Università di Bergamo

Dipartimento di Ingegneria dell'Informazione e Metodi Matematici

Reti di Telecomunicazione

Prof. Fabio Martignon

Università di Bergamo

Dipartimento di Ingegneria dell'Informazione e Metodi Matematici

1 - Introduzione alle Reti

Reti di Telecomunicazione

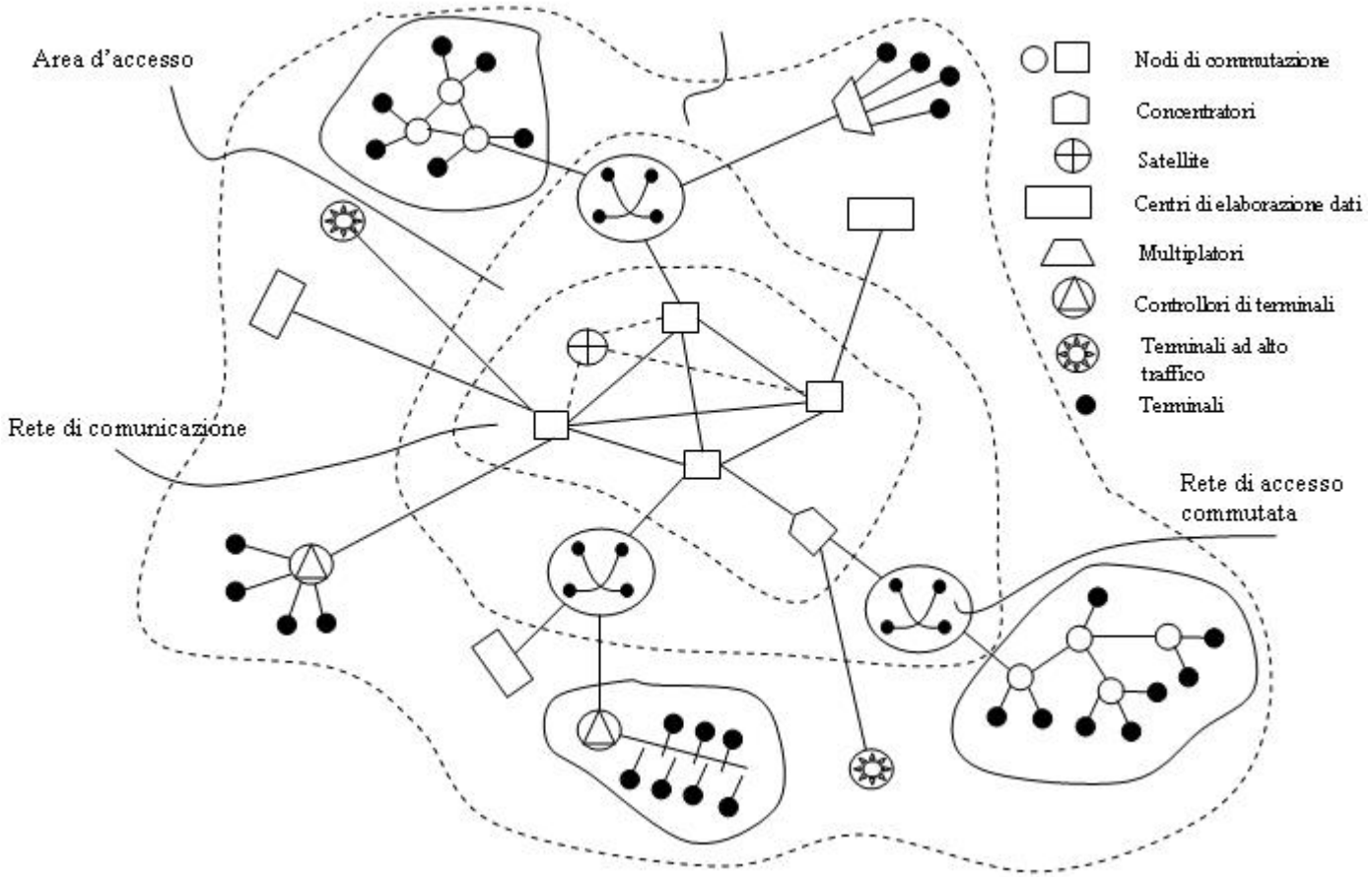
Introduzione alle Reti di Telecomunicazione

- Classificazione reti e tecniche di commutazione
- Architettura a Strati
  - Richiami sull'Architettura ISO-OSI
- Funzioni svolte dai vari livelli di rete:
  - Multiplazione
  - Recupero degli Errori
    - ✓ Tecniche ARQ:
      - ✓ Stop & Wait
      - ✓ Go-Back-N
      - ✓ Selective Repeat
  - Inoltro dei Pacchetti
    - ✓ Determinazione lunghezza trama ottima ed effetto pipelining nelle reti di telecomunicazione

Rete di Telecomunicazione: Definizione

- **Una Rete di Telecomunicazione può essere definita come un insieme di componenti, meccanismi e protocolli mediante i quali gli utenti connessi alla rete possono scambiarsi informazioni intellegibili.**

Schema Generale di una Rete Dati



Rete di Telecomunicazione: Funzioni

- Tipiche funzioni richieste ad una Rete di Telecomunicazione:
  - Fornire un cammino attraverso il quale i segnali elettrici/ottici possono essere trasmessi
  - Fornire un meccanismo che converte i bit a/da segnale elettrico/ottico
  - Fornire meccanismi per sopperire ad inefficienze che causano errori di interpretazione del segnale (per es. ARQ)
  - Fornire meccanismi per scegliere e mantenere il cammino nella rete in grado di eseguire le precedenti funzioni

Classificazione delle Reti

- **Commutate:**
  - **I segnali devono essere instradati per raggiungere la destinazione desiderata (attraverso nodi, commutatori, switch, router)**
- **Broadcast:**
  - **I segnali trasmessi da un terminale sono automaticamente sentiti da tutti i terminali connessi in rete**
- **Ibride:**
  - **Con caratteristiche a cavallo fra le due categorie (per es. una serie di LAN connesse da router)**

Tecniche di Commutazione: classificazione

- Commutazione di Circuito
- Commutazione di Messaggio
- Commutazione di Pacchetto
  - Modalità Datagram
  - Modalità Circuito Virtuale

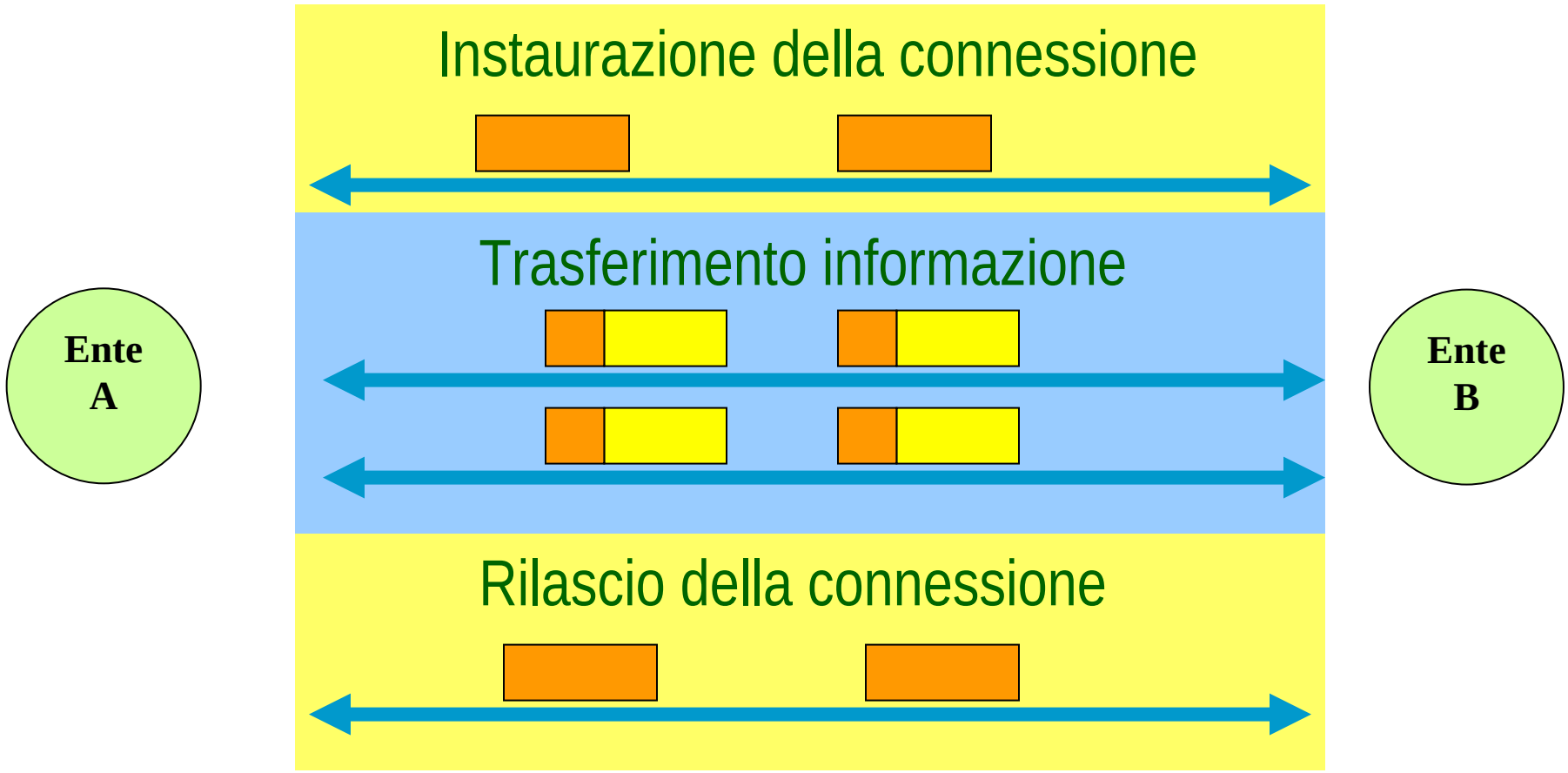
Commutazione di Circuito

- **Commutazione di Circuito (Circuit Switch):**
  - **Viene fornito un cammino dedicato (una connessione dedicata) fra i terminali in modo esclusivo per tutta la durata della comunicazione**

- **Esempio classico: *Telefonia***

Commutazione di Circuito

- **Il servizio a connessione richiede tre fasi:**



Commutazione di Messaggio

- **Commutazione di Messaggio:**
  - **Il messaggio viene trasferito da un nodo al successivo nel cammino (path) verso la destinazione.**
  - **Il messaggio è ricevuto ad ogni nodo, memorizzato ed inviato al nodo successivo (store & forward).**
    - ✓ **Il messaggio "salta" da un nodo al successivo con la possibilità di dover aspettare se il canale è occupato.**

