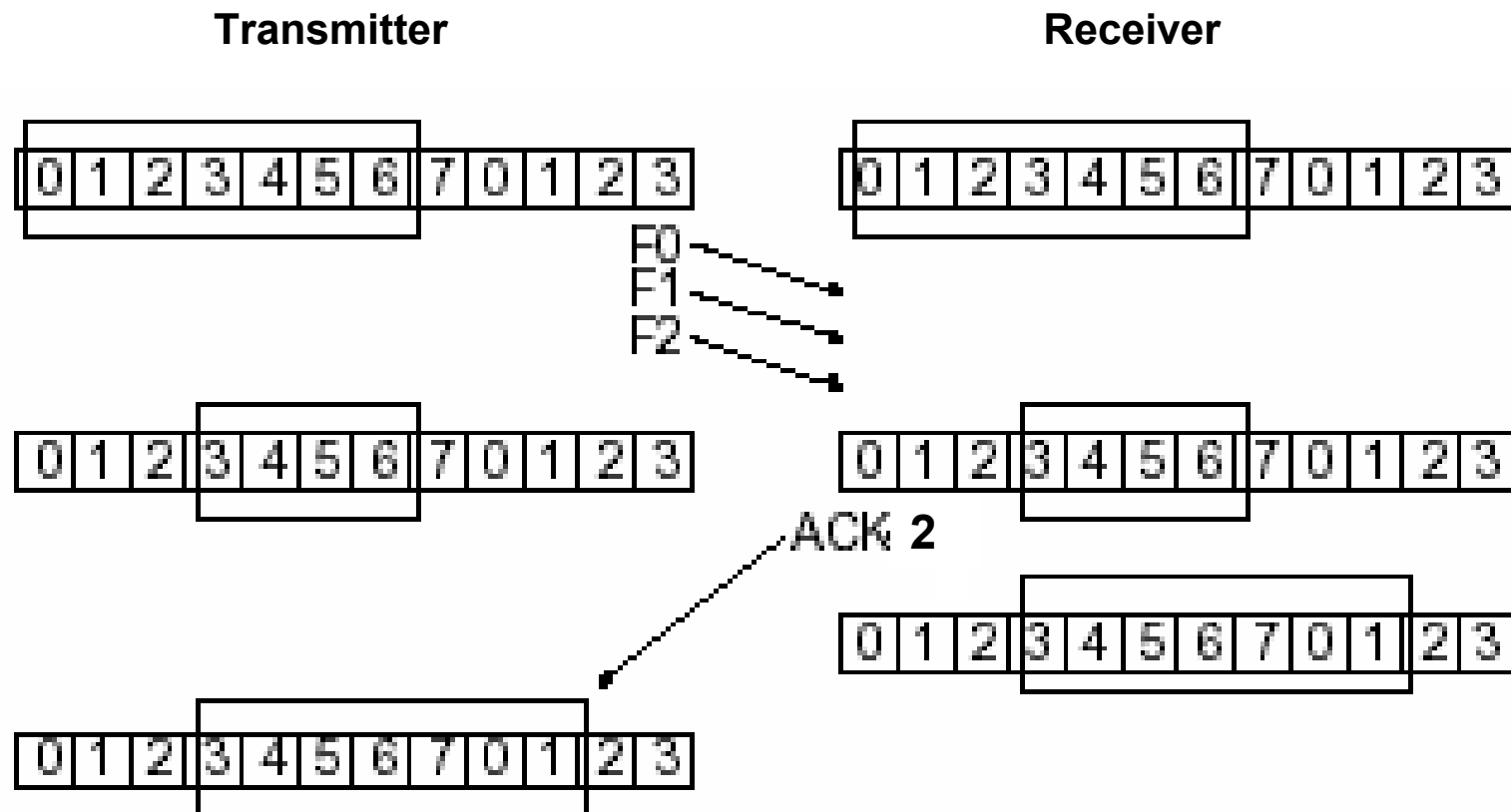
- Finestra di ricezione
  - Il ricevitore mantiene una finestra di ricezione corrispondente ai numeri di sequenza delle frame che sono accettate





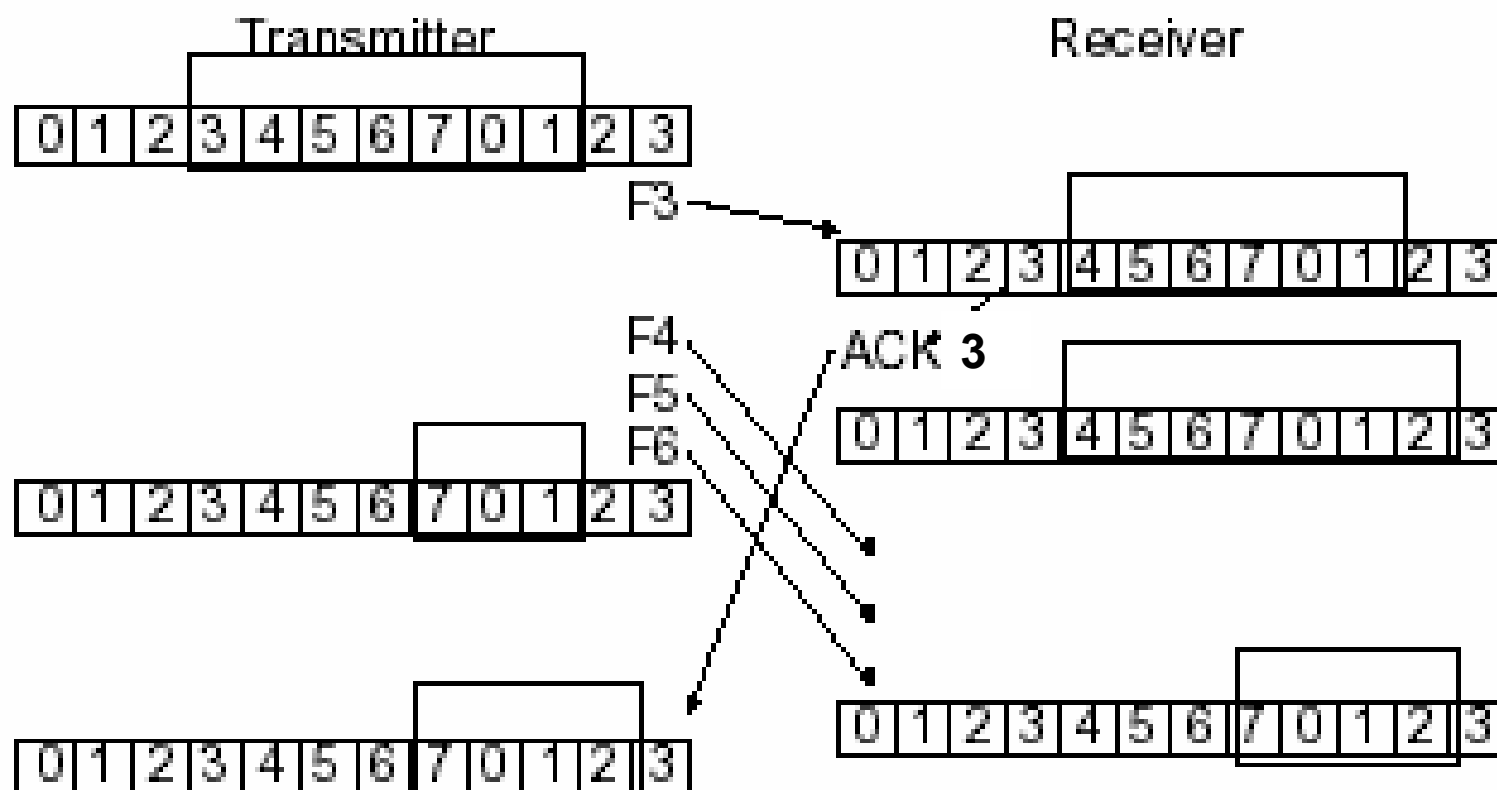
# Controllo di flusso a finestra (4)

