

# Reti di Telecomunicazioni



Introduzione

---



# Autori



Queste slides sono state scritte da

Michele Michelotto:

[michele.michelotto@pd.infn.it](mailto:michele.michelotto@pd.infn.it)

che ne detiene i diritti a tutti gli effetti



# Copyright Notice

Queste slides possono essere copiate e distribuite gratuitamente soltanto con il consenso dell'autore e a condizione che nella copia venga specificata la proprietà intellettuale delle stesse e che copia e distribuzione non siano effettuate a fini di lucro.



# Bibliografia

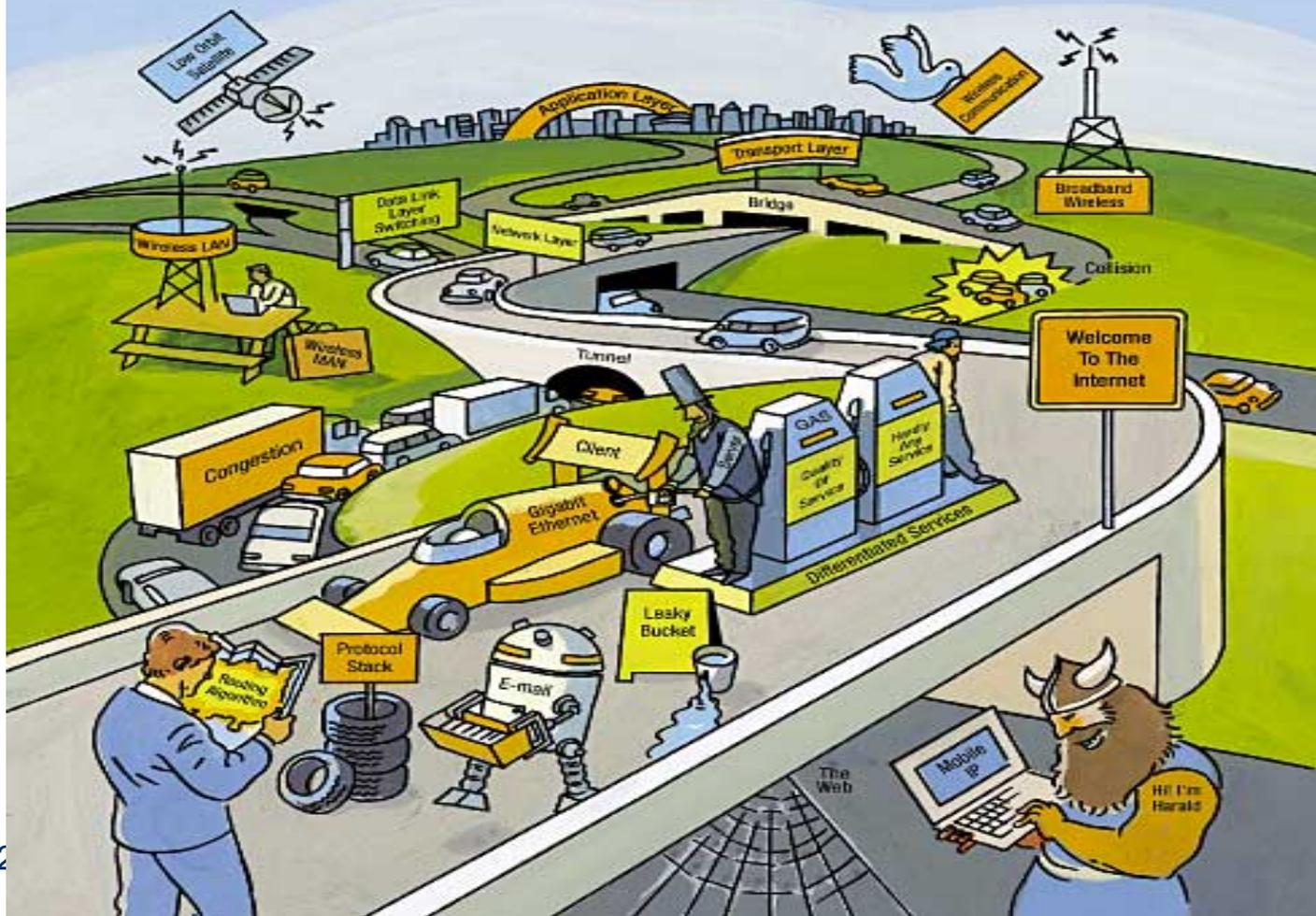
- **Computer Networks – Andrew S. Tanenbaum**
  - Quarta edizione 2002 , Prentice Hall
  - Esiste anche una versione in italiano
- **Networking e Internet – Fred Halsall**
  - Quinta Edizione 2006, Pearson
- **Reti di calcolatori e Internet – Behrouz A. Forouzan, 2007 McGraw-Hill**
  - Prima Edizione 2008, traduzione della quarta edizione in lingua inglese
- **NB le slides seguono per gran parte il Tanenbaum**



FOURTH EDITION

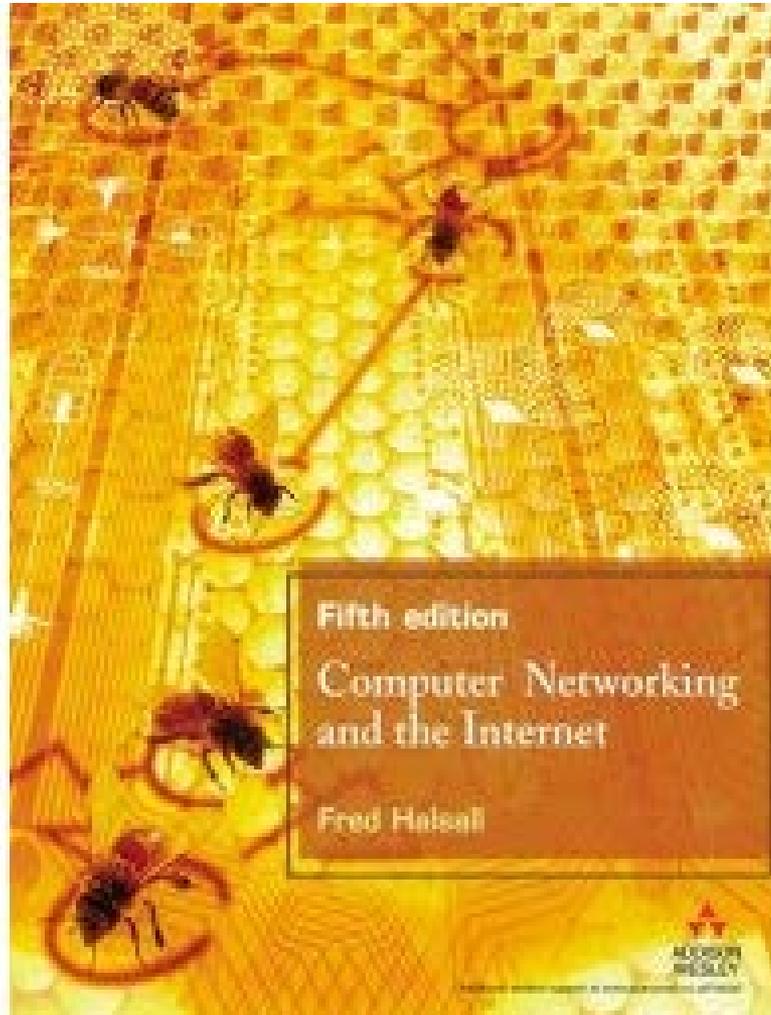
# Computer Networks

ANDREW S. TANENBAUM





# Halsall





# Introduzione



## Introduzione

Modello OSI e TCP/IP

Physics Layer

Data Link Layer

MAC sublayer

Network Layer

Transport Layer

Application Layer



# Perchè una rete

- **Condivisione di risorse**
  - In origine per collegare diversi computer isolati (inventario, paghe, magazzino, ordini, fatture)
  - Permettere di accedere remotamente a **informazioni**, utensili a controllo numerico e dati
- **Alta affidabilità**
  - Distribuire dati su diversi computer per accedere in caso di guasti, anche se ne basterebbe uno solo
  - Se un nodo si rompe gli altri continuano a lavorare



# Comunicazione



- Mezzo di comunicazione
  - Collaborare remotamente a progetti comuni, scambio di informazioni → Internet, e-commerce, e-government, e-learning, dematerializzazione documentale, voip, social network)





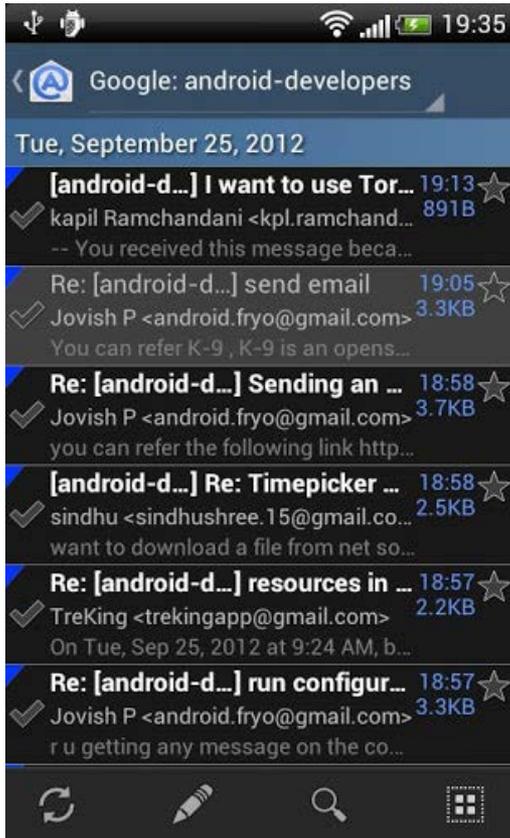
# Mezzo di comunicazione



- E-mail – comunicazione quotidiana.
- Groupware, Social Network
- Audio / Video Conferenza, chat, Voip
- B2B – Business to business (ordini, supply chain, real time inventory, fatturazione elettronica)
- B2C – Business to consumer, E-commerce



# Smartphone





# Tablet and iPad





# Perché un corso di reti?



- Cosa vi spinge a seguire un corso di reti?