Commutazione di Pacchetto

- **Commutazione di Pacchetto:**
  - **Come commutazione di messaggio, ma con lunghezza limitata dei messaggi. Un messaggio è suddiviso in pacchetti, ognuno indirizzato separatamente, e numerati sequenzialmente.**
- **Possiamo distinguere due modalità di trasferimento dei pacchetti:**
  - **Datagram**
  - **Circuito Virtuale**

Commutazione di Pacchetto: Modalità Datagram

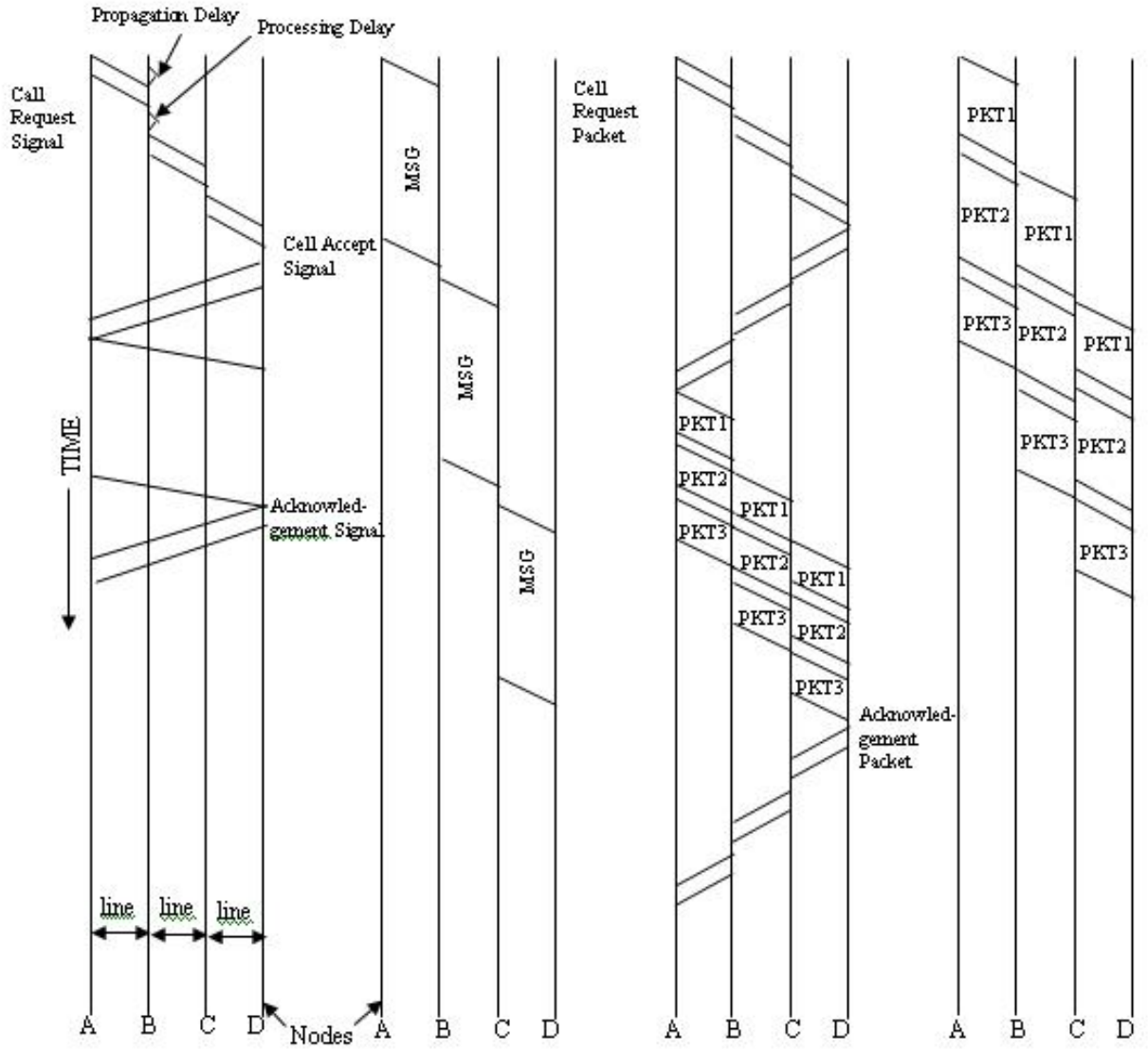
- **Modalità Datagram:**
  - **I pacchetti che compongono un messaggio sono trasmessi indipendentemente verso la propria destinazione. L'intestazione di ogni pacchetto contiene l'indirizzo del destinatario ed il numero di sequenza. Ogni nodo instrada il pacchetto verso la destinazione utilizzando il cammino "migliore".**

- **Esempio tipico: *l'invio di una lettera.***

Commutazione di pacchetto: modalità Circuito Virtuale

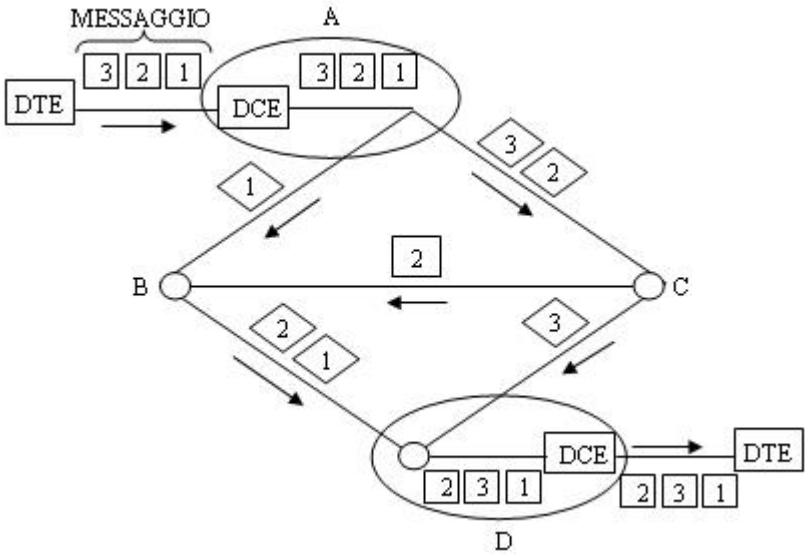
- **Modalità Circuito Virtuale:**
  - **Per iniziare la comunicazione il nodo origine invia una richiesta di connessione alla destinazione ed attende la risposta di chiamata accettata.**
  - **In questa fase si definisce il cammino da utilizzare per il trasferimento dati. Il cammino è una connessione logica: nessuna risorsa è dedicata alla singola connessione ed i pacchetti sono memorizzati ad ogni nodo.**

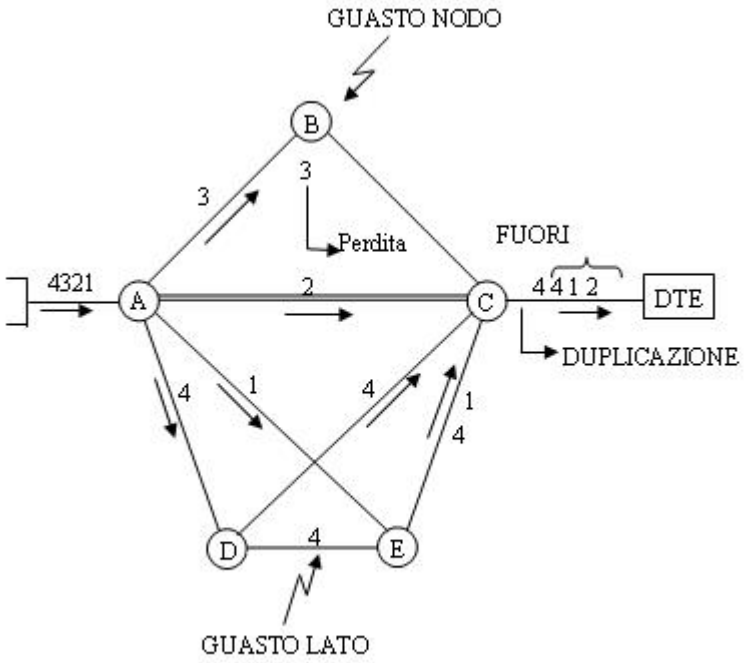
Confronto Tecniche di Commutazione



Circuito | Messaggio | Pacchetto Circuito Virtuale | Pacchetto Datagram

Commutazione Datagram

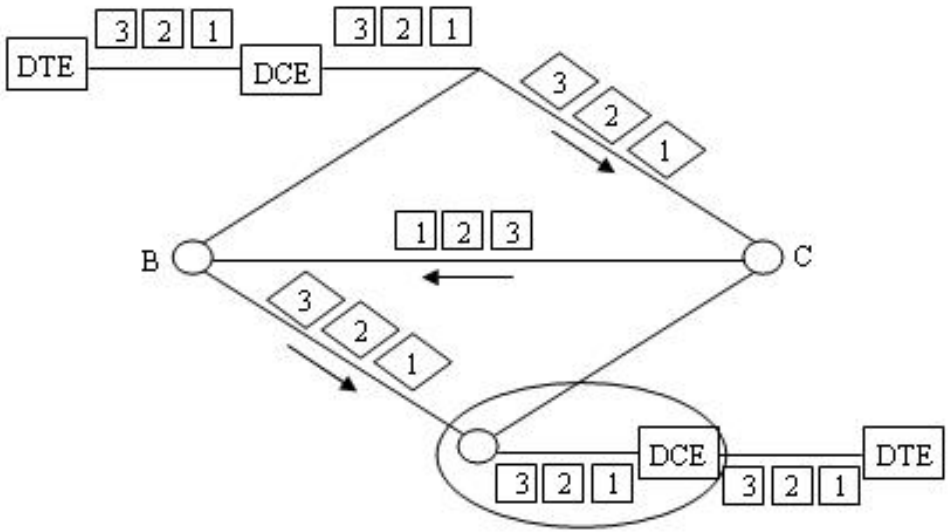




DTE = Data Terminal Equipment (ad es. un PC, o un router)
DCE = Data Communications Equipment (ad es. un modem), ovvero quei dispositivi che si frappongono per permettere la trasmissione sul canale

- **E' possibile che i pacchetti siano persi, duplicati o giungano fuori sequenza**
- **E' compito del ricevente rimediare a questi inconvenienti**

Circuito Virtuale



- In generale i pacchetti giungono ordinati e non duplicati al ricevitore
- E' sempre possibile, però, che tali pacchetti vadano persi