- Esempio



## Controllo di flusso a finestra (5)

- Continuazione dell'esempio



# La finestra di scorrimento

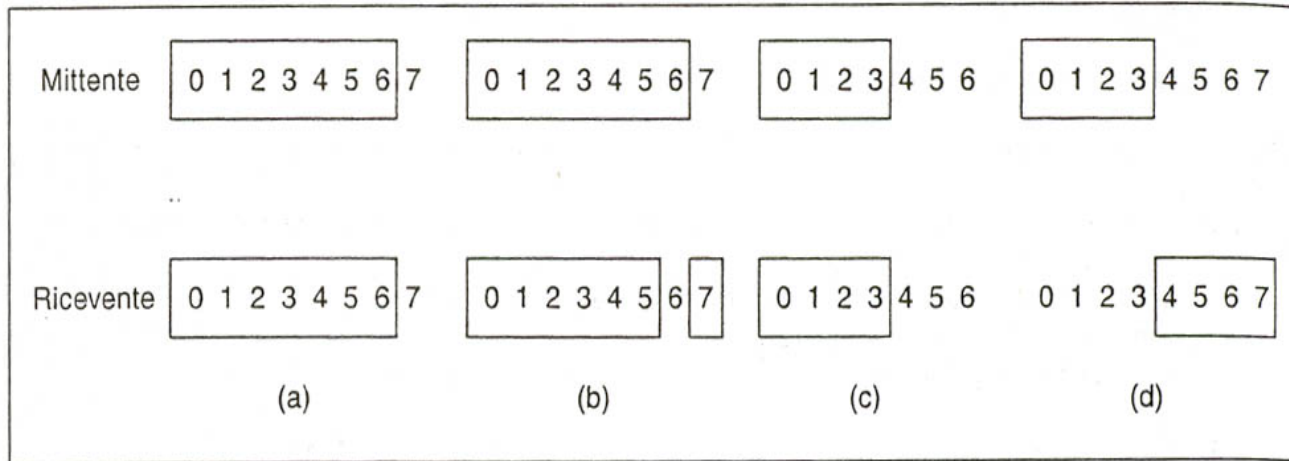


Fig. 3-19 (a) Situazione iniziale con una finestra di dimensione 7. (b) Dopo la spedizione e ricezione ma non il riscontro di 7 pacchetti. (c) Situazione iniziale con una finestra di dimensione 4. (d) Dopo la spedizione e ricezione ma non il riscontro di 4 pacchetti.

È opportuno che la finestra a scorrimento non sia superiore alla metà del massimo numero di sequenza disponibile, altrimenti la perdita multipla di riscontri può portare il ricevitore ad accettare trame ritrasmesse come trame nuove.