# Perché un corso di reti



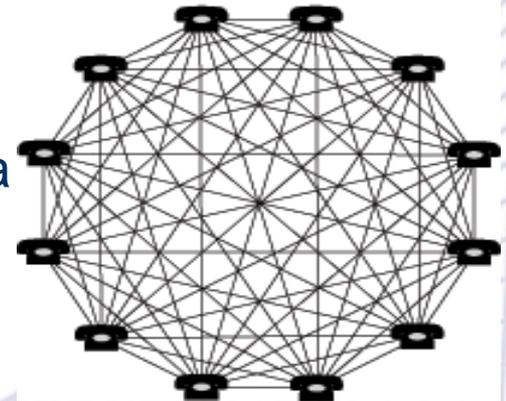
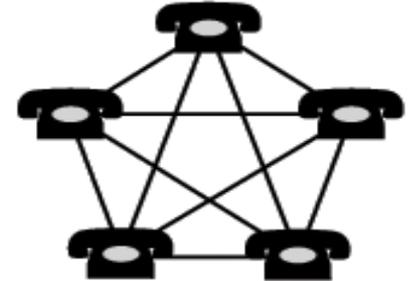
- Le reti di telecomunicazioni non interessano solo gli informatici ma tutti i campi della scienza, della conoscenza, dell'informazione ma anche della società
- Nelle computer science, conoscere il funzionamento del networking è fondamentale anche per chi segue indirizzi di tipo Database o Multimediale



# Valore della rete

- Legge di Metcalf

- Il valore della rete cresce con il quadrato del numero dei nodi  $n(n-1)/2$
- Se ho 4 PC ognuno parla con altri 3; il valore è 6
- Se aggiungo un altro PC in rete il valore sale a 10, e poi 15, 21...
- Alcuni, Odlyzko e Tilly, sostengono che non tutti i nodi hanno lo stesso valore e dicono che il valore della rete sale solo come  $n * \log n$
- Altri come Reed dicono il contrario, il valore della rete non è dato solo dalla rete nelle sua interezza ma anche da tutti i sottoinsiemi di nodi che si possono creare all'interno della rete stessa





# Classificazione delle reti



- Modi d'uso
- Tecnologia trasmissiva
- Dimensioni
- Aspetti software



# Modi di uso

| <b>Wireless</b> | <b>Mobile</b> | <b>Applicazione</b>                             |
|-----------------|---------------|---|
| No              | No            | PC in ufficio, edifici cablati                  |
| No              | Si            | Notebook in albergo, es via modem o via cavo    |
| Si              | No            | Notebook in ufficio o casa, edifici non cablati |
| Si              | Si            | Tablet, SmartPhone, schede GPRS, UMTS           |



# Aspetti hardware



- Tecnologia trasmissiva
  - Reti broadcast
  - Reti punto a punto
- Scala
  - PAN, LAN, MAN, WAN, Internet
- Topologia
  - Mesh, Stella, Bus, Anello, Ibrida



# Broadcasting

- Broadcasting

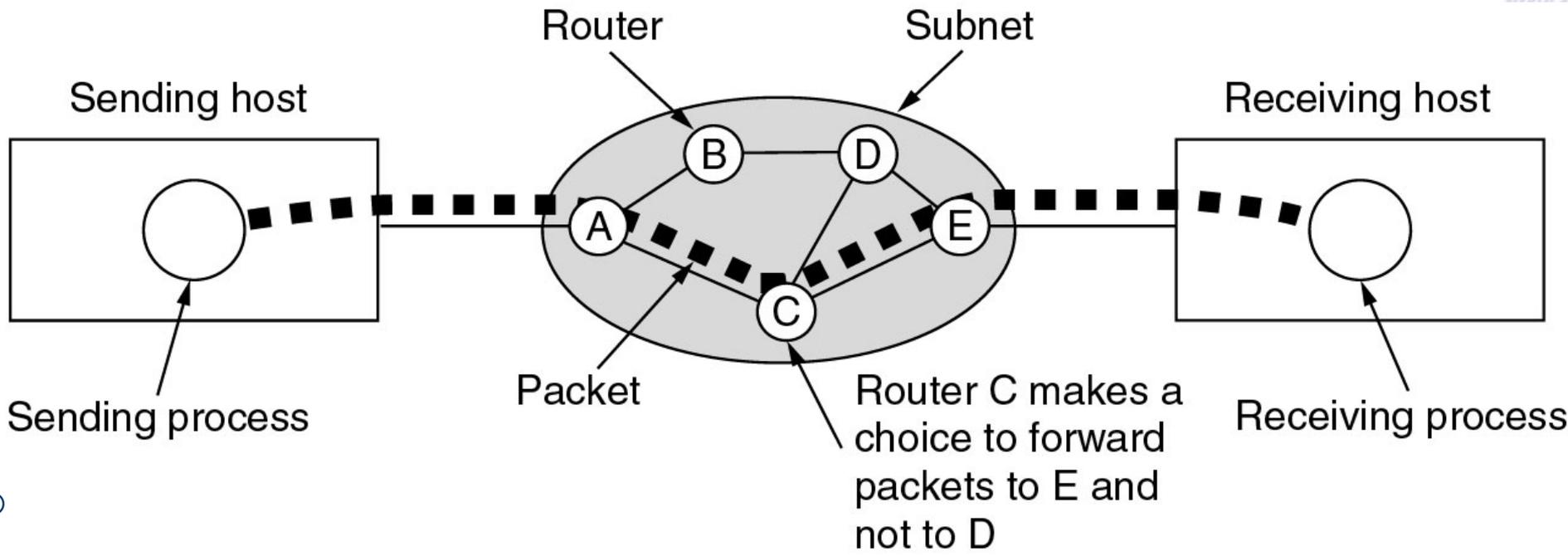
- Il canale di comunicazione è condiviso da tutte le macchine della rete.
- Piccoli messaggi (**packets**) sono mandati da una macchina a tutte le altre ma con un indirizzo del destinatario. (Es. "Signor Rossi venga qua!", "I passeggeri del volo 712 al gate 6")
- Alcuni sistemi broadcast possono mandare un pacchetto ad un sottoinsieme delle macchine:  
**multicasting**



# Unicast

- Reti punto a punto

- Diverse connessioni solo tra coppie di macchine.
- Per andare dal mittente al destinatario il pacchetto può passare per macchine intermedie e seguire diversi percorsi





# Dimensioni

- Dimensioni della rete

- Rete interne (multicomputer, matrici di switch, Infiniband, Myrinet)
- **PAN** Personal Area Network (pochi metri intorno alla persona, bluetooth)
- **LAN** da 10 m a 1 km (stanza, edificio, campus)
- **MAN** 10 km (città)
- **WAN** da 10 km a 10.000 km (regione, stato, intercontinentale)
- **Internet** (pianeta)

- Regola generale

- le reti piccole e localizzate tendono ad essere di tipo broadcast
- reti geograficamente disperse tendono ad essere punto a punto
- ma ci sono moltissime eccezioni



# Classificazione - distanza

| Interprocessor distance | Processors located in same | Example                   |
|-------------------------|----------------------------|---------------------------|
| 1 m                     | Square meter               | Personal area network     |
| 10 m                    | Room                       |                           |
| 100 m                   | Building                   |                           |
| 1 km                    | Campus                     | Local area network        |
| 10 km                   | City                       |                           |
| 100 km                  | Country                    | Metropolitan area network |
| 1000 km                 | Continent                  |                           |
| 10,000 km               | Planet                     | Wide area network         |
|                         |                            |                           |
|                         |                            | The Internet              |



# WAN



- Collega a lunga distanza macchine chiamate **hosts** oppure **LAN** su cui stanno gli host
- Sono connessi da una subnet di comunicazione che si divide in
  - Linee di trasmissione (rame, fibra, radio)
  - Elementi di commutazione (switching) spesso chiamati router
- Se due router non condividono una linea di trasmissione devono servirsi di router intermedi (store and forward o packet switched)