Valutazioni Qualitative

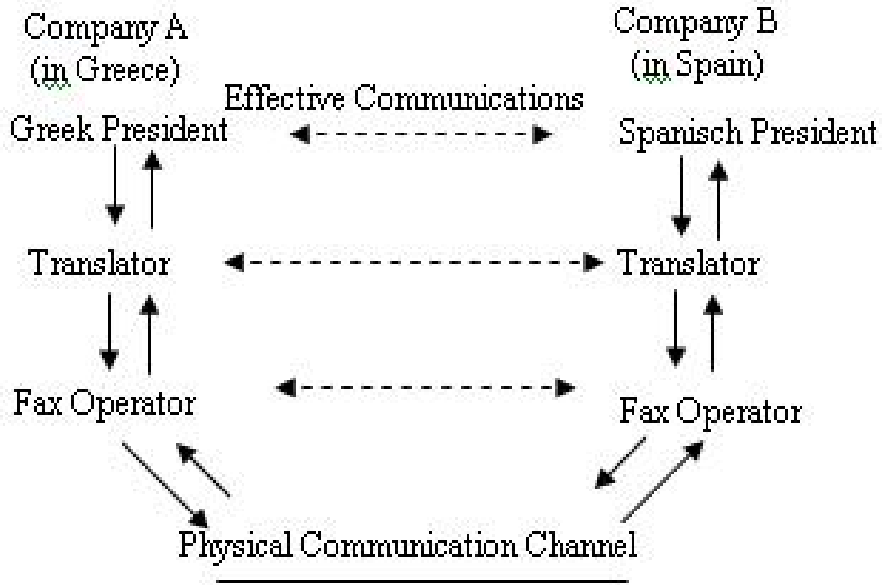
- **Commutazione di Circuito:**
  - **Indicata per la trasmissione di un lungo e continuo flusso di dati**
- **Commutazione di Pacchetto:**
  - **Indicata per la trasmissione di dati discontinui e concentrati (bursty)**
  - **Permette dinamicità nell'instradamento**
  - **Permette di sfruttare il cosiddetto effetto "pipelining", che quantificheremo tra poco**

Architettura a Strati

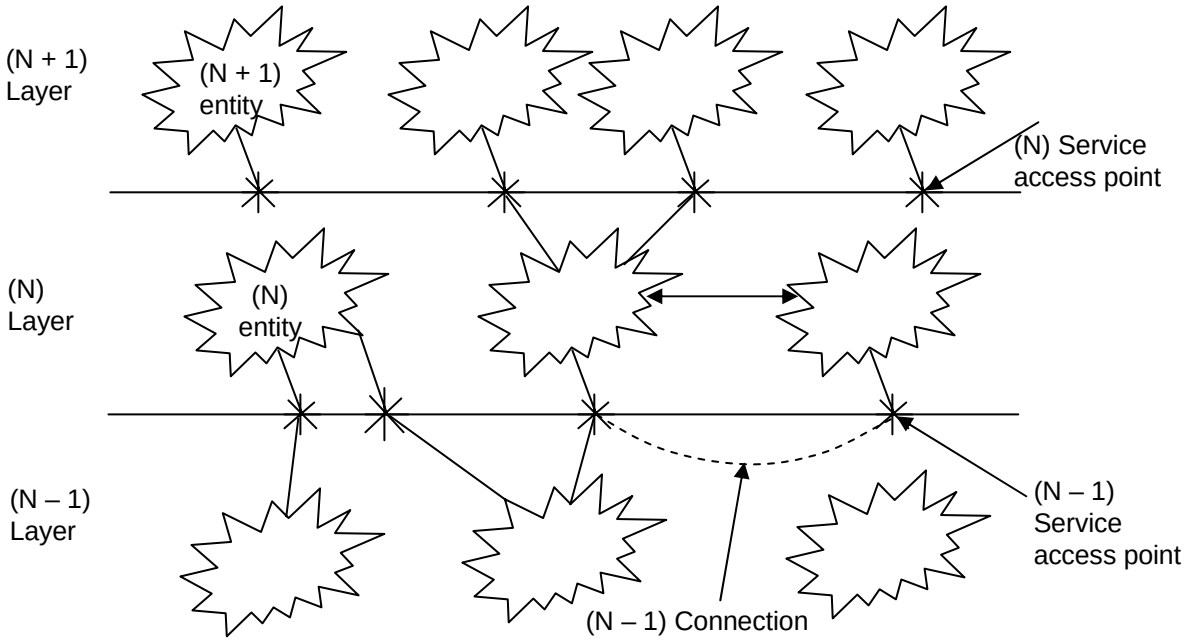
- **Protocollo di Comunicazione**
  - **Ha lo scopo di assicurare che due entità comunicanti possano inviare, ricevere e interpretare correttamente l'informazione che vogliono scambiarsi**

- **Architettura di Protocollo**
  - **Struttura logica in cui vengono posizionate le diverse funzioni del protocollo. Ha lo scopo di ridurre la complessità concettuale inerente la comunicazione end-to-end.**
  - **La maggior parte di tali architetture è basata sul concetto di stratificazione: la comunicazione end-to-end viene eseguita mediante funzioni a "valore aggiunto" eseguite ad ogni strato del protocollo.**

Architettura a Strati



Architettura a Strati

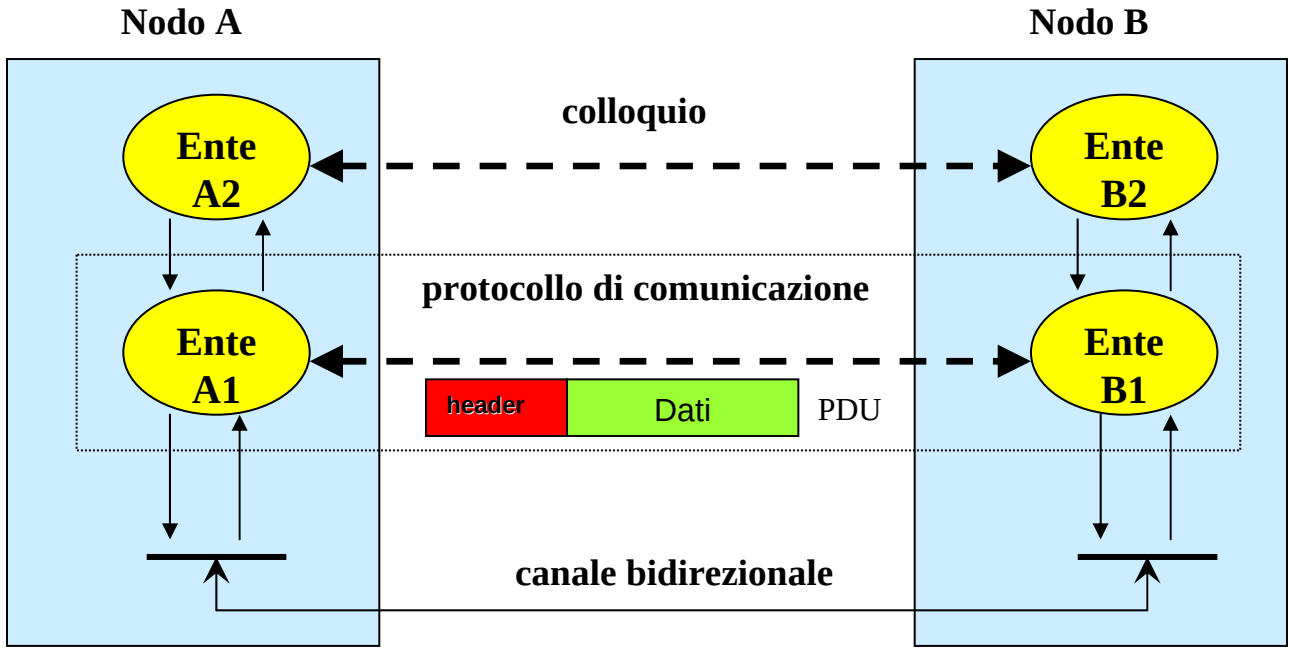


Architettura a strati

- I servizi di comunicazione complessi possono essere articolati a strati, ad esempio:
  - da un livello che garantisce solo il trasporto dei bit
  - a un livello dove sono definite complessi servizi caratterizzati da molti parametri e funzionalità

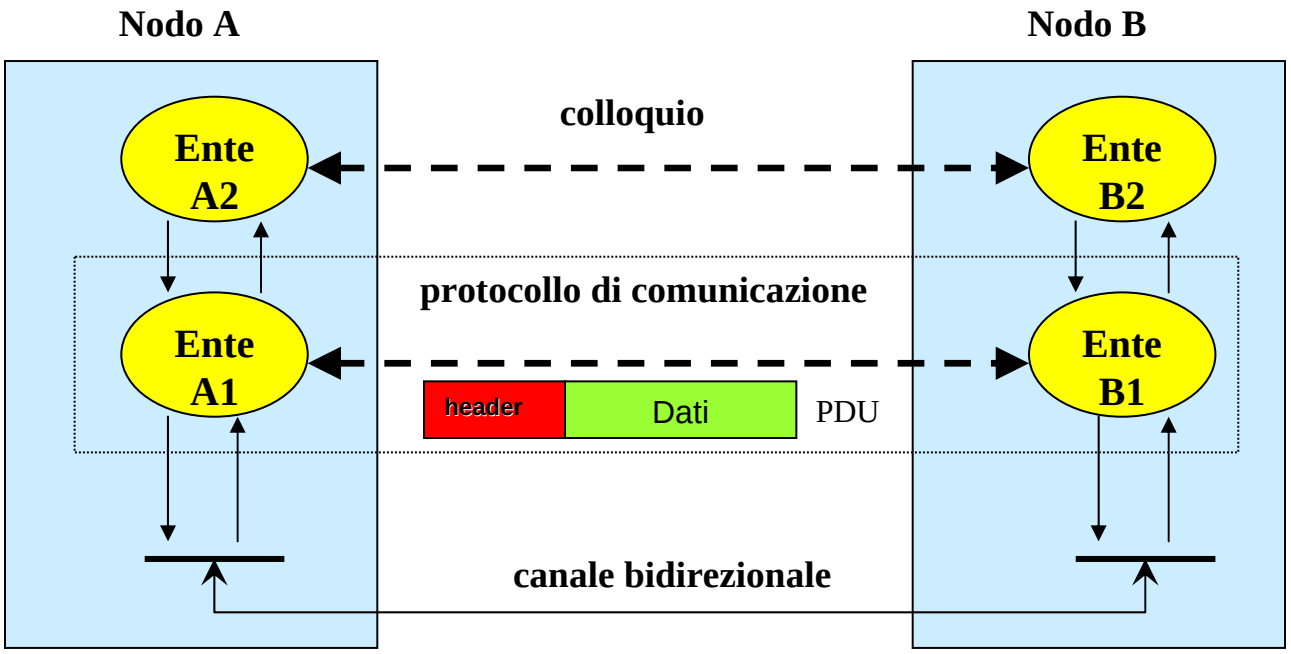
Livelli

- le entità che colloquiano in un servizio di telecomunicazione possono anche offrire un servizio di comunicazione a entità terze, dette di livello superiore