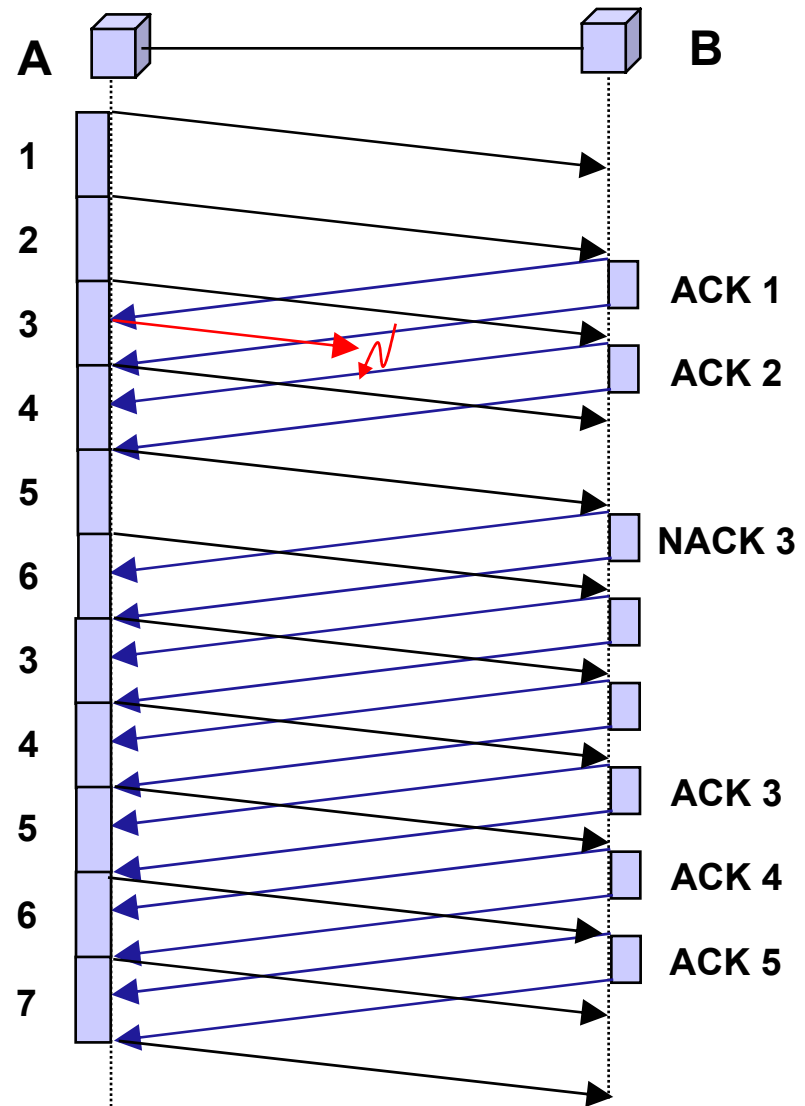
## Es: finestra di 7 trame

- trasmetto trame da 0 a 6 → ricevuti corretti, next=7
- invio ack da 0 a 6, ma ack 0 viene perso
- ritrasmette trama 0 dopo time-out,
- sta nella window, è corretto → memorizzo e invio ack6
- riprende sequenza trasmissione: 7, 0, 1, 2... ma la trama 0 risulta già arrivata, ma è sbagliato

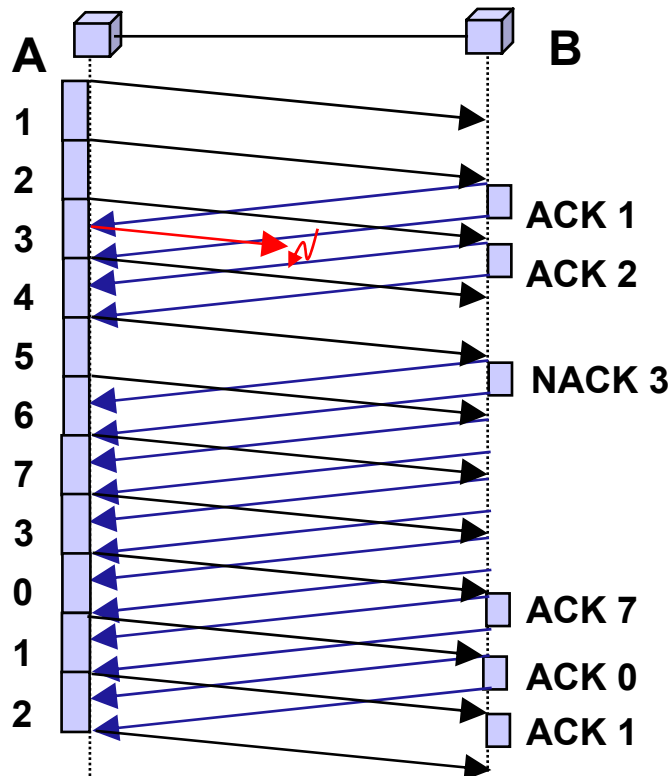
# Protocollo Go-back-n (GBN)

**NACK  $i$**  indica non solo la mancata ricezione da parte della stazione della trama  **$i$ -esima**, ma anche la corretta ricezione di tutte le trame fino alla numero  **$(i - 1) \bmod N$** .

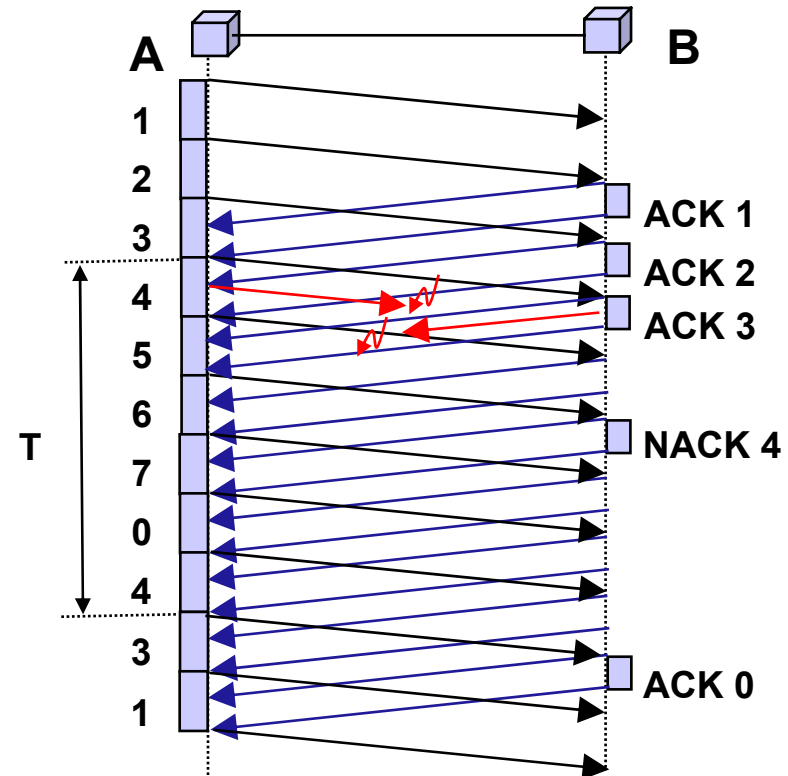
Dopo l'invio di un NACK, la stazione B scarta tutte le trame ricevute con numeri crescenti fino alla corretta ricezione della trama  **$i$ -esima**.