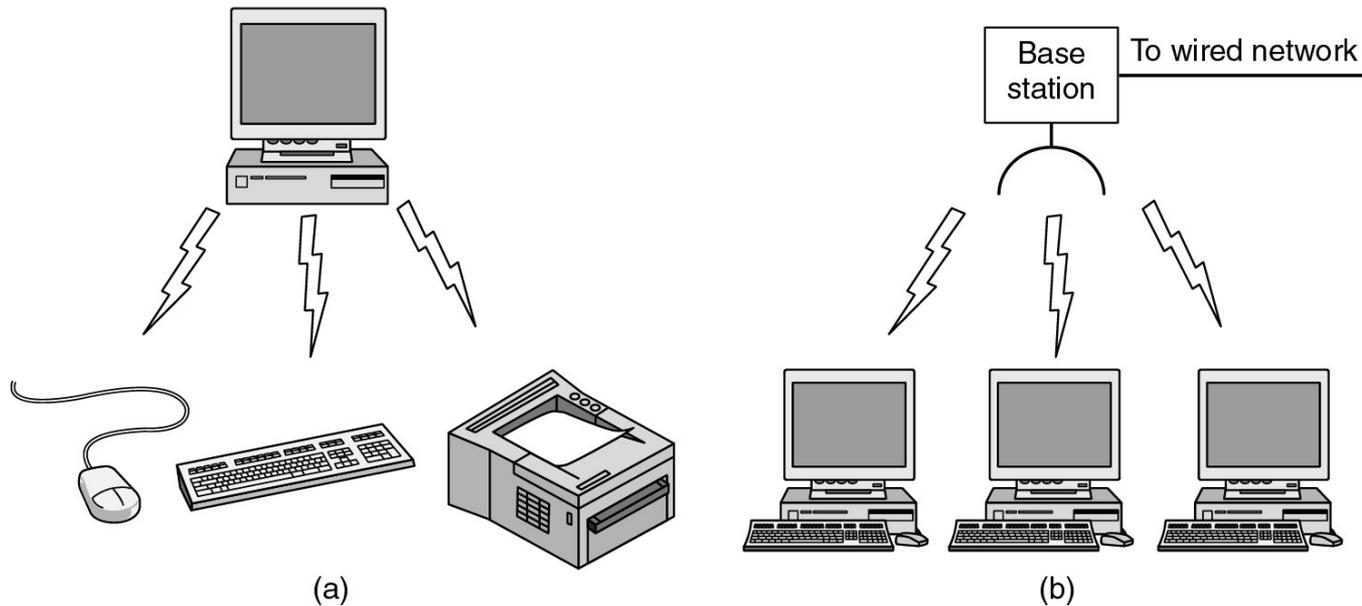


- Metropolitan Area Network

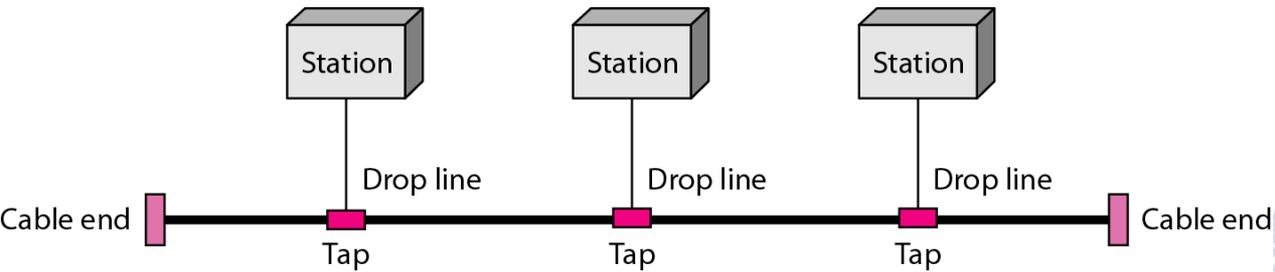
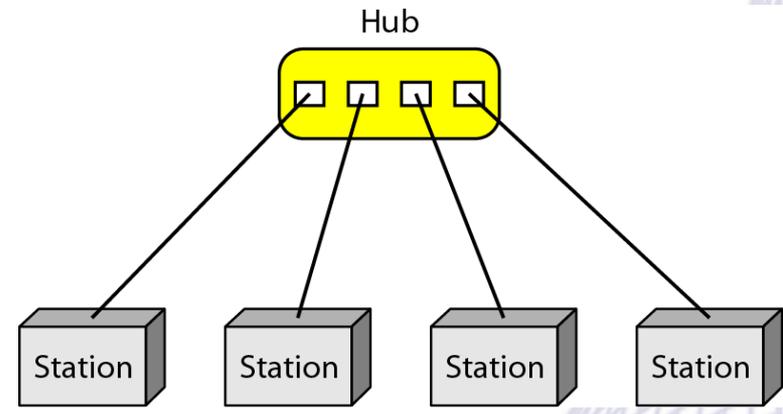
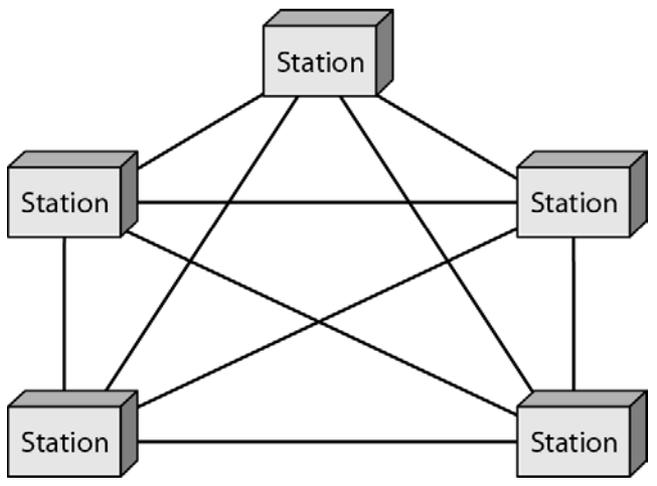
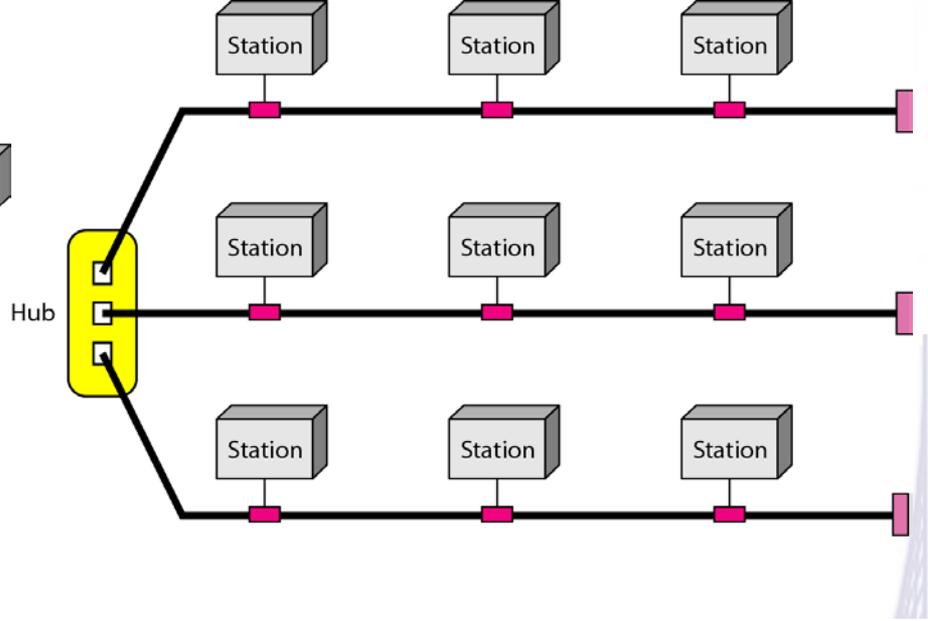
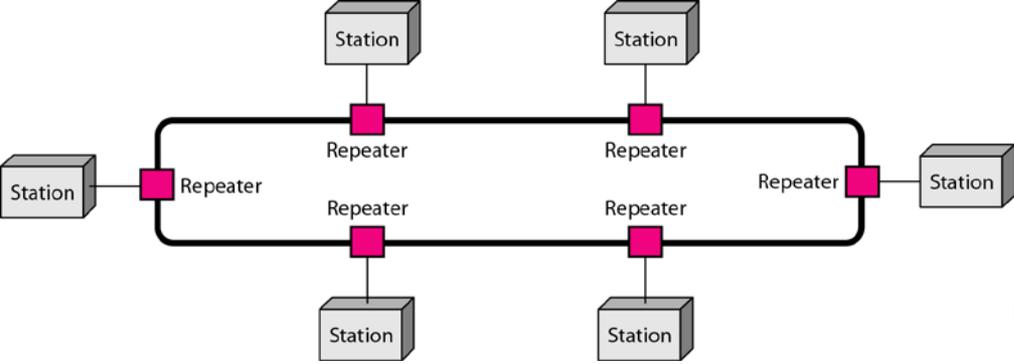
- Sta tra la LAN e la WAN, copre una città o una parte di essa
- Connette utenti fisicamente localizzati con altre reti (tipicamente internet)
- Es. La parte della rete telefonica che unisce le connessioni DSL
- Es. Collegamenti in fibra per unire ad alta banda tutte le LAN di una università cittadina ed altri enti



# Tipi di rete Wireless



- Interconnessioni tra dispositivi: Bluetooth, infrarosso, radio per tastiere, cellulari, palmari, stampanti, mouse, cuffie
- Wireless LAN: WiFi (802.11)
- Wireless WAN: e.g. Ethernet su Laser, Microonde, Radio, Satellite, UMTS, GPRS, GSM, WiMax (802.16)



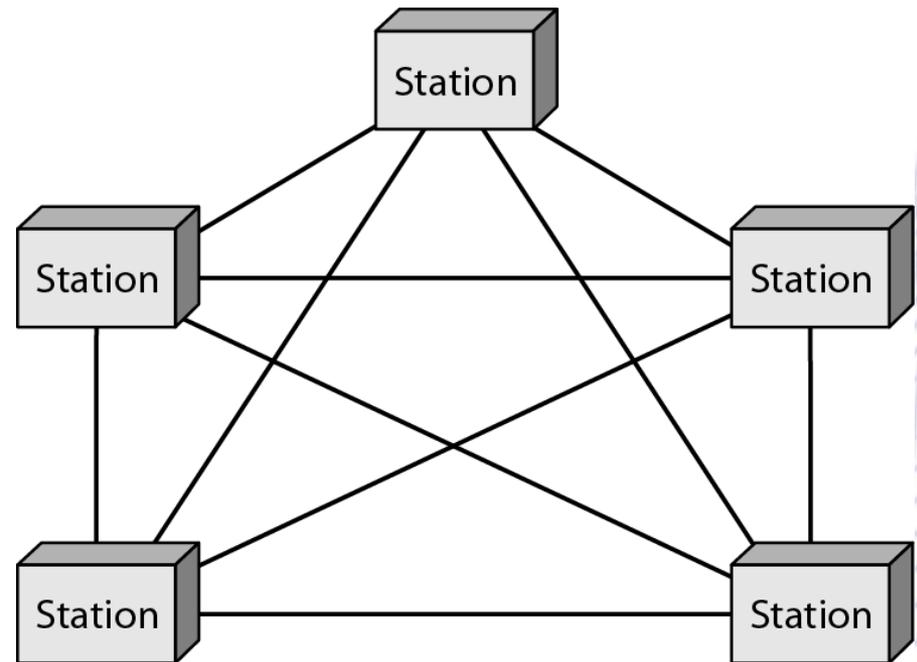


# Mesh



- Magliata

- I collegamenti necessari sono  $n * (n-1)$  quindi aumentano con il quadrato dei nodi
- Svantaggio economico (costo di link e di porte di I/O)
- Vantaggio: affidabile, sicura, permette di isolare parti guaste (nodi, link)
- Usata soprattutto per connettere centrali telefoniche o POP di reti IP