Livelli

- A che serve?
  - il servizio offerto alle entità di livello superiore può essere diverso da quello base



Protocolli di comunicazione

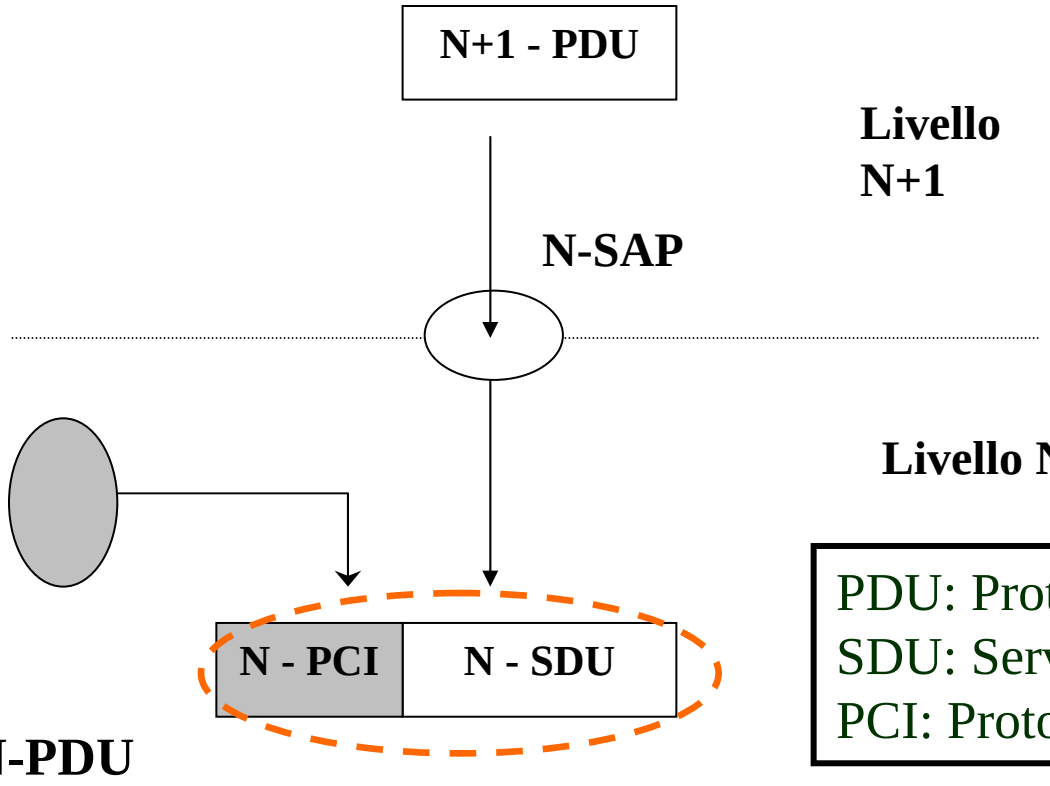
- le entità di un livello collaborano per fornire il servizio di comunicazione al livello superiore e si scambiano messaggi mediante il servizio offerto dal livello inferiore
- **Protocollo**:
  - Insieme delle regole che sovrintendono al colloquio tra entità dello stesso livello
    - ✓ formato dei messaggi
    - ✓ informazioni di servizio
    - ✓ algoritmi di trasferimento
    - ✓ ecc.

Protocol Data Units (PDU)

- un protocollo utilizza per il colloquio tra entità dello stesso livello delle unità di trasferimento dati dette PDU o anche trame del protocollo
- Le PDU possono contenere:
  - informazione di servizio necessaria al coordinamento tra le entità
  - informazione vera e propria ricevuta dai livelli superiori

Relazioni tra i livelli

- **Il servizio offerto da uno strato è rappresentato da un Service Access Point (SAP)**



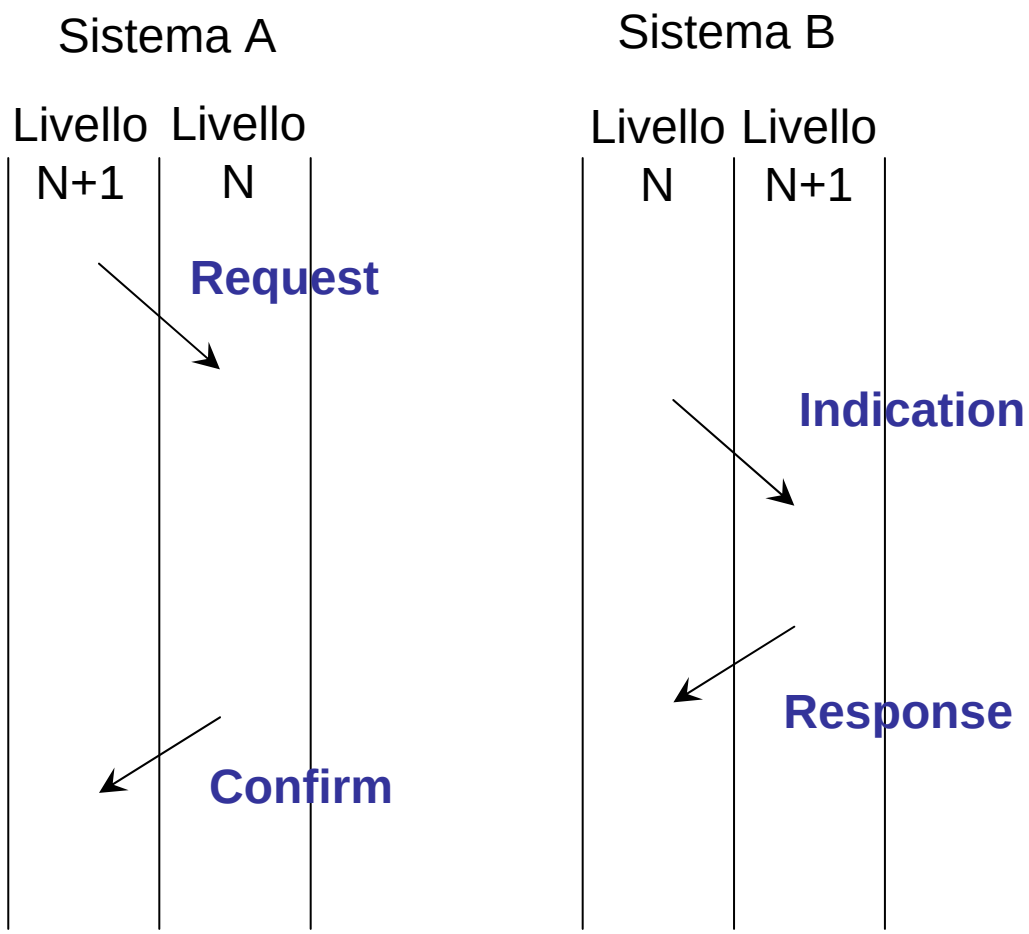
PDU: Protocol Data Unit
SDU: Service Data Unit
PCI: Protocol Control Information

N-PDU=(N+1)-PDU + N-PCI

Primitive di Servizio

- Le primitive di servizio denotano le interazioni con l'interfaccia verso il protocollo, necessarie ad espletare il servizio.
- Se ne sono individuate quattro, non tutte sempre necessarie:
  - Request
  - Indication
  - Response (facoltativa)
  - Confirm (facoltativa)