# Protocollo Selective Repeat (SR)



a) ritrasmissione per riscontro negativo



b) ritrasmissione per riscontro negativo e per time-out

Il riscontro negativo **NACK i** indica solamente che la trama **i-esima** non è stata ricevuta dalla stazione. Come nel caso del protocollo Go-back-n, il riscontro positivo **ACK i** indica la corretta ricezione delle trame fino alla **i-esima** compresa

# Efficienza dei protocolli GBN e SR

**Ipotesi:**  $T_a = T_p \cong 0$  ;  
il buffer del trasmettitore è sempre non vuoto

$$\eta = \frac{T_f}{N_s T_f} = \frac{1}{N_s}$$



$$\eta = \begin{cases} 1-P & \text{SR} \\ \frac{1-P}{1+2aP} & \text{GBN} \end{cases}$$

$T_f$  : tempo di trasmissione di una trama (s)  
 $T_a$  : tempo di trasmissione di un riscontro (s)  
 $T_p$  : tempo di elaborazione di un'unità informativa (s)  
 $N_s$  : numero medio di tentativi di trasmissione di una trama  
 $P$  : probabilità di errore nella trasmissione di una trama

$$a = \frac{\tau}{T_f} = \frac{d/v}{L_f/c} = \frac{dc}{L_f v},$$

tempo di propagazione normalizzato

# Il protocollo HDLC (High Level Data Link Control)

## Tre tipi di stazioni:

- **stazione primaria**, responsabile del controllo del collegamento attuato per mezzo di trame denominate **comandi**;
- **stazione secondaria**, che opera sotto il controllo di una stazione primaria ed emette trame che vengono perciò chiamate **risposte**;
- **stazione combinata**, che combina le funzioni di stazione primaria e secondaria, e quindi emette sia **comandi** sia **risposte**.

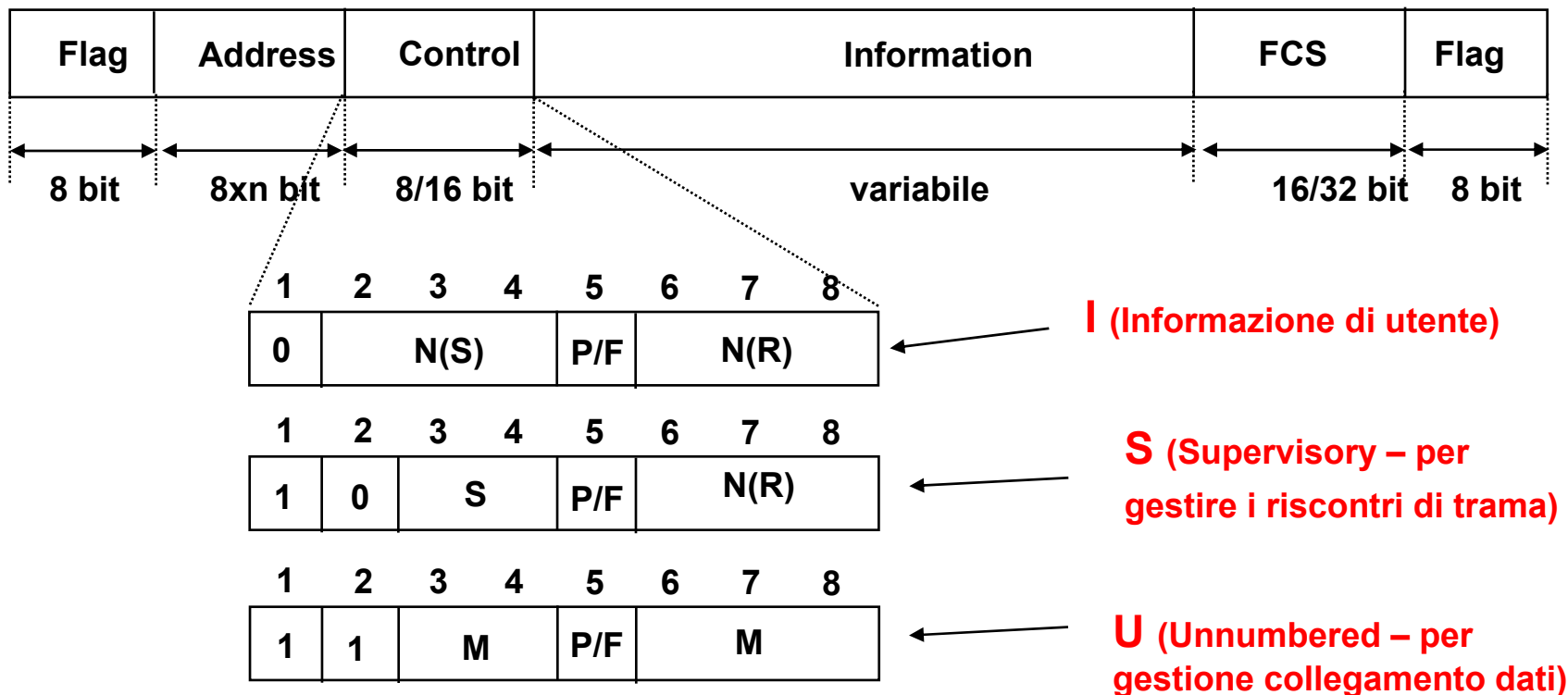
## Due tipi di configurazioni di linea:

- **configurazione sbilanciata**: una **stazione primaria** e una o più **stazioni secondarie**;
- **configurazione bilanciata**: due **stazioni combinate**.

## Tre modalità di trasferimento delle trame:

- **Asynchronous Balanced Mode (ABM)**, per **configurazioni bilanciate**: ognuna delle due stazioni combinate può iniziare a trasmettere senza dover attendere alcuna autorizzazione dall'altra stazione;
- **Normal Response Mode (NRM)**, per **configurazioni sbilanciate**: solo la stazione primaria può iniziare a trasmettere sul collegamento, mentre una secondaria può solamente rispondere a un comando ricevuto dalla primaria ;
- **Asynchronous Response Mode**, per **configurazioni sbilanciate**: a differenza della modalità NRM, una stazione secondaria può iniziare a trasmettere senza esplicita autorizzazione da parte della stazione primaria.