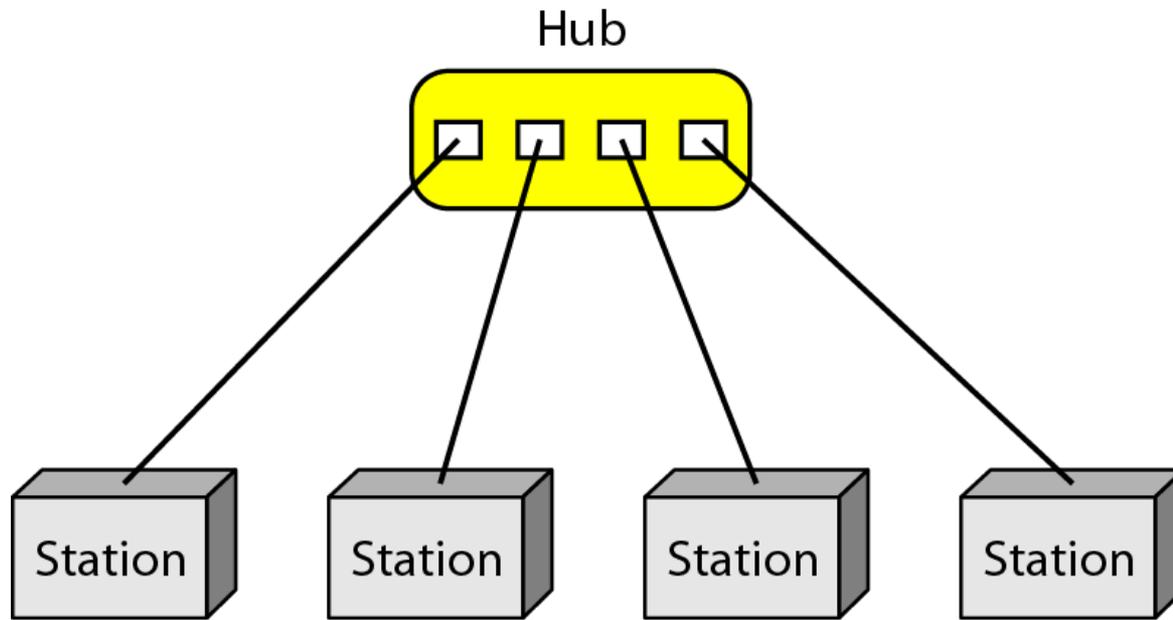


# Star



- Stella

- Connessa ad un Hub centrale
- Il traffico ora non può andare diretto da un nodo all'altro ma deve passare per l'hub.
- Molto semplice ed economico, un solo collegamento per nodo
- Svantaggio. Se l'hub si rompe nulla funziona

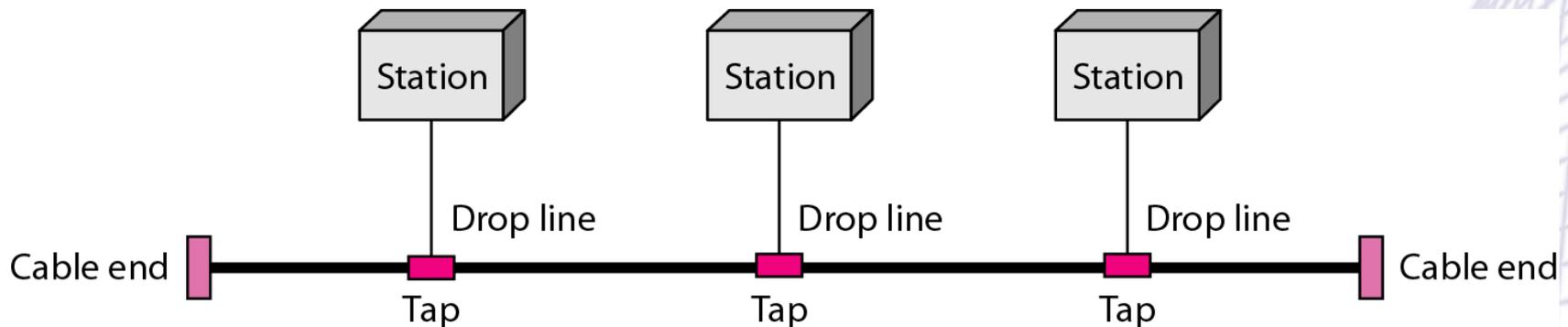




# Bus



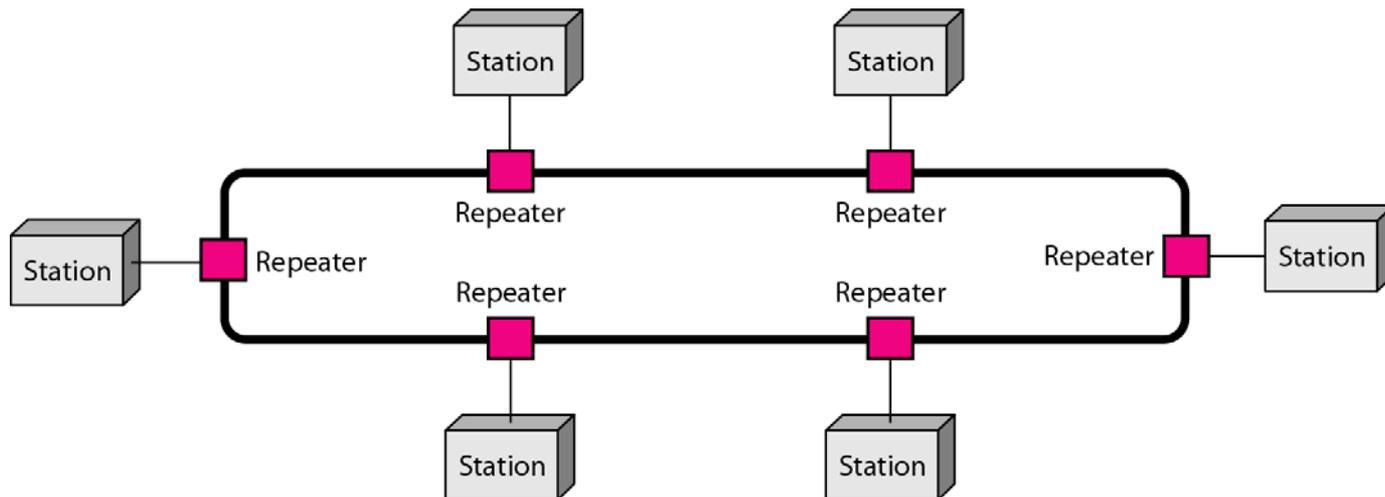
- Utilizza un collegamento multipunto (il bus) che collega tutti i nodi
- Ogni nodo è fisicamente collegato al bus, quindi quando un segnale passa lo sente ma il segnale diminuisce di intensità attraversando il bus. Questo limita il numero di nodi
- Facile inserire nuovi nodi, basta attaccare un nuovo connettore sul bus
- Svantaggi: Difficile risolvere problemi relativi al bus, es se i connettori non sono distanziati correttamente o se uno provoca rumore.





# Ring

- Anello
- Ogni nodo si collega punto punto con altri due nodi
- I dati viaggiano in una direzione e i nodi si passano il messaggio da uno all'altro fino alla destinazione
- Facile da installare (due collegamenti da cambiare per ogni inserimento o rimozione)
- Svantaggi: i dati devono fare percorsi lunghi e se un nodo non funziona interrompe l'anello

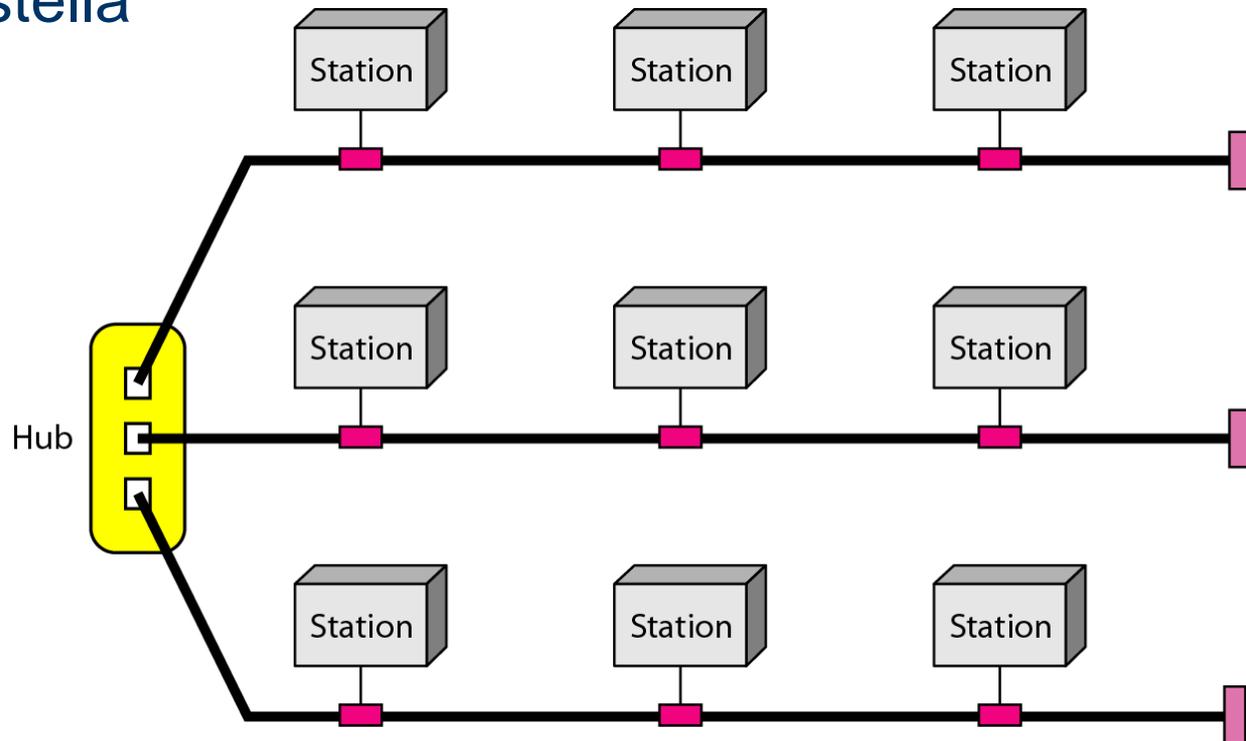




# Ibride



- Un mix delle topologie indicate
  - Es. un hub centrale unisce tre reti con topologia a bus, anello e stella





# Aspetti software

- Dal punto di vista software le reti sono organizzate a **layer** o **livelli**, costruiti uno sopra l'altro.
- Il numero di livelli e le funzioni di un livello cambiano da rete a rete
- Un layer è come una macchina virtuale che offre servizi al layer superiore