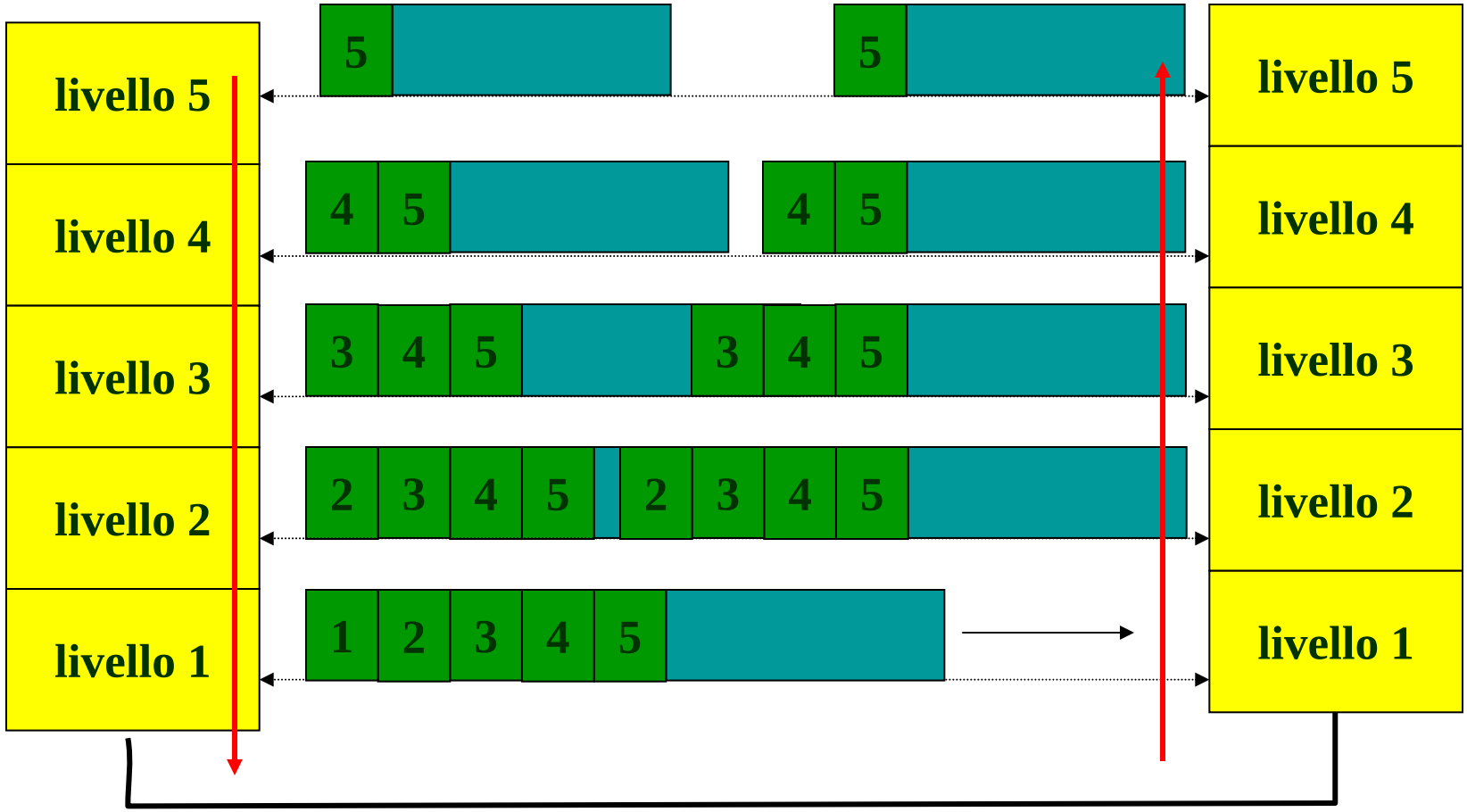
Primitive di Servizio



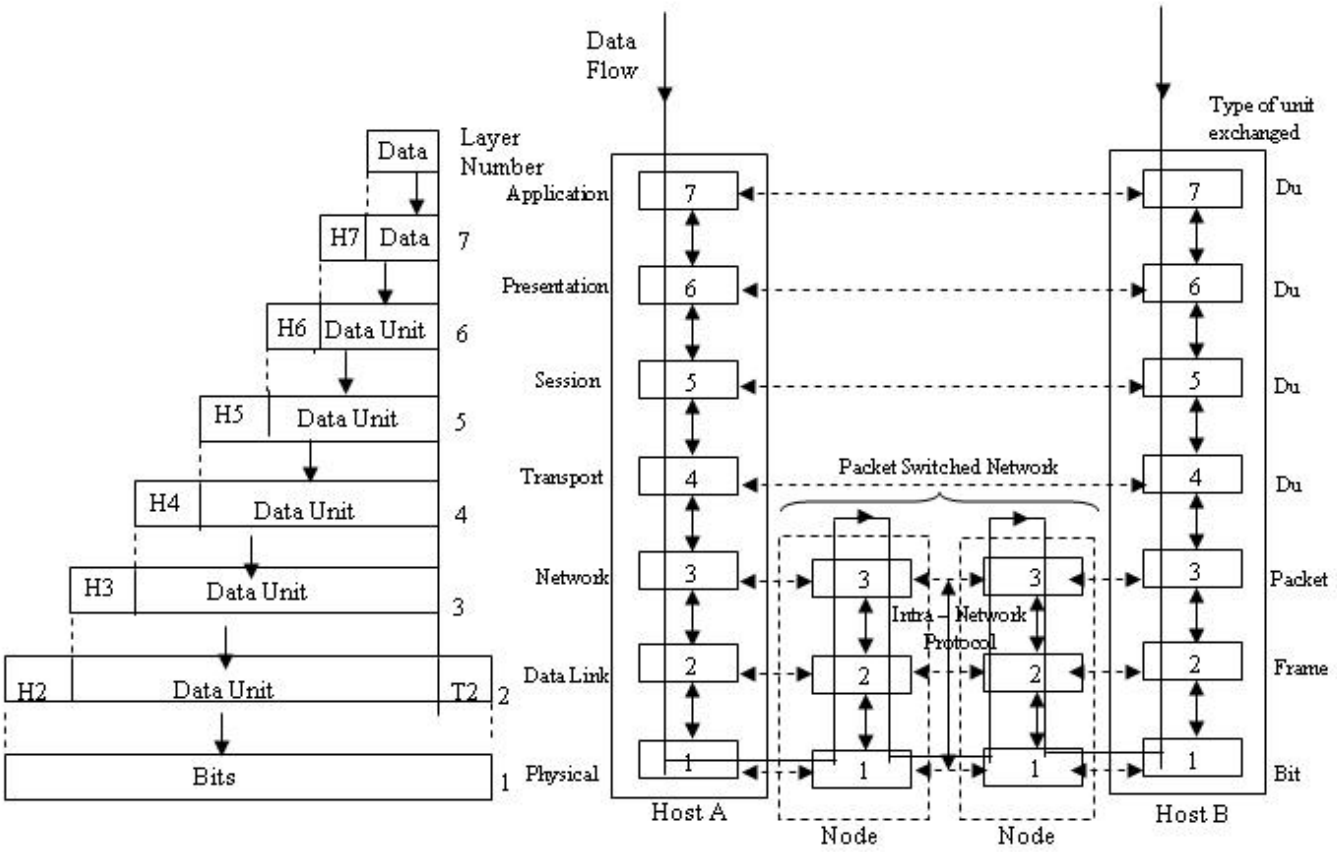
Primitive di Servizio

- **Request**
  - **la richiesta di A a livello N+1 fa inviare una N-PDU da A a B**
- **Indication**
  - **alla ricezione della N-PDU in B viene inviata una segnalazione a livello N+1**
- **Response (facoltativa)**
  - **il livello N+1 risponde alla segnalazione. Il livello N di B genera una PDU per il livello N di A**
- **Confirm (facoltativa)**
  - **il livello N di A invia conferma della richiesta al livello N+1. Se non usata, in pratica, il livello N+1 di A "si fida"**

Esempio di Architettura a 5 livelli



Architettura OSI



Dun : Data Unit
H1 : Layer 1 Header (1= 1, 2, 3…..,7)

Architettura OSI

- I livelli 1, 2, 3 forniscono funzioni di trasmissione e di rete (eseguiti dalla rete)
- I livelli 5, 6, 7 forniscono funzioni di elaborazione, colloquio e controllo (eseguiti dall'utente)
- Il livello 4 fa da collegamento fra gli strati dedicati alla comunicazione e quelli orientati alla elaborazione

Funzioni Tipiche svolte dai livelli inferiori

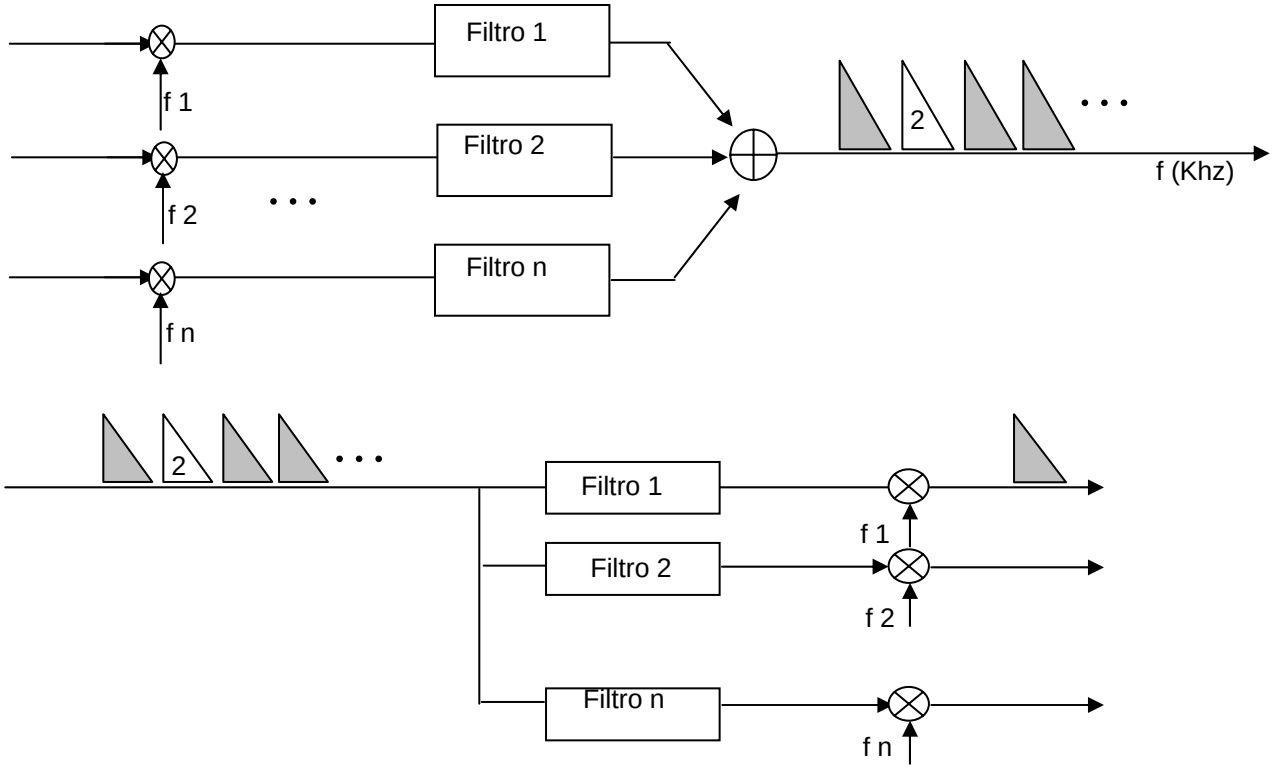
- **Multiplazione**
- **Recupero degli Errori**
- **Instradamento**

Multiplazione

- Più PDU a livello fisico (bit, byte, pacchetti), provenienti da diversi SAP, utilizzano lo stesso supporto trasmissivo rimanendo distinguibili mediante associazioni puramente fisiche (frequenza, tempo, codice...)