# Il protocollo HDLC – Struttura della trama



- **P (Poll)** significa che il contenuto del messaggio è una richiesta
- **F (Final)** significa che il contenuto del messaggio è una risposta
- **M** e **S** sono codici di funzioni particolari per le trame **U** e **S** rispettivamente



# Trasparenza dei dati

01111110	10111111	10111111	11111010	00111111	01111110	(a)	
01111110	10111111	01101111	01111110	10100011	11110101	11111110	(b)

# Rivelazione di errore nell'HDLC

Il codice di rivelazione di errore è contenuto nel campo FCS e ha lunghezza  $k$  che vale 8 o 16: esso è rappresentato dai coefficienti del polinomio  $R(X)$ :

$$\frac{P(X) \cdot X^k}{D(X)} = Q(X) + \frac{R(X)}{D(X)}$$

$$D(X) = X^{16} + X^{15} + X^{12} + 1$$

La trama trasmessa, così costruita è:

$$P'(X) = P(X) \cdot X^k + R(X)$$

In ricezione, se

$$\frac{P'(X)}{D(X)} \text{ da' resto } \neq 0$$

**Si è verificato almeno un errore!**

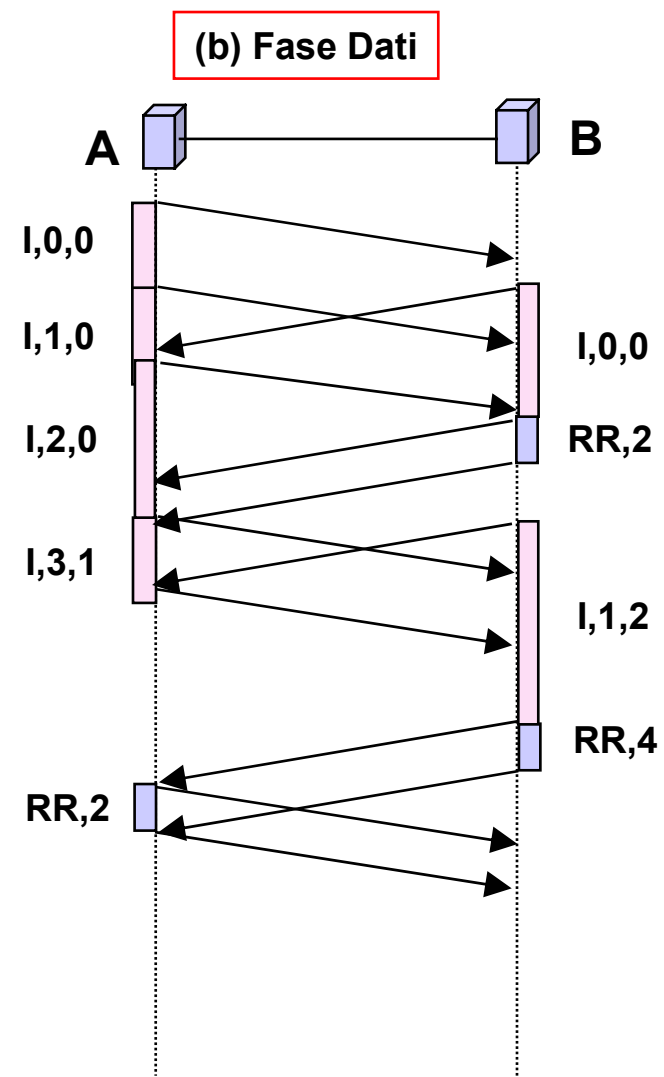
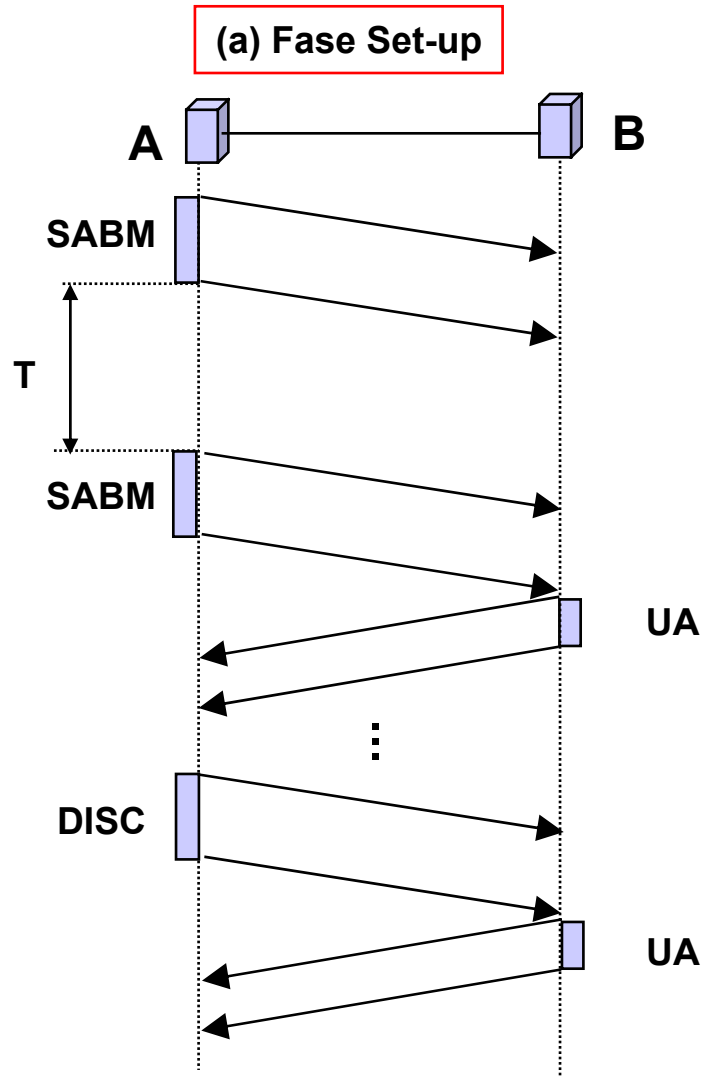
- Errori su un bit singolo
- Errori su due bit
- Errori su un numero dispari di bit
- Errori a *burst* di lunghezza inferiore al grado di  $D(X)$  meno 1

# Comandi e risposte dell'HDLC

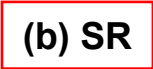
**Tabella 3.1** Trame definite nel protocollo HDLC.