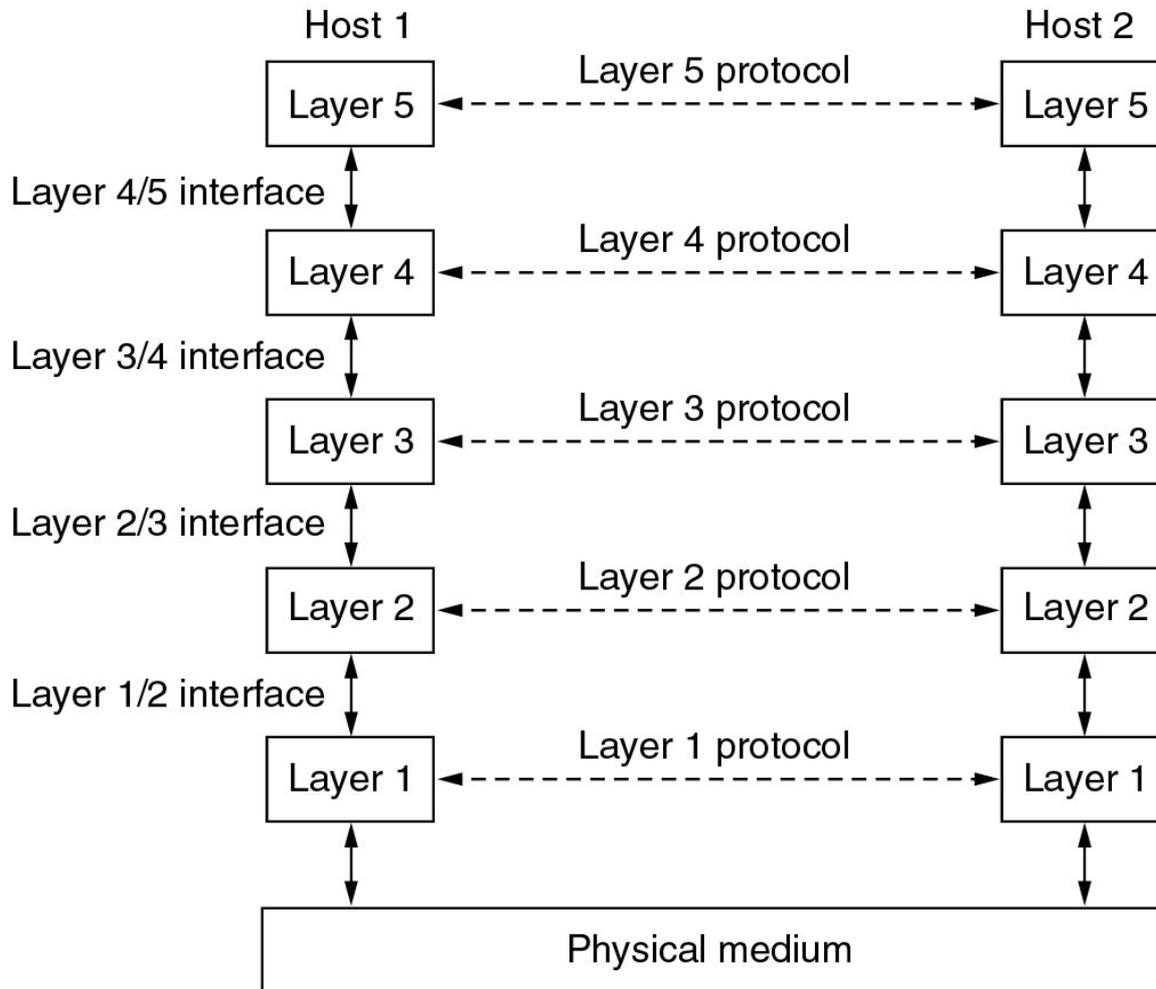


# Aspetti software

- Ogni livello fornisce servizi ai livelli superiori schermando questi livelli dai dettagli su **come** i servizi sono in pratica implementati
- Tipico concetto della computer science: information hiding, abstract data types, data encapsulation, oo programming



# Layer di protocolli





# Aspetti software

- Il livello **N** di una macchina parla con il livello **N** di un'altra macchina. Le regole e le convenzioni usata in questa conversazione sono dette **protocollo** di livello **N**
- È un **accordo** su come procedere nella **comunicazione**
- Cfr: protocolli “sociali”. Ci sono regole diverse: come si fa un baciamento ad una principessa, come si stringe la mano ad un avvocato.