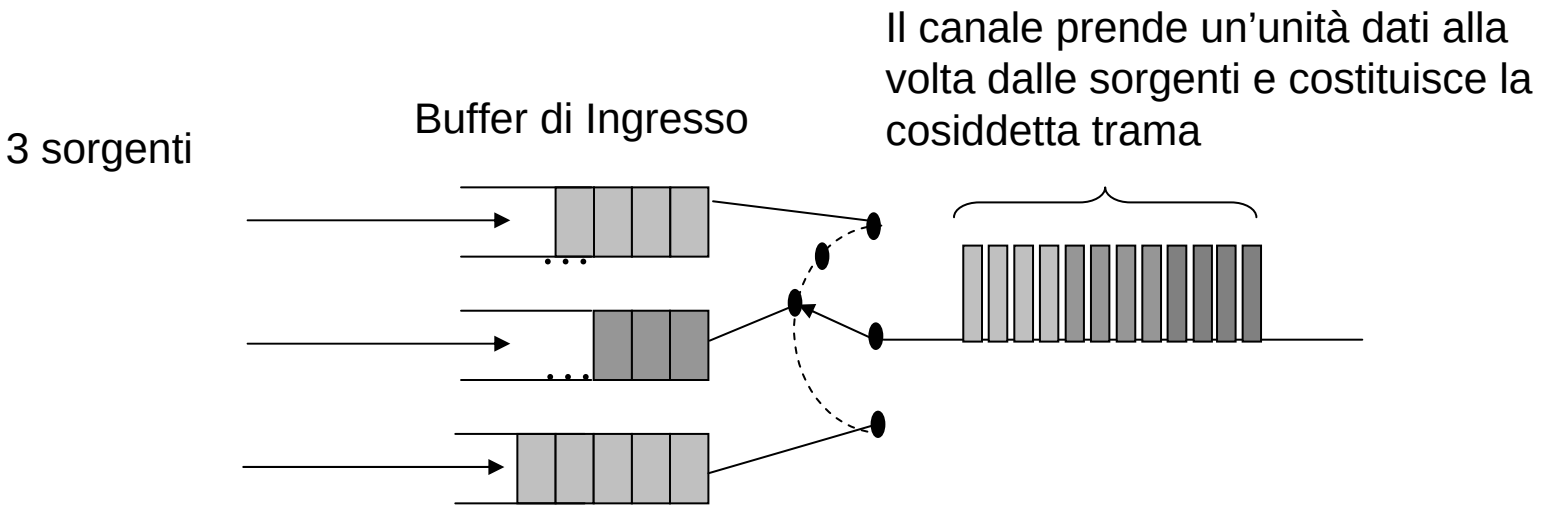
- Tecniche di multiplazione a Divisione di:
  - Frequenza, FDM (Frequency Division Multiplexing)
  - Tempo, TDM (Time Division Multiplexing)
  - Codice, CDM (Code Division Multiplexing)
  - Lunghezza d'onda, WDM (Wavelength Division Multiplexing)

Multiplazione a Divisione di Frequenza



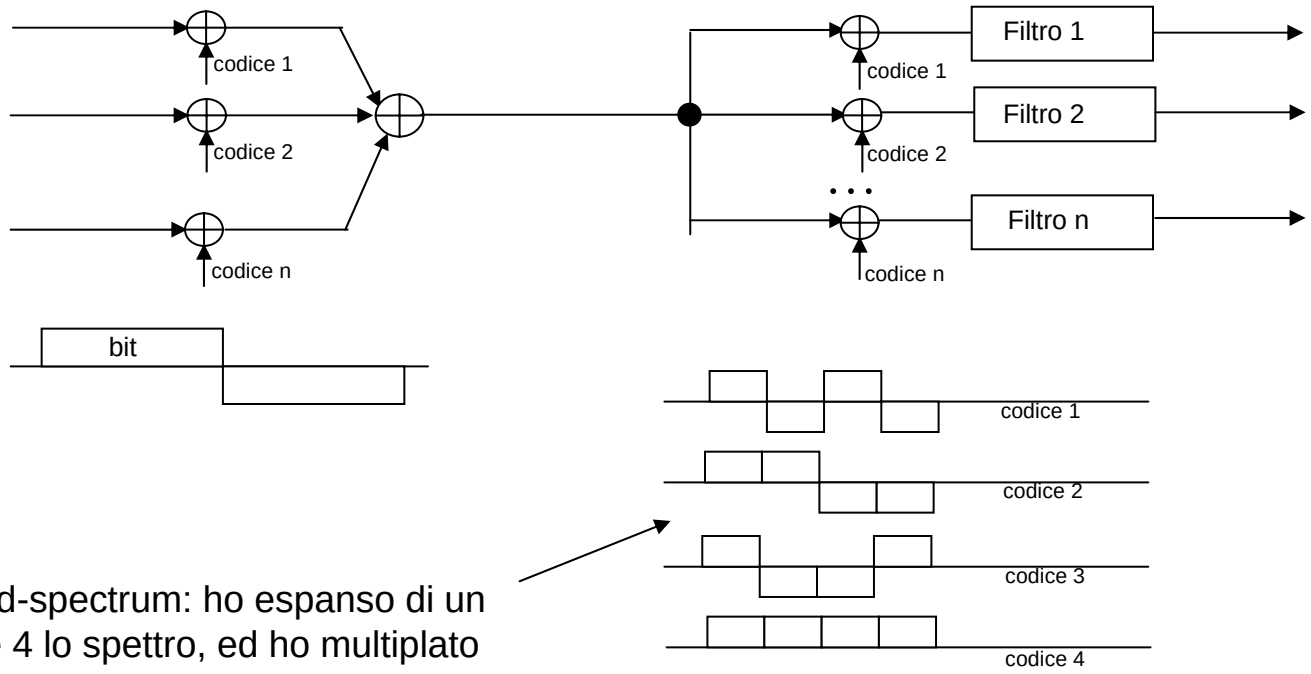
- Schema di Multiplazione e Demultiplazione FDM a banda laterale unica
- N canali di banda B vengono multiplati in una banda NB + Bande di Guardia

Multiplazione a Divisione di Tempo



- **Struttura logica di un multiplatore TDM**

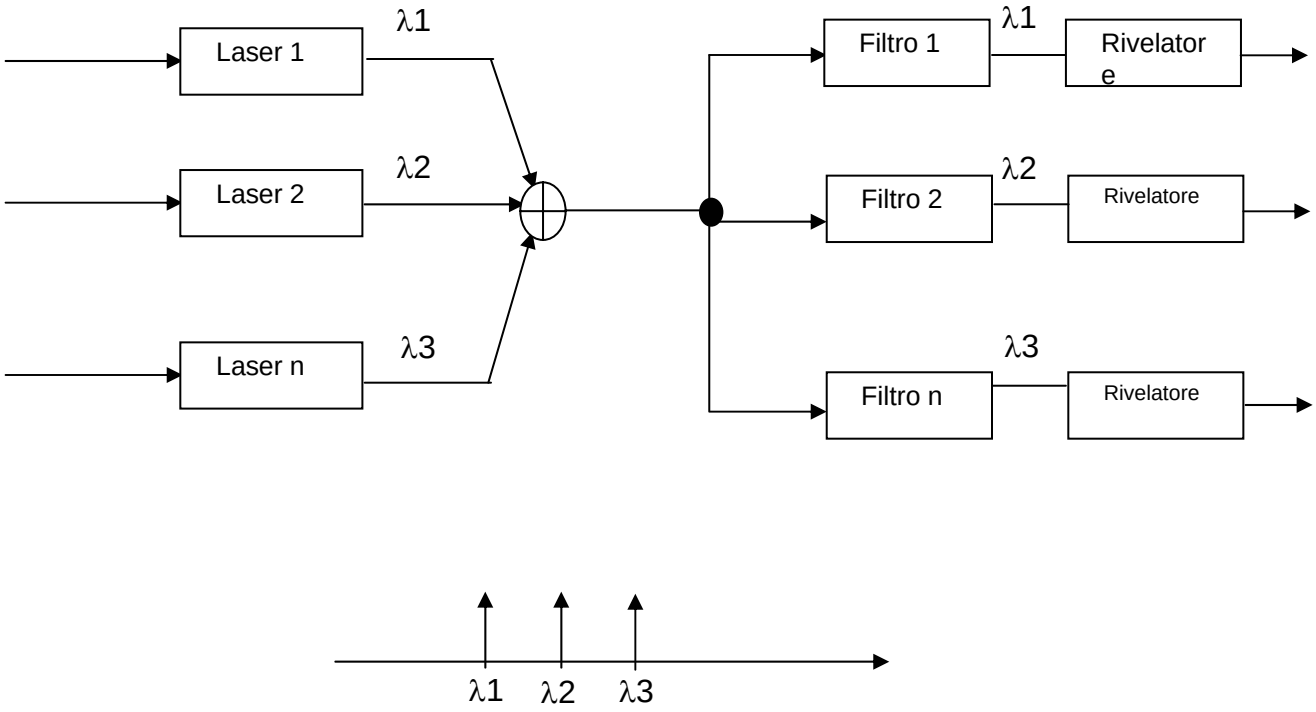
Multiplazione a Divisione di Codice



Spread-spectrum: ho espanso di un fattore 4 lo spettro, ed ho multiplato dunque 4 canali

- **Struttura logica di un multiplatore CDM**
- **N canali di banda B vengono multiplati in una banda NB**

Multiplazione a Divisione di Lunghezza d'onda



- **Schema di multiplazione WDM**

Controllo di Errore

- **Automatic Repeat reQuest (ARQ)**
- **Questo meccanismo riconosce le trame errate al nodo ricevente e ne richiede la ritrasmissione al nodo trasmittente**
- **Questo meccanismo è spesso implementato a livello 2 della pila OSI (Data Link)**

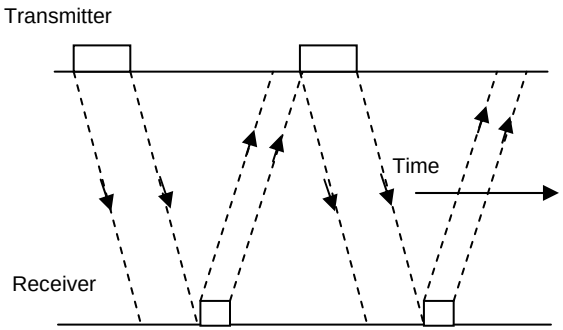
- **Tali algoritmi vengono valutati sulla base di due parametri:**
  - **Correttezza: il nodo ricevitore riceve tutte e sole le trame correttamente?**
  - **Efficienza: quanta capacità del canale viene sprecata?**
- **Algoritmi utilizzati:**
  - **Stop & Wait**
  - **Go-Back-N**
  - **Selective Repeat**

Efficienza (e complessità implementativa) crescente

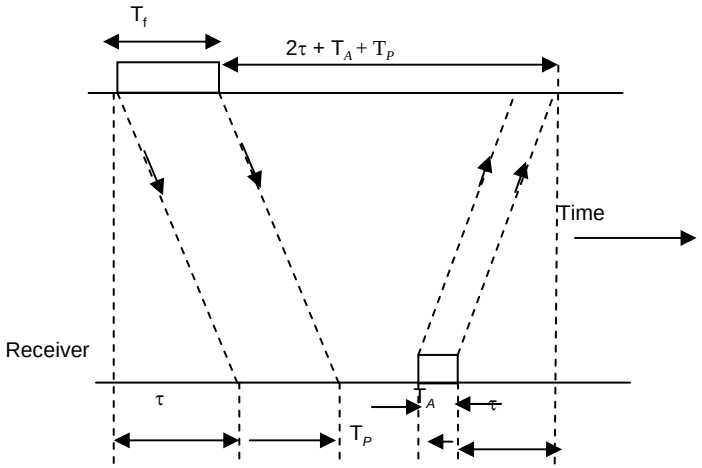
Stop & Wait

- Viene trasmessa una sola trama alla volta
- Il trasmettitore aspetta ACK/NACK (Negative ACKnowledgment), o lo scadere di un TIME-OUT
- Se viene ricevuto un ACK: la trasmissione è corretta e la trama viene eliminata dal buffer di trasmissione
- Se viene ricevuto NACK o scade un TIME-OUT: trasmissione errata. La trama viene ritrasmessa