Tipo	Trama	Significato	Comando	Risposta
I	I	Information	x	x
S	RR	Receive ready	x	x
	RNR	Receive not ready	x	x
	REJ	Reject	x	x
	SREJ	Selective reject	x	x
U	SIM	Set initialization mode	x	
	SNRM/SNRME	Set normal response mode/SNRM extended	x	
	SARM/SARME	Set asynchronous response mode/SARM extended	x	
	SABM/SABME	Set asynchronous balanced mode/SABM extended	x	
	UA	Unnumbered acknowledgment		x
	DISC	Disconnect	x	
	RD	Request disconnect		x
	DM	Disconnect mode		x
	RSET	Reset	x	
	FRMR	Frame reject		x
	UI	Unnumbered information	x	x
	UP	Unnumbered poll	x	
	RIM	Request information mode		x
	XID	Exchange ID	x	x

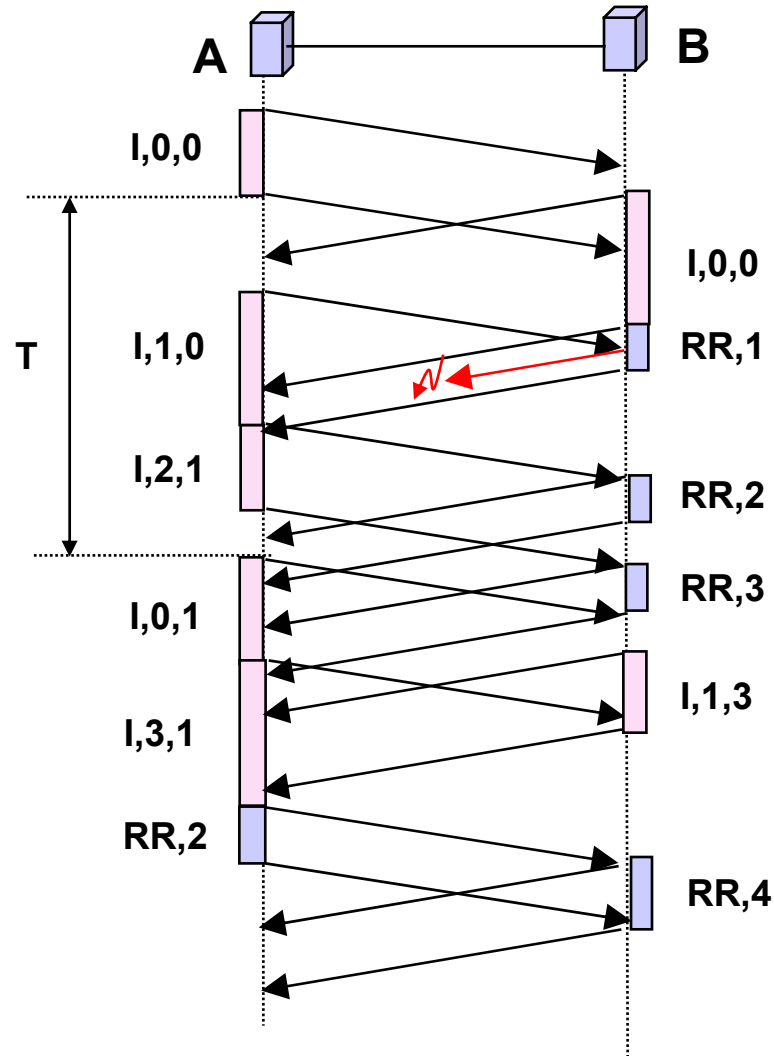
## Operazioni con protocollo HDLC – ABM (assenza errori)



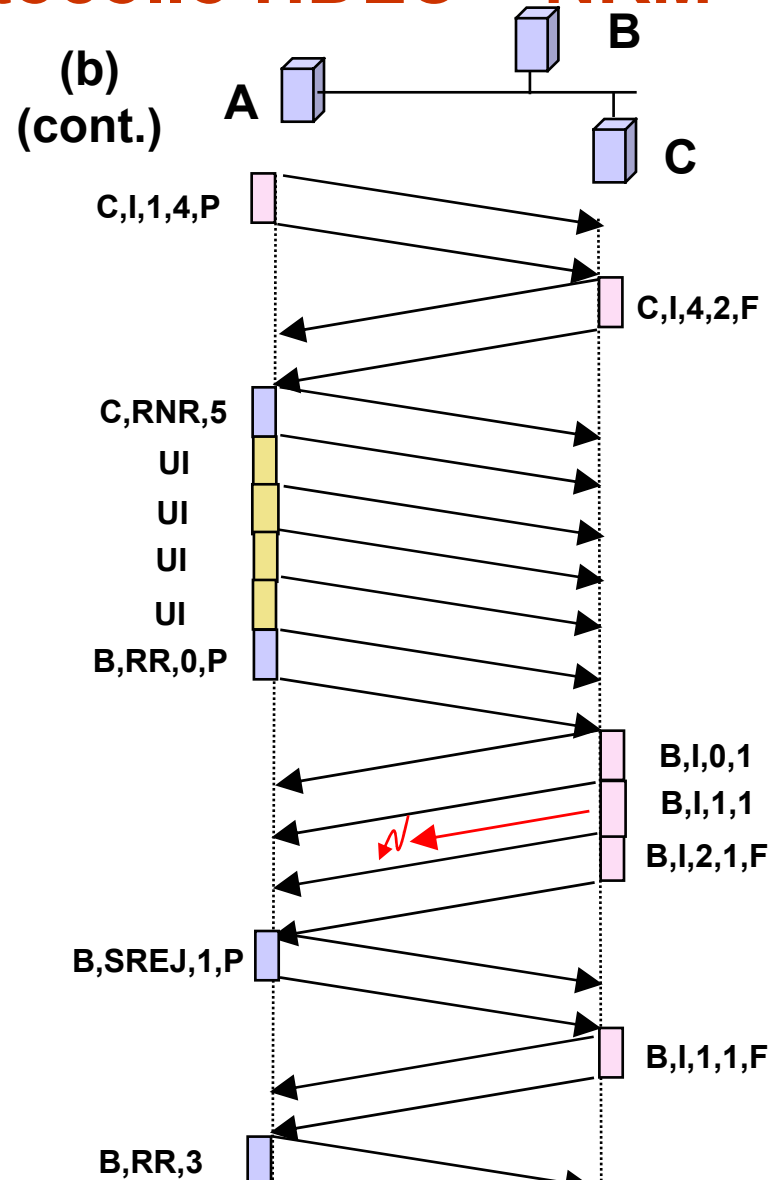
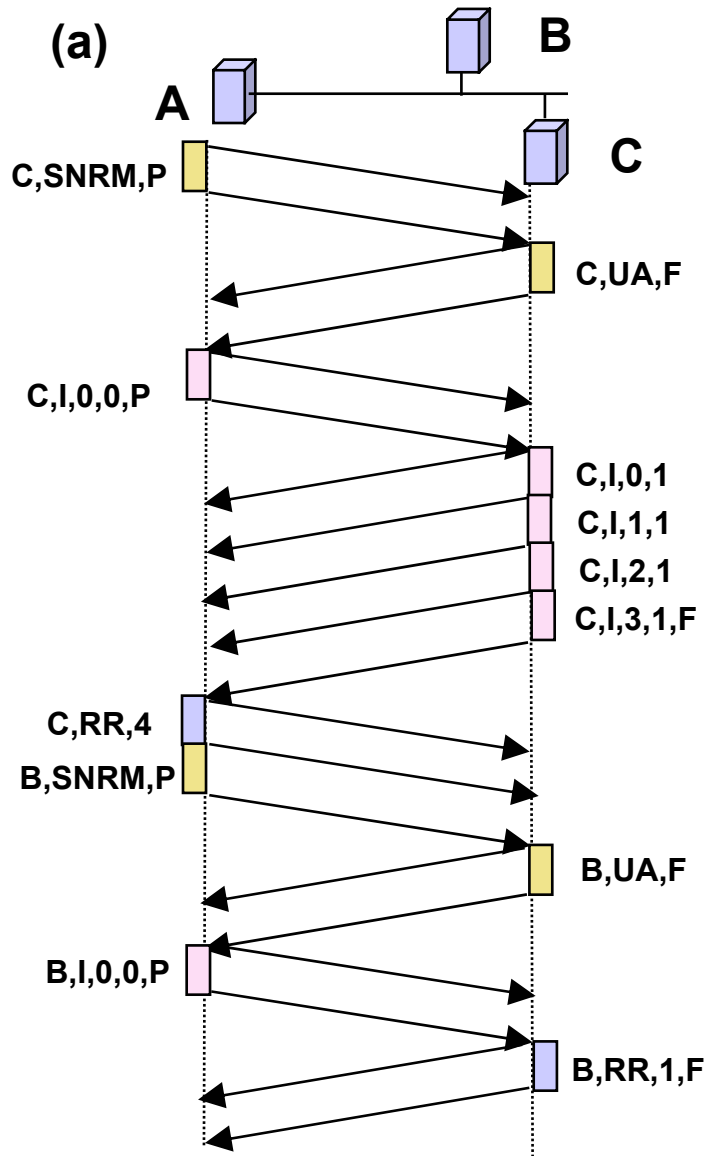
1. *Journal of the American Medical Association*, 2000; 283: 2689-2695.