# Protocollo tra persone

Locazione A Messaggio in Russo

Locazione A tradotto dal Russo in Inglese

Locazione A mandato via fax

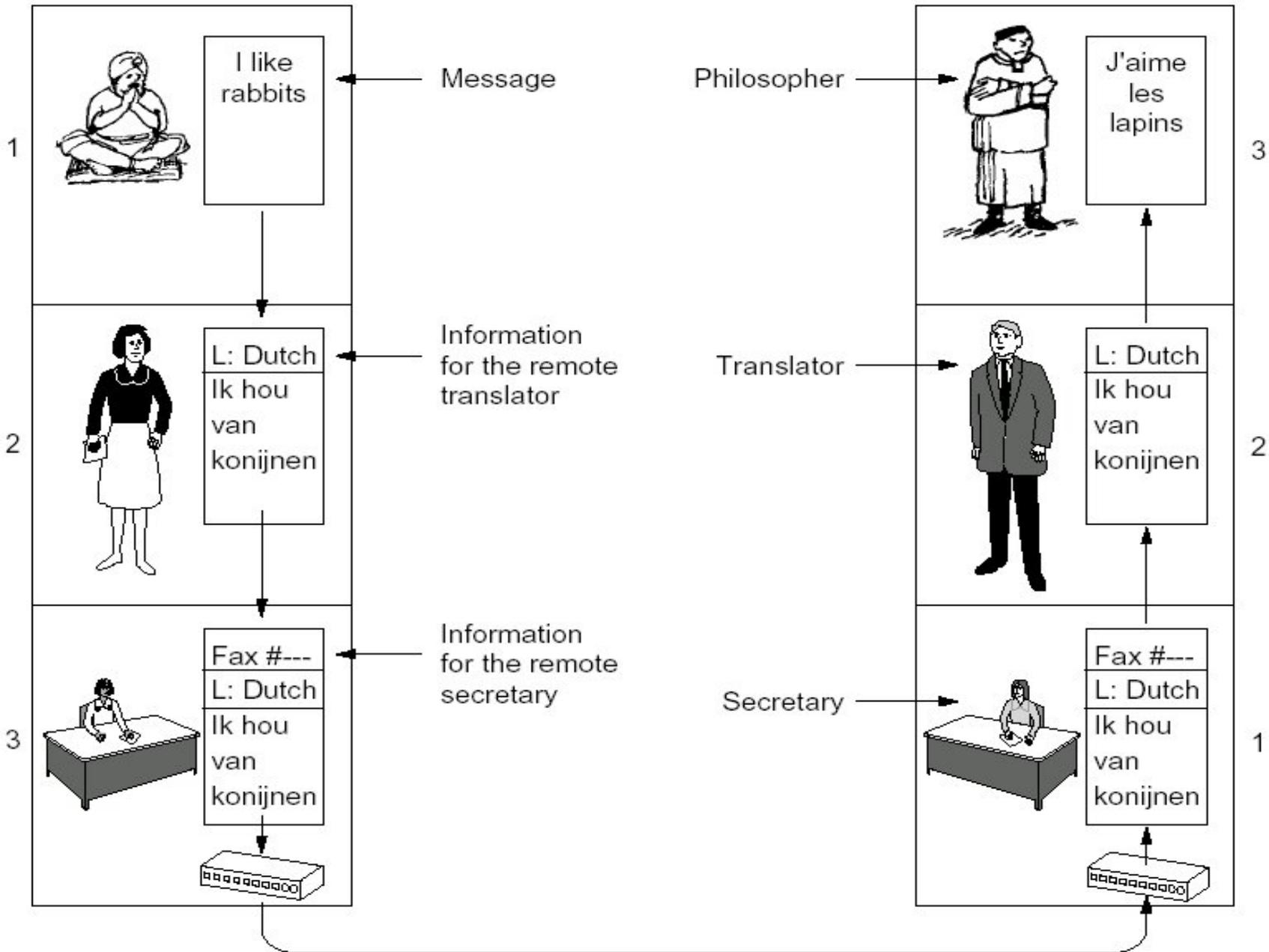
Locazione B ricevuto via fax

Locazione B tradotto dall'Inglese in Italiano

Locazione B Messaggio in Italiano

Location A

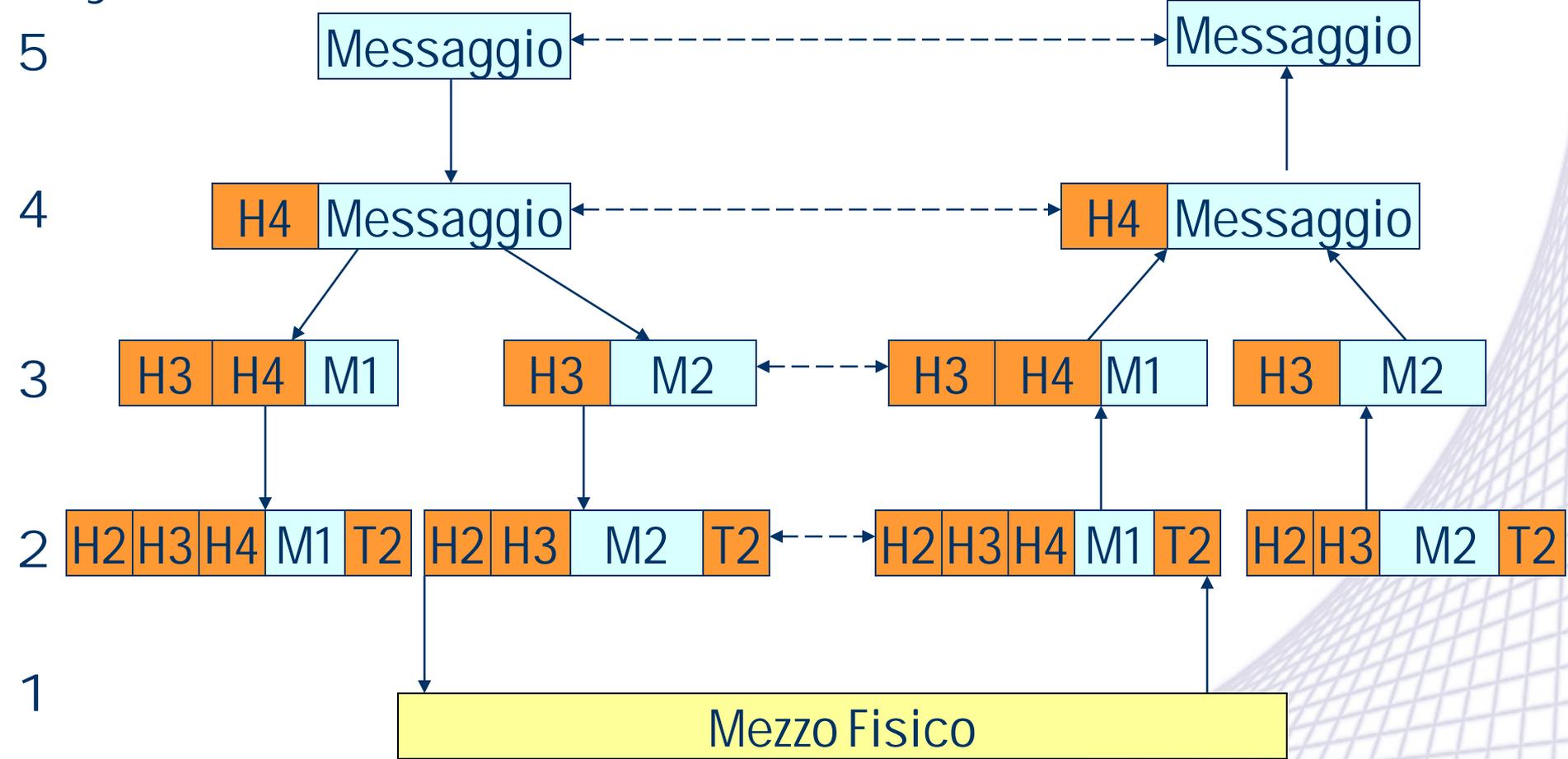
Location B





# Protocollo tra computer

Layer





# Protocollo tra computer

---

- La comunicazione a livello **logico** avviene tra pari (**PEER**) allo stesso layer – linee tratteggiate
- A livello **reale** la comunicazione avviene solo al livello inferiore. (linee continue)
- L'astrazione dei processi PEER permette di **separare un problema molto difficile** (il progetto dell'intera rete) in **problemi semplici** (il progetto di un layer)



# Interfaccia



- Tra ogni coppia di layer adiacenti è definita una interfaccia
- L'interfaccia definisce quali operazioni “primitive” e quali servizi il livello inferiore fornisce al superiore



# Elementi chiave

- Le reti comunicano utilizzando le caratteristiche fisiche della rete per spedire sequenze di bit
- Le sequenze di bit devono avere lo stesso significato per entrambi i nodi per cui servono delle regole di:
- **SINTASSI**: Il formato dei dati, l'ordine con cui gli elementi devono essere presentati
  - Es. Primi 8bit source address, 8 bit destination address poi i dati
- **SEMANTICA**: Il significato della sequenza di bit
  - Come interpretare una sequenza di bit, che azione eseguire in risposta
- **SINCRONIZZAZIONE**: quando i dati devono essere spediti e a quale velocità
  - Una sorgente a 100 Mbps ma un ricevente a 1 Mbps



# Progetto di un protocollo

- Quanti layer devo mettere?
- Cosa deve fare ogni layer?
- Come avere una interfaccia semplice?
  - Ogni layer deve fare un insieme ben definito di funzioni
  - Una interfaccia semplice minimizza le informazioni da passare tra i layer
  - Permette facilmente di rimpiazzare un layer con una implementazione completamente diversa.



# Indirizzamento



- Addressing:
  - Ogni layer deve identificare mittente e ricevente
  - Ogni rete ha diversi computer ognuno con molti processi. Un processo in una macchina deve sapere come specificare con quale vuole parlare dell'altra macchina
  - Serve quindi una forma di indirizzamento per specificare una specifica destinazione

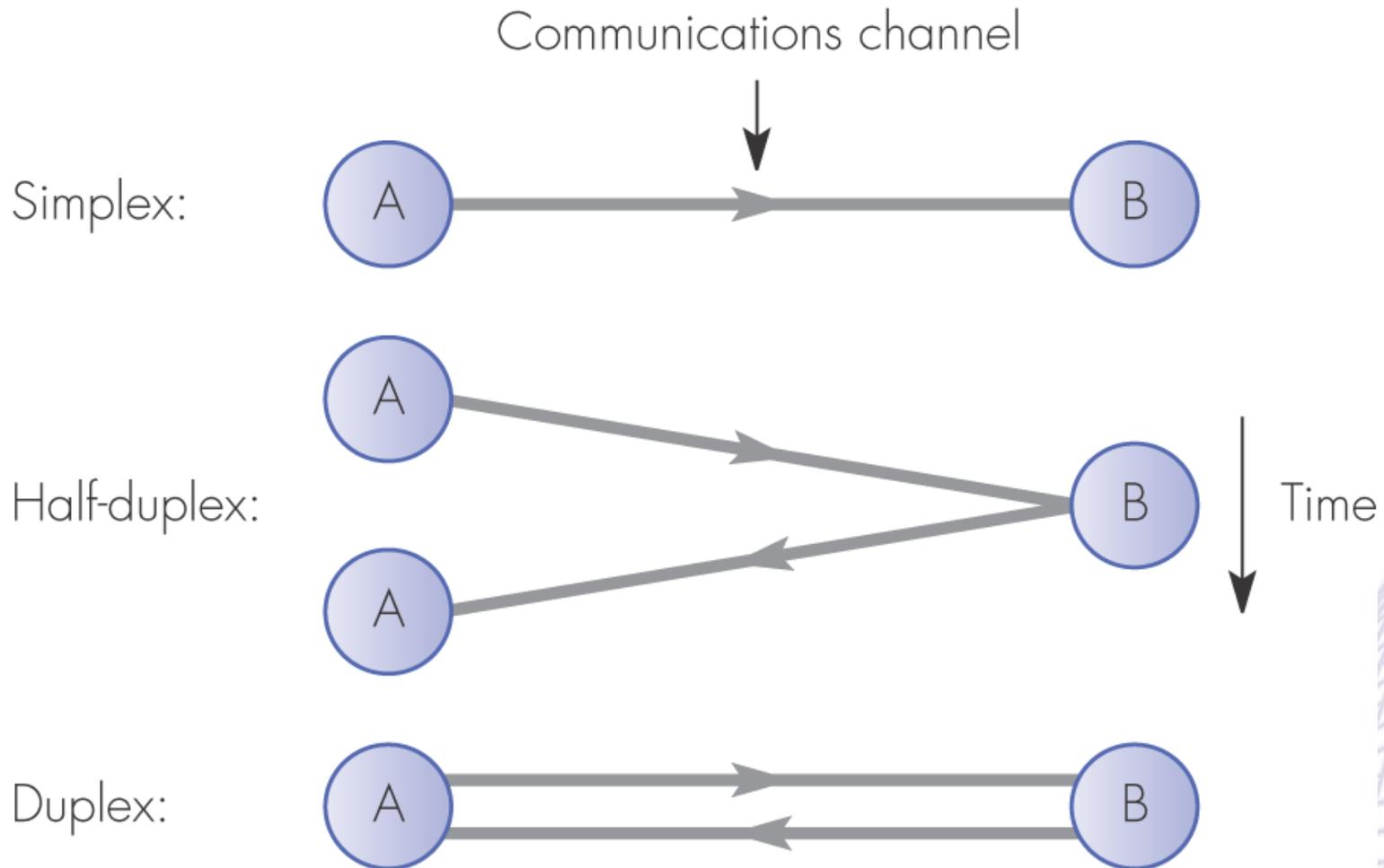


# Data Transfer

- Direzione dei dati (slide seguente)
  - **Simplex**: I dati vanno in una unica direzione
  - **Half Duplex**: I dati vanno in entrambe le direzioni ma non contemporaneamente
  - **Full Duplex**: I dati viaggiano in entrambi le direzioni allo stesso istante
- Quanti canali logici sullo stesso canale fisico
  - Spesso ci sono almeno due canali, uno per i dati normali e uno per i dati urgenti (o per la gestione dei dati)



# Modi di comunicazione





# Error control

- Controllo degli errori
  - I circuiti di comunicazione non sono perfetti
  - Ci sono diversi metodi di **error correction** e **error detection** ma entrambe le parti devono concordare quale usare e poi serve un modo per dire al mittente cosa non è arrivato bene



# Data order

- Ordine dei messaggi
  - Se il canale non garantisce l'ordine dei messaggi il protocollo deve assegnare un numero sequenziale per permettere il riassettaggio
  - Poi resta da decidere cosa fare dei messaggi fuori sequenza



# Flow control

- Flow Control

- Come evitare che un trasmittente veloce intasi un ricevente lento
- Qualche forma di feedback (implicito o esplicito) sulla situazione del ricevente
- Contrattazione tra i due di un transmission rate adeguato



# Lunghezza messaggi



- A vari layer si deve decidere la lunghezza dei messaggi da elaborare
  - Spesso serve un metodo per disassemblare, trasmettere e riassemblare messaggi
  - Al contrario potrebbe essere inefficiente mandare messaggi troppo corti: serve allora un meccanismo per raggrupparli, spedirli alla comune destinazione e dividere di nuovo i messaggi alla fine.