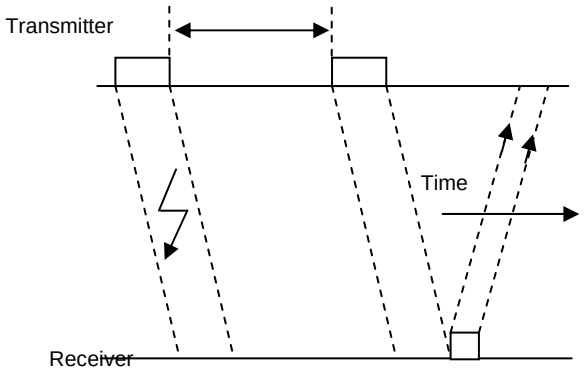
Stop & Wait



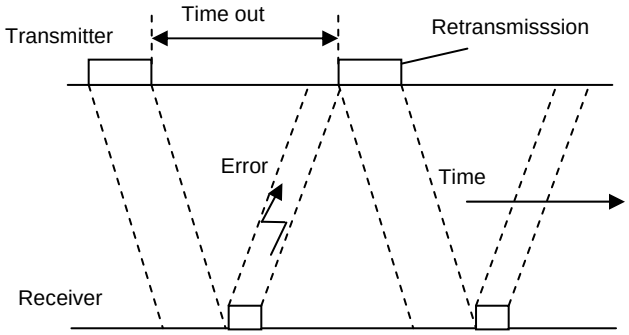
Operazione senza errori



Calcolo Time-Out
di durata minima



Recupero da Errori sulla Trama



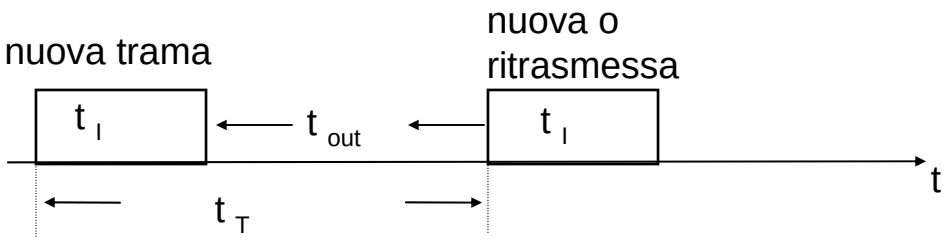
Recupero da Errori sull'ACK

Stop & Wait: Osservazioni

- Adatto per collegamenti half-duplex
- Inefficiente nei collegamenti full-duplex
- Problema: senza numerazione delle trame può verificarsi una ripetizione di una trama corretta se un ACK va perso
  - Infatti: se l'ACK viene perso, allo scadere del time-out il trasmettitore la re-invia, ed il ricevitore la considera come una nuova trama!

- E' dunque necessaria la numerazione di trama per riconoscere i duplicati
  - è sufficiente un Sequence Number SN=0,1 in quanto l'ambiguità può riguardare solo trame consecutive
- Dualmente, necessità di numerazione degli ACK (portano nell'intestazione il numero della prossima trama attesa, Request Number RN=0,1)
- NOTA: per aumentare l'efficienza, l'ACK può essere inviato (*piggybacked)* nell'intestazione di trame inviate nella direzione opposta (in caso lo scambio di informazioni sia full-duplex)

Stop & Wait: Prestazioni



t_l = tempo di trasmissione trama

t_{out} = time-out $\geq 2\, t_p + t_{proc} + t_{ack}$

Propagazione

Elaborazione trama

Trasmissione ACK

- **Ipotesi:**
  - ✓ **A trasmette a B**
  - ✓ **A ha sempre trame pronte da inviare verso B (backlogged, o saturazione)**
  - ✓ **al tempo t_{out} o arriva ACK o NACK (ovvero: A conosce l'esito delle sua trasmissione)**
  - ✓ **A al massimo trasmette una trama ogni $t_T = t_I + t_{out}$**
  - ✓ **p = probabilità che la trama sia ricevuta errata**
  - ✓ **La trama viene ritrasmessa finché non viene ricevuta correttamente**
  - ✓ **t_v = tempo medio per una corretta trasmissione di trama (che ora determineremo)**

Stop & Wait: Prestazioni

$$t_v = t_T + (1-p)\sum_{i=1}^{\infty} t_T i p^i =$$

t_v = tempo medio per una corretta trasmissione di trama

$$= t_T + t_T (1-p) p \frac{d}{dp} \sum_{i=1}^{\infty} p^i$$

La probabilità di sbagliare ACK è stata trascurata (trama più corta implica minore prob. di errore)

$$= t_T + t_T (1-p) p \frac{d}{dp} \left(\frac{1}{1-p} - 1 \right)$$

$$= t_T \left(1 + \frac{p}{1-p} \right) = t_T / (1-p)$$

$$\text{max throughput} = \lambda_{\max} = \frac{1}{t_v} = \frac{1-p}{t_T} = \frac{1-p}{a t_I}$$

Capacità massima del sistema, espressa in **Trame/secondo** che il protocollo smaltisce

$$a \equiv \frac{t_T}{t_I} \geq 1$$

Se λ = effettiva frequenza degli arrivi dei pacchetti < λ_{max}, l'utilizzo del canale risulta

$$g = \lambda t_I \leq \frac{1-p}{a} < 1$$

L'utilizzo del canale cala con ***p*** e ***a***

Stop & Wait: Passaggio Matematico

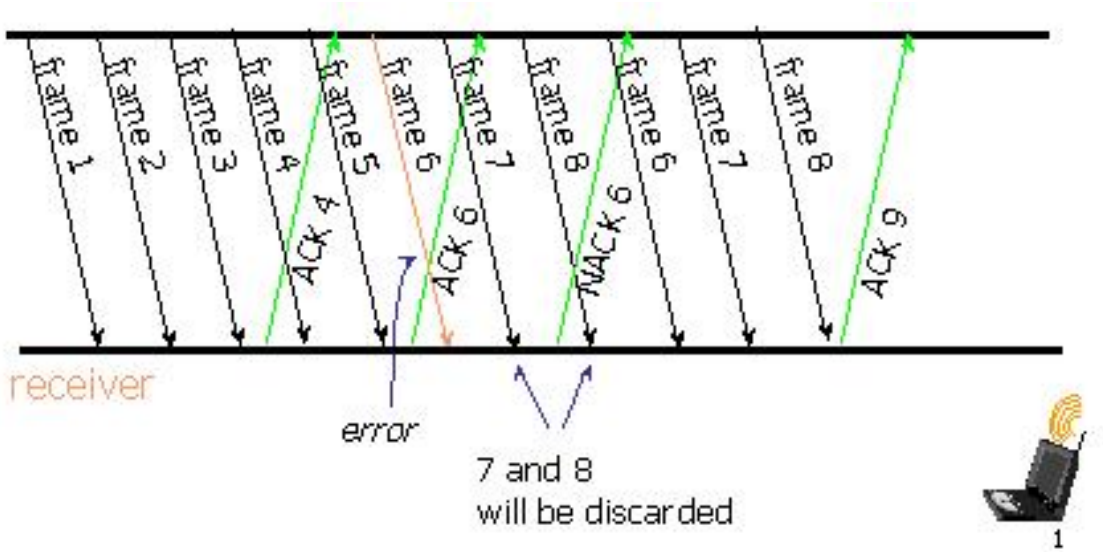
$$\sum_{i=1}^{\infty} ip^i = p \cdot \sum_{i=1}^{\infty} ip^{i-1} = p \cdot \frac{d}{dp}\left(\sum_{i=1}^{\infty} p^i\right) = p \cdot \frac{d}{dp}\left(\frac{1}{1-p} - 1\right) = \frac{p}{(1-p)^2}$$

Go-Back-N

- **Nell'algoritmo di Stop & Wait il protocollo è "strozzato" dalla sua parte di "Wait" (il canale rimane inutilizzato in attesa che A riceva un ACK o scada un timeout)**
- **Nell'algoritmo Go-Back-N si permette al trasmettitore di trasmettere trame prima di ricevere un ACK per ciascuna di esse**
- **N, dimensione della finestra, rappresenta il "credito" di trame che il trasmettitore può inviare prima di ricevere un ACK**
  - **Nei canali "normali" terrestri valori tipici di N=8**
  - **Per canali satellitari si può arrivare anche ad N=128**

- **Le trame portano la numerazione SN e, se disponibili, vengono trasmesse in modo continuo. Il trasmettitore può trasmettere fino alla trama *i+N* se la trama *i* è l'ultima trama riscontrata**
- **Alla ricezione di NACK o allo scadere del TIME-OUT per la trama *i*, la trama *i* e tutte quelle successive già trasmesse vengono ritrasmesse**
- **Alla ricezione di ACK con numerazione RN vengono riscontrate tutte le trame fino alla RN-1 (si parla di *ACK cumulativi*)**

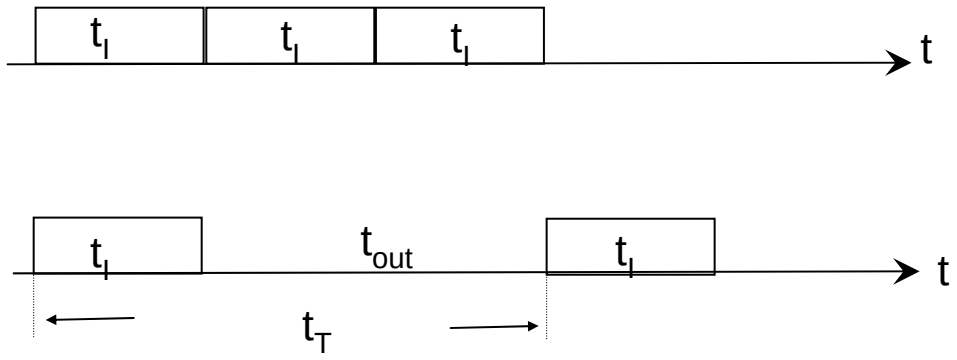
Go-Back-N: Funzionamento



- + Tempo di ritorno ACK meno critico rispetto a Stop & Wait
- + Minore overhead per trasmissione ACK (che sono cumulativi)
- + Un eventuale errore nella trasmissione di ACK può essere recuperato dal successivo ACK
- - Necessità di memorizzare N trame nel trasmettitore: è dunque necessario un buffer più grande nel Tx (rispetto a Stop & Wait). In ricezione NON ho bisogno di buffer (le trame fuori ordine non vengono accettate)