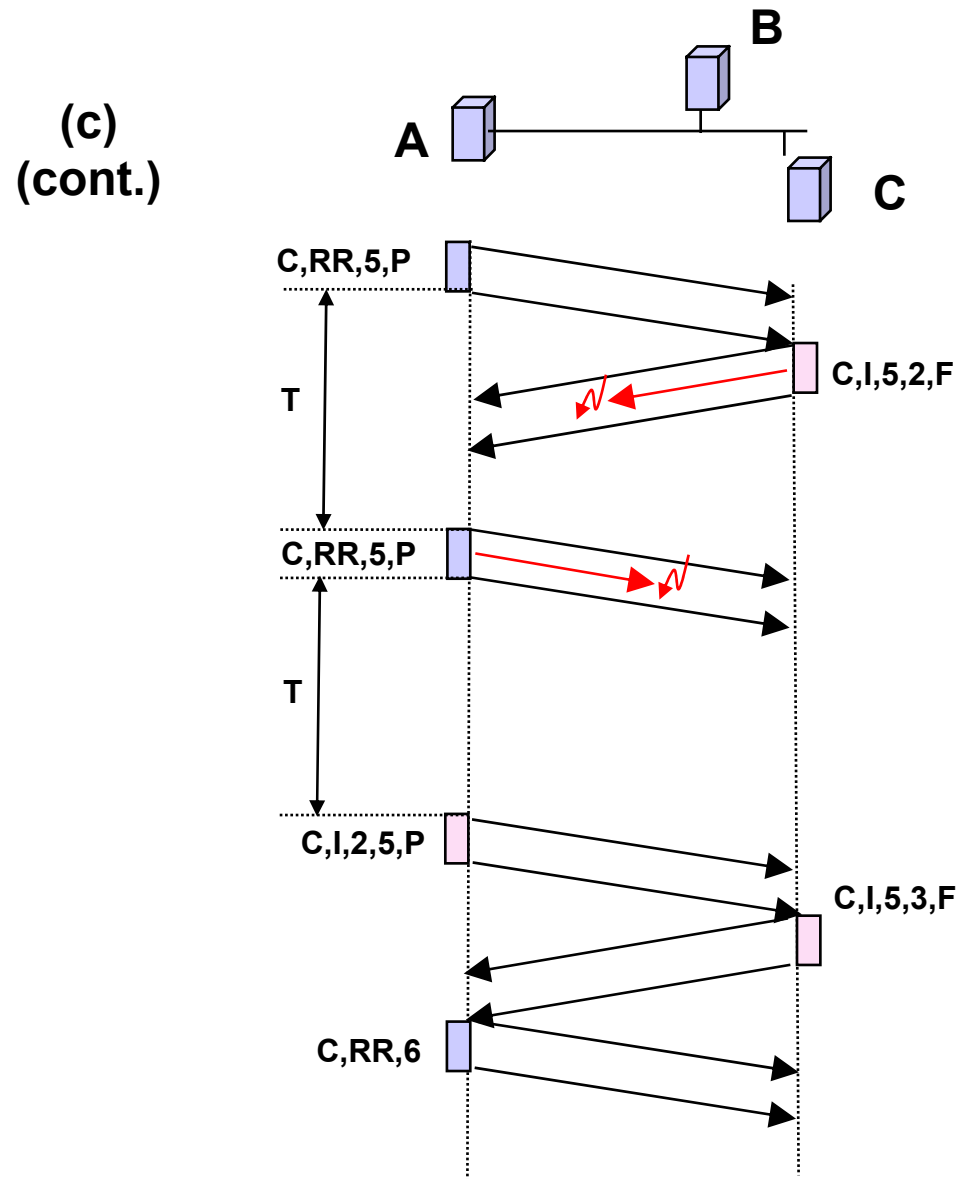
# Operazioni con protocollo HDLC – ABM (con ritrasmissione per time-out)