# Mux - DeMux

- Multiplex – Demultiplex:
  - Quando non conviene stabilire una connessione separata per ogni coppia di processi.
  - Il layer inferiore decide di usare la stessa connessione per conversazioni non correlate
  - A qualsiasi layer basta che sia trasparente al layer superiore.
  - Necessario a livello fisico

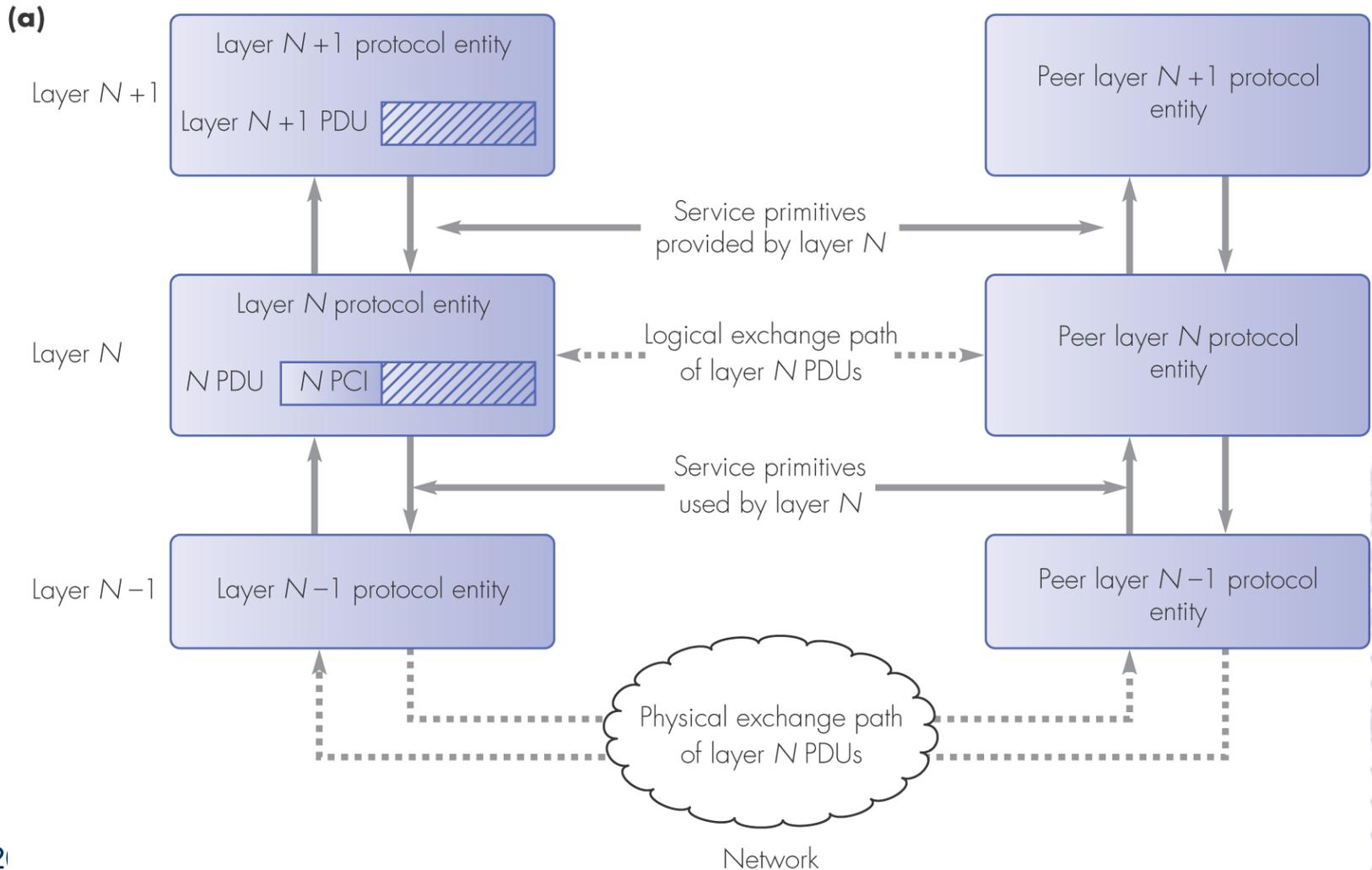


# Interfacce e Servizi

- Gli elementi attivi di un layer si chiamano **entities**
  - Possono essere software (processi) o hardware (un chip di I/O intelligente)
- Entities di livello **n** implementano un servizio per il livello **n+1**
  - Livello **n** → **Service provider**
  - Livello **n+1** → **Service user**
- I servizi sono disponibili ai **SAP (Service Access Points)**.
  - I SAP del livello **n** sono i posti in cui il livello **n+1** accede ai servizi offerti. Ogni SAP ha un indirizzo che lo identifica



# Livelli e servizi





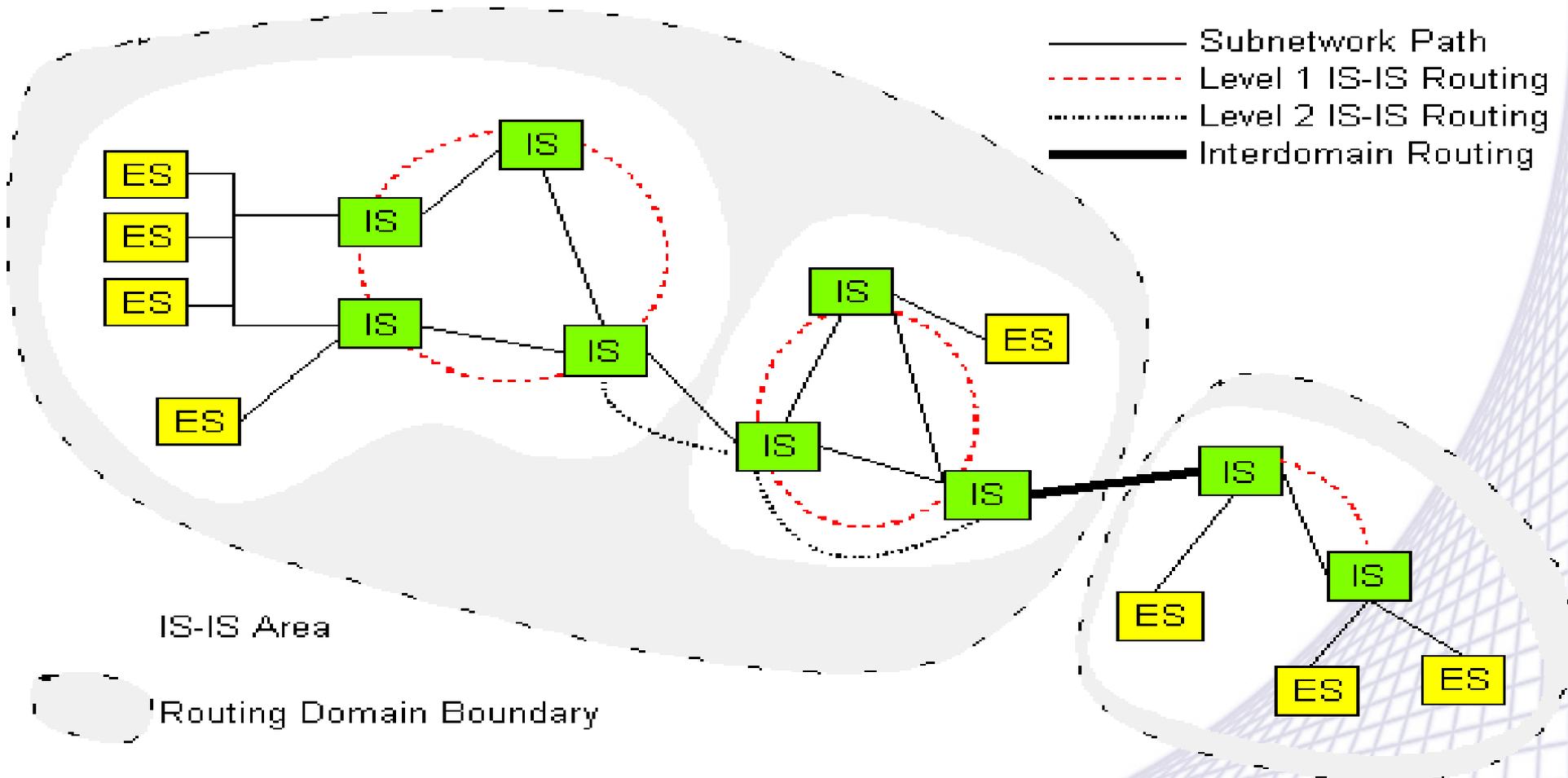
# Routing



- Se ci sono diversi cammini tra mittente e destinatario se ne deve scegliere uno
  - A volte una decisione viene presa a livello alto e cambiata a livelli inferiori basandosi p.e. sulle condizioni di traffico



# Routing



IS-IS Area

Routing Domain Boundary



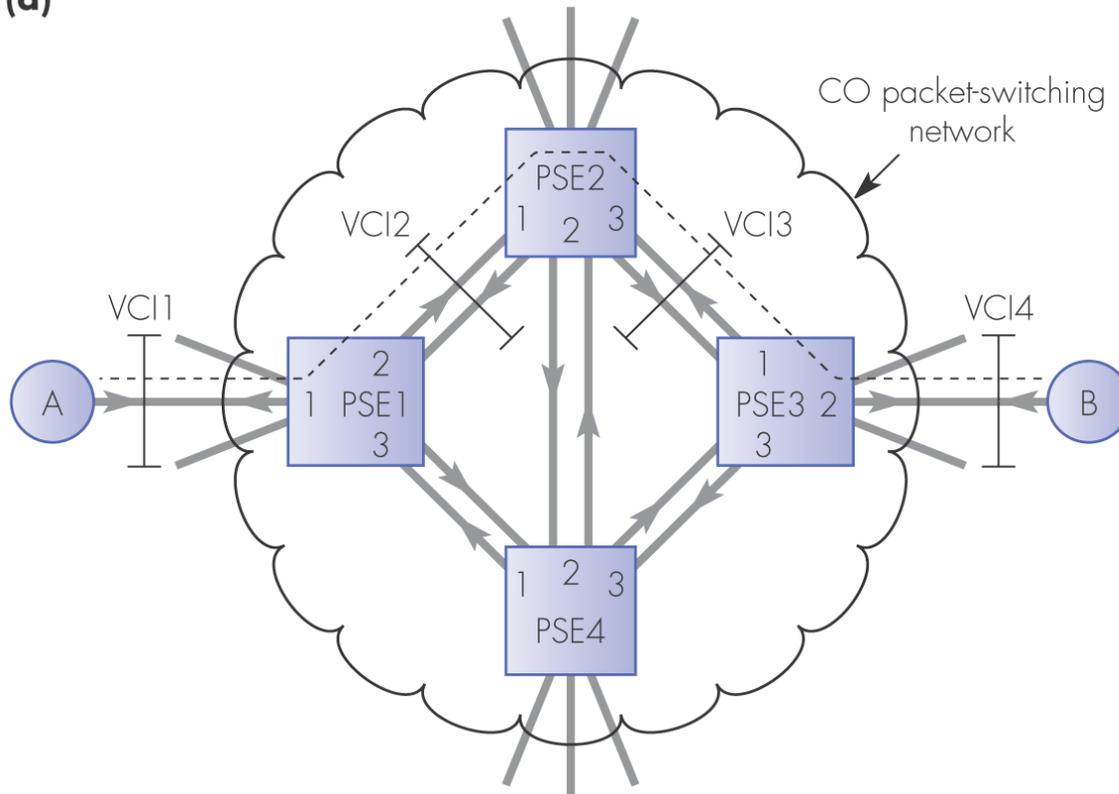
# Servizi e connessioni

- I layers possono offrire due tipi di servizi al layer superiore
- Servizi Connection-Oriented
  - Come il sistema telefonico: Alzi la cornetta, fai il numero, stabilisci la connessione, attacchi la cornetta. Funziona come un tubo. Si mandano gli oggetti (bits) nel tubo e solitamente gli oggetti escono in ordine all'altro lato
- Servizi Connectionless
  - Come il sistema postale: Ogni messaggio ha l'indirizzo completo del destinatario. Ogni messaggio viaggia per la sua strada. Di due messaggi allo stesso destinatario uno potrebbe non arrivare e o arrivare in gran ritardo