Go-Back-N: Prestazioni

- Se non ci sono errori la trasmissione avviene in modo continuo (supposto che il trasmettitore abbia sempre trame da trasmettere)
- Se c'è errore perdo le trasmissioni avvenute durante t_T (infatti le ritrasmetto tutte di nuovo, come da algoritmo)



Go-Back-N: Prestazioni

$$t_v = t_I + (1-p)\sum_{i=1}^{\infty} i p^i t_T$$

Quando mi va bene: impiego solo t_I
Quando va male: è come per Stop & Wait

$$= t_I + t_T \frac{p}{1-p} = t_I \frac{1+(a-1)p}{1-p}$$

$$\begin{cases} \lambda_{\max} = \dfrac{1}{t_v} = \dfrac{1-p}{t_I(1+(a-1)p)} \\ \rho = \lambda_{\max} t_I < \dfrac{1-p}{1+(a-1)p} \end{cases}$$

Capacità Massima del sistema (trame/secondo)

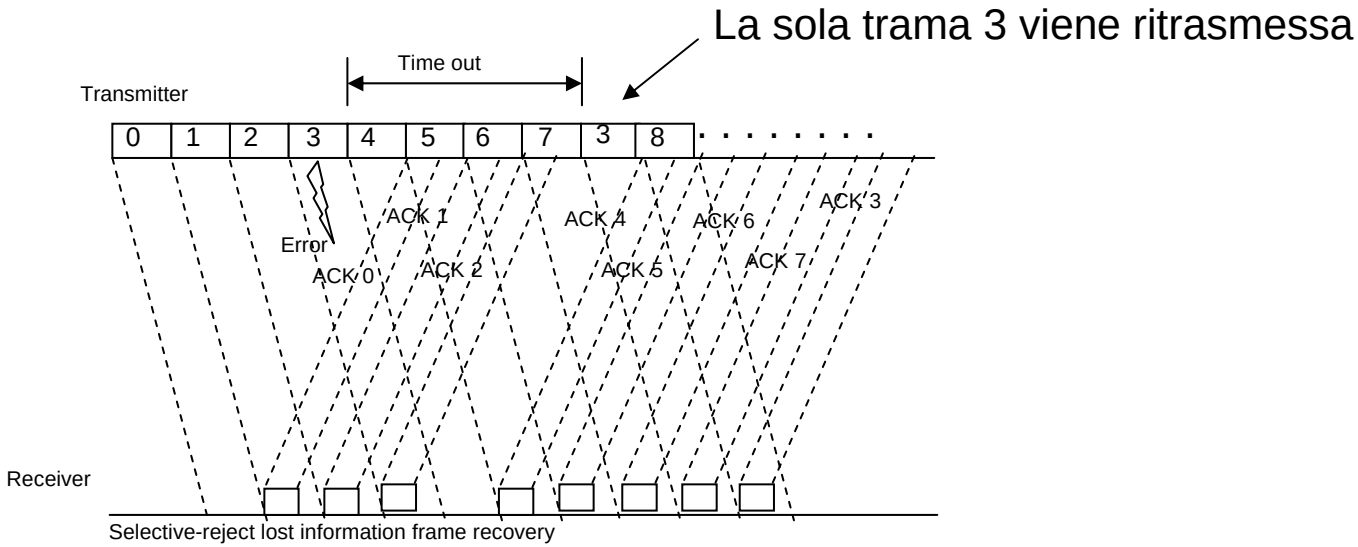
Traffico smaltito < Capacità Massima del sistema

- **Se a = 1 (tempi di propagazione ed elaborazione trascurabili rispetto a t_I) Go-Back-N = Stop & Wait**
- **Infatti, anche nello Stop & Wait, in questo caso, saprei immediatamente se la trame ha subìto errori o meno, permettendomi di ritrasmettere se necessario**

Selective Repeat

- **Le trame portano la numerazione SN e, se disponibili, vengono trasmesse in modo continuo.**
- **Alla ricezione di NACK o allo scadere del TIME-OUT per la trama *i*, la trama *i* (e solo la trama *i*) viene ritrasmessa.**
- **Alla ricezione di ACK con numerazione RN vengono riscontrate tutte le trame fino alla RN-1 (ACK cumulativi)**

Selective Repeat: Funzionamento



- Causa possibili errori le trame sono trasmesse fuori sequenza
- Necessità di buffer in trasmissione ed anche in ricezione. In trasmissione per poter ritrasmettere le trame errate. In ricezione per poter riordinare le trame corrette
- Lunghezza del buffer dipende dal tempo di propagazione (# di trame in viaggio)

Selective Repeat: Prestazioni

- Si ritrasmettono solo le trame errate
- Necessità di un buffer di riordinamento
- E (n) = numero medio di trasmissioni per trama corretta =

$$= (1-p)\sum_{i=1}^{\infty} i p^{i-1} = \frac{1}{1-p}$$

$$\rho = 1-p$$

- Non dipende da *a*
- Coincide con Go-Back-N se *a* =1
- Selective Repeat non è il meccanismo più usato proprio in virtù del non ordinamento delle trame

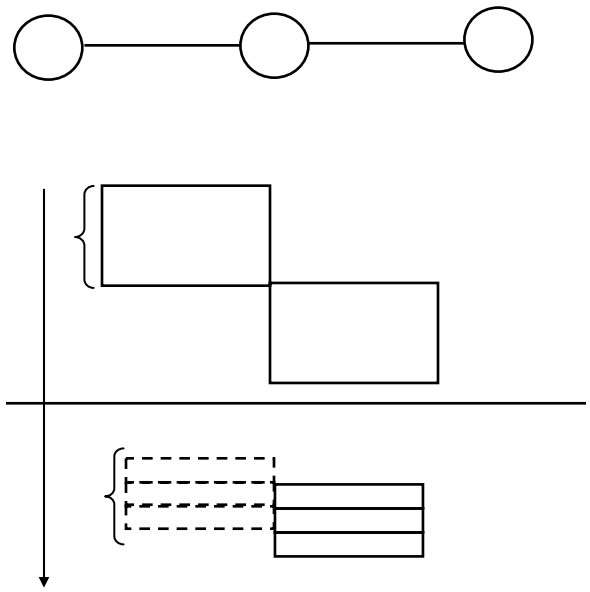
Instradamento: Determinazione Lunghezza Ottima Trama

- **Overhead vs Pipelining**
- **Si definiscano:**
  - **M** **Lunghezza messaggio**
  - **K_{max}** **Lunghezza massima campo dati di un pacchetto**
  - **V** **Overhead di trama (dovuto tipicamente all'header)**

$$\text{Bit trasmessi per messaggio} = M + \left\lceil \frac{M}{K_{\max}} \right\rceil \cdot V$$

Determinazione Lunghezza Ottima Trama

- Esiste il Trade-off seguente:
- Se scelgo K_{max} grande:
  - meno trame, meno overhead
  - meno processing nei nodi
- Se scelgo K_{max} piccolo:
  - più efficienza controllo errore
  - minori richieste di buffer
  - minor ritardo di accesso (effetto pipe-lining)



Determinazione Lunghezza Ottima Trama

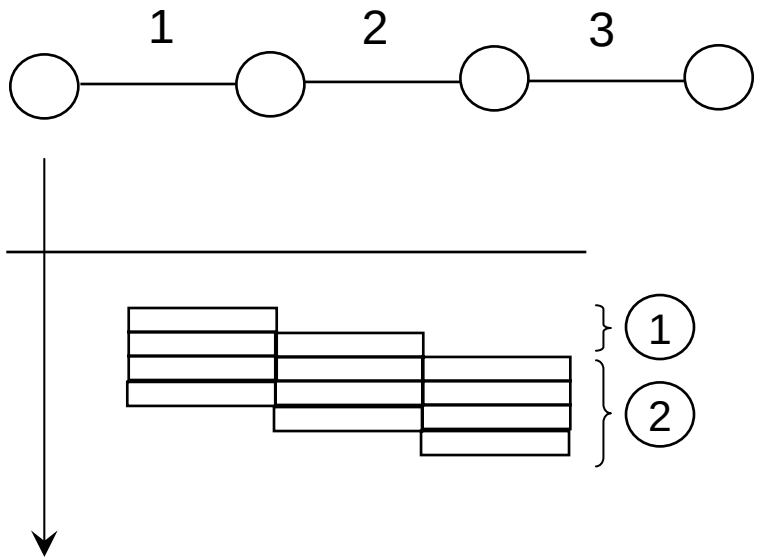
- **Ipotesi:**
  - **ritardo di propagazione trascurabile**
  - **probabilità di perdita dei pacchetti nulla**
  - **Nessuna coda nei nodi**
- **Detti:**
  - **T: ritardo di trasmissione su *j* canali in cascata di uguale capacità**
  - **C: capacità dei canali bit/s**
  - **T*C: ritardo di trasmissione misurato in bit (Nota: calcolo T*C anziché T direttamente solo per comodità di notazione, per evitare di avere nell'equazione a destra un fattore 1/C)**

$$T \cdot C = (K_{\max} + V)(j-1) + M + \left\lceil \frac{M}{K_{\max}} \right\rceil V$$

tempo (bit) impiegato dal 1° pacchetto per attraversare j-1 canali

tempo (bit) di trasmissione dell'intero messaggio nell'ultimo canale

Determinazione Lunghezza Ottima Trama: Esempio per j=3



$$TC = (K_{\max} + V)(j-1) + M + \left\lceil \frac{M}{K_{\max}} \right\rceil V$$

1

2

tempo (bit) impiegato dal I° pacchetto per attraversare j-1 canali

tempo (bit) di trasmissione intero messaggio nell'ultimo canale

Determinazione Lunghezza Ottima Trama

Mediando sulla lunghezza dei messaggi

$$E[TC] \cong (K_{\max} + V)(j-1) + E[M] + \left[\frac{E[M]}{K_{\max}} + \frac{1}{2}\right]V$$

Non cambia: il 1° pacchetto è sempre pieno

$$E\left(\left\lceil \frac{M}{K_{\max}} \right\rceil\right) = E\left(\frac{M}{K_{\max}}\right) + \frac{1}{2}$$

L'ultimo pacchetto è, in media, mezzo pieno (o mezzo vuoto)

Derivo rispetto a K_{max} e minimizzo il ritardo:

$$\frac{\partial E[TC]}{\partial K_{\max}} = (j-1) - \frac{E[M]V}{K_{\max}^2} = 0 \qquad K_{\max} \cong \sqrt{\frac{E[M]V}{j-1}}$$