# Operazioni con protocollo HDLC – NRM



## Operazioni con protocollo HDLC – NRM (2)



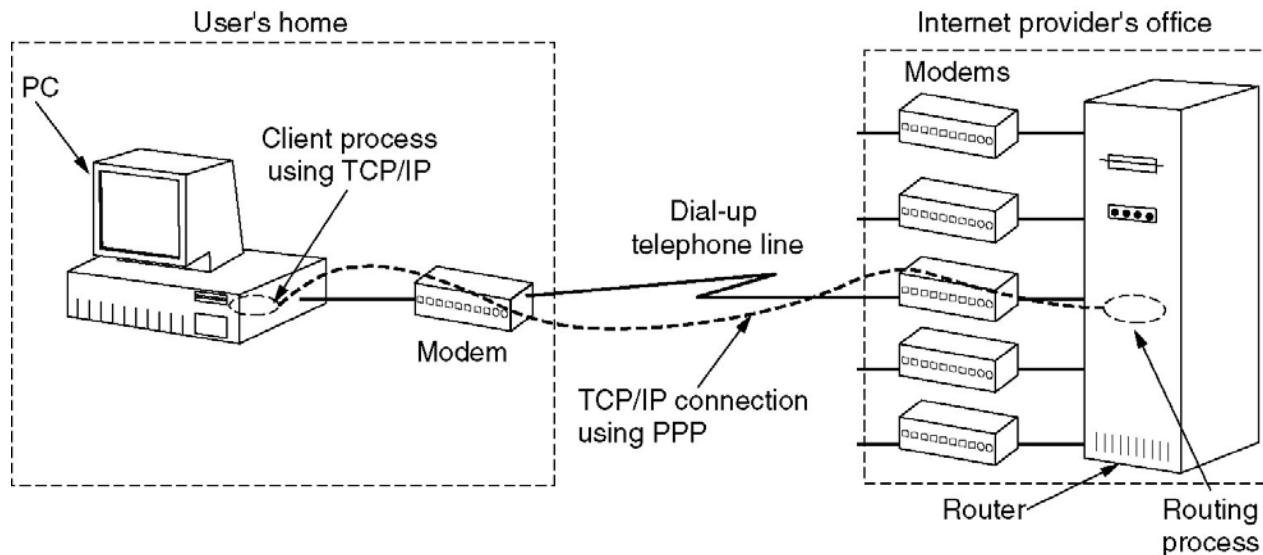


# Parametri del protocollo HDLC

- $T_1$ : time-out per ri-trasmissione in assenza di riscontro;
- $T_2$ : massimo intervallo di tempo ammesso per elaborare una trama ricevuta e inviare il relativo riscontro;
- $N_1$ : massimo numero di bit del campo informativo;
- $N_2$ : massimo numero di ri-trasmissioni ammesse per ogni trama, prima di generare una segnalazione di allarme;
- $W_s$  e  $W_r$ : ampiezza delle finestre in trasmissione e ricezione.

$$T_2 \leq T_1 - 2\tau - 2T_p - T_a$$

## Il Protocollo PPP (Point-to-Point Protocol)



**Il Protocollo PPP è utilizzato nei collegamenti su linea telefonica tra host di utenza residenziale e Internet Service Provider, oltre che su connessioni SDH o ISDN**

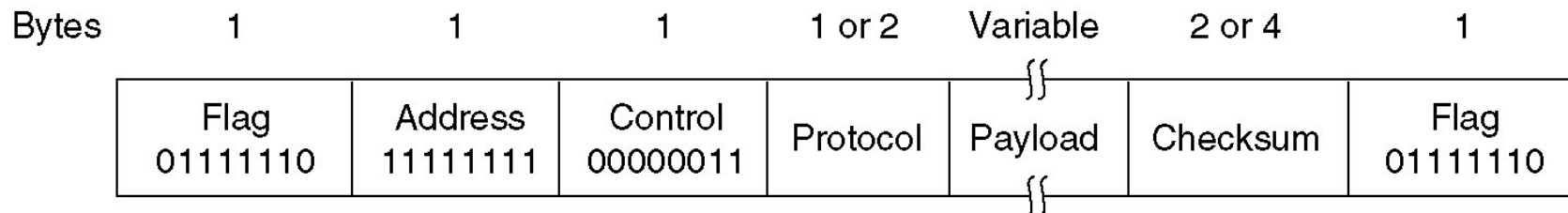


## Il Protocollo PPP (Point-to-Point Protocol) (2)

Il Protocollo PPP fornisce tre prestazioni fondamentali:

- Un metodo di costruzione delle trame che ne definisce in modo certo l'inizio e la fine. Gestisce anche il rilevamento degli errori.
- Un protocollo di controllo per stabilire i collegamenti, testarli, negoziare le opzioni (es. indirizzo IP dinamico), e chiudere in modo "soft" i collegamenti. Il protocollo si chiama LCP (Link Control Protocol)
- Un modo per negoziare le opzioni dello strato di rete indipendente dallo strato di rete che viene usato. Il metodo scelto è quello di avere un diverso NCP (Network Control Protocol) per ogni strato di rete supportato.

## Il Protocollo PPP (Point-to-Point Protocol) (3)



- In sintesi il **PPP** è un meccanismo di framing multi-protocollo adatto per essere usato in collegamenti con modem, con linee seriali HDLC, SDH e altri strati fisici.
- Il campo “protocol” dice che tipo di pacchetto c’è nel campo “payload” (LCP, NCP, IP, ecc.)
- Supporta la rivelazione di errore, la negoziazione di opzioni (lunghezza max. del frame, protocollo di autenticazione, elimin. campi “Address” e “Control”), la compressione dell’header.

## Esempio semplificato di instaurazione e abbattimento di un collegamento tramite PPP