# Connection Oriented

(a)



PSE1 routing table:

| IN         | OUT          |
|------------|--------------|
| VCI1/Link1 | → VCI2/Link2 |
| VCI2/Link2 | → VCI1/Link1 |

PSE2 routing table:

|            |              |
|------------|--------------|
| VCI2/Link1 | → VCI3/Link3 |
| VCI3/Link3 | → VCI2/Link1 |

PSE3 routing table:

|            |              |
|------------|--------------|
| VCI3/Link1 | → VCI4/Link2 |
| VCI4/Link2 | → VCI3/Link1 |

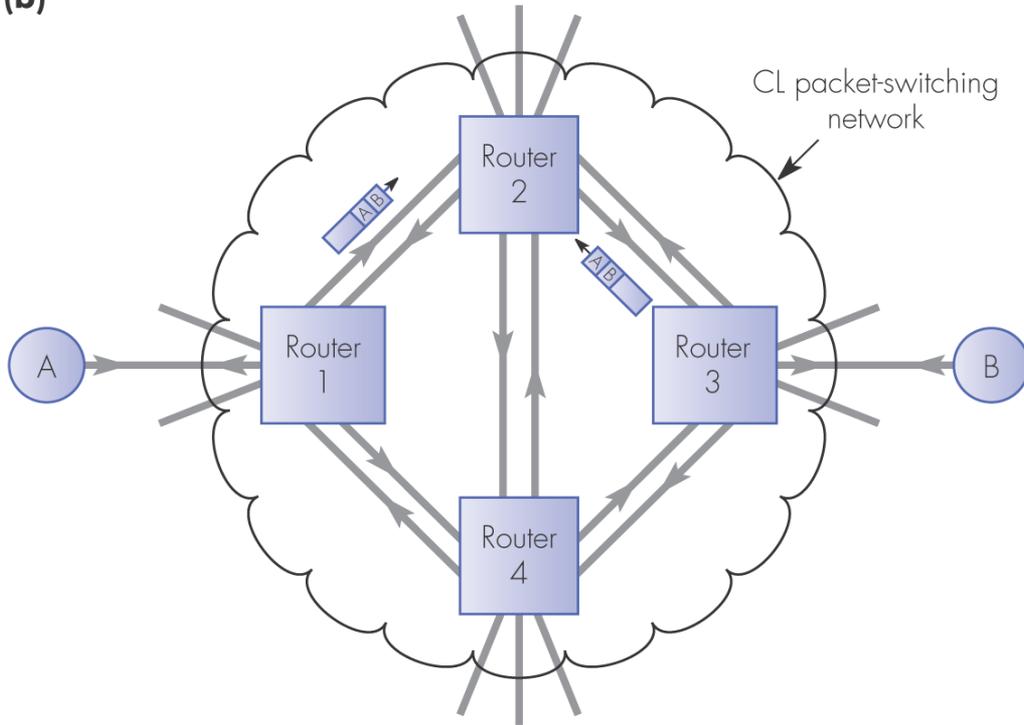
CO = connection-oriented  
 --- = virtual circuit

VCI = virtual circuit identifier  
 PSE = packet-switching exchange

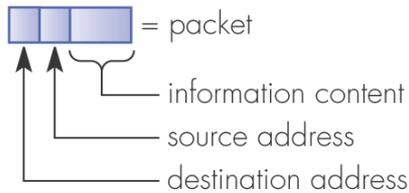


# Connectionless

(b)



CL = connectionless  
A, B = full network-wide addresses





# Tipi di servizi

|                     | <b>Service</b>          | <b>Example</b>       |
|---------------------|-------------------------|----------------------|
| Connection-oriented | Reliable message stream | Sequence of pages    |
|                     | Reliable byte stream    | Remote login         |
|                     | Unreliable connection   | Digitized voice      |
| Connection-less     | Unreliable datagram     | Electronic junk mail |
|                     | Acknowledged datagram   | Registered mail      |
|                     | Request-reply           | Database query       |



# Conn. Oriented affidabile



- Servizi affidabili (reliable)
  - Nel senso che non perdono i dati
  - Si implementano obbligando il ricevente a dare una ricevuta (**acknowledgement**) per ogni messaggio ricevuto.
  - La gestione della ricevuta introduce **overhead** e ritardi (**delays**)
  - Usato nei File Transfer



# Due varianti

- **Message Sequences:**
  - Le dimensioni dei messaggi sono preservate: due messaggi di 1K arrivano come due distinti messaggi da 1K e non come uno da 2K
- **Byte Streams:**
  - Non c'è modo di sapere se c'è una separazione nel flusso dei dati (es. Login in un server remoto)



# Conn. Oriented non affidabile



- Connection Oriented senza ricevuta
  - Una telefonata via cellulare avviene dopo una connessione
  - Ma non richiedo ack per ogni pacchetto voce in uno dei due sensi
  - Il setup della connessione mi garantisce che ci sono le risorse per portare a buon fine la telefonata



# Servizi connectionless



- Servizi connectionless “non affidabili” (datagram services)
  - Es.: Streaming Voce (non importa se sento rumori strani o manca mezza parola, ma non voglio ritardi) o video digitali (alcuni pixel sbagliati ok, ma non fermi immagine)
  - Come i Telegrammi e la posta ordinaria
  - Voglio mandare un messaggio con alta probabilità che sia consegnato ma non con la garanzia.
  - “Non affidabili” significa solo che non c’è la ricevuta



# Servizi connectionless affidabili



- Servizi connectionless con ricevuta
  - A volte conviene non stabilire una connessione (per non perdere tempo) ma importa invece la affidabilità
  - Acknowledged Datagram service: come la Raccomanda con ricevuta alla risposta.



# Request - reply

- **Servizi Connectionless Request – Reply**
  - Il mittente manda un datagram con una richiesta e il reply contiene la risposta.
  - Es Richiesta risoluzione indirizzo DNS



# Chi vuole servizi unreliable?



- Chi potrebbe volere dei servizi non affidabili?
- Quando non sono disponibili servizi affidabili
  - La comunicazione affidabili potrebbe non essere possibile (es. Ethernet). I pacchetti potrebbero rovinarsi durante la trasmissione
  - I protocolli superiori eventualmente gestiranno il problema
- Quando i ritardi sarebbero non accettabili
  - Applicazioni real-time o multimediali.



# Operazioni di accesso

- Un servizio è definito da un insieme di operazioni (primitives) disponibili ad un processo utente che vuole accedere al servizio
- Se lo stack del protocollo è in un sistema operativo le primitives sono chiamate di sistema (system calls)
- L'insieme di primitives è diverso per servizi connection oriented da servizi connection-less



# Reliable byte stream

| Primitives | Significato                          |
|------------|--------------------------------------|
| LISTEN     | Aspetta una richiesta di connessione |
| CONNECT    | Connetti ad un peer in attesa        |
| RECEIVE    | Aspetta un messaggio in arrivo       |
| SEND       | Manda un messaggio al peer           |
| DISCONNECT | Chiudi la connessione                |

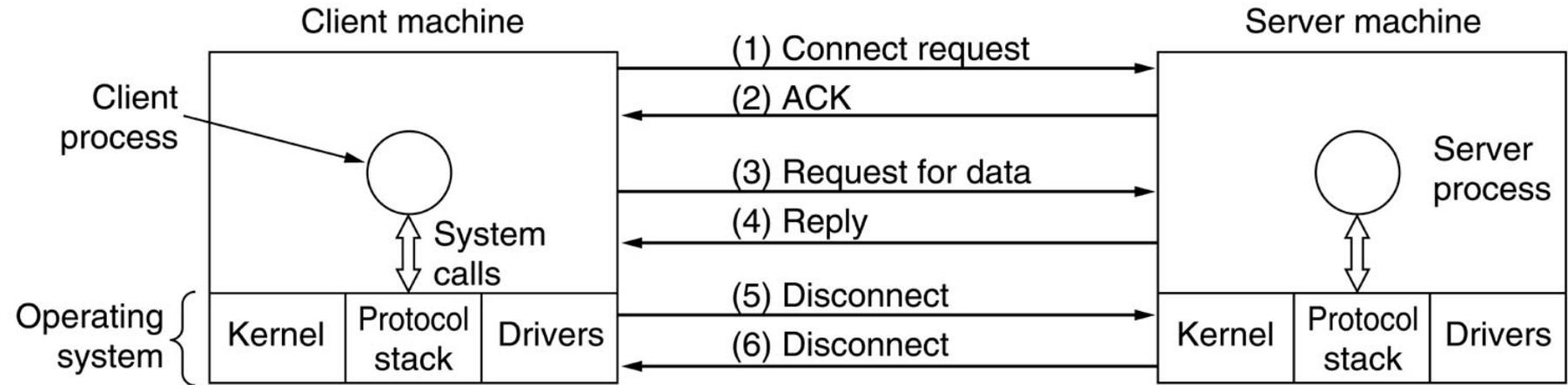


# Come le uso?

- 1 Il server esegue la LISTEN (per esempio con una system call bloccante). Il processo server rimane in attesa
- 2 Il client esegue una CONNECT (avrà bisogno di un parametro per passare l'indirizzo del server)
- 3 Il sistema operativo manda un pacchetto (1) al peer chiedendo la connessione e resta sospeso in attesa di una risposta



# Connessione



- I numeri sono i pacchetti che viaggiano
- In questo caso 6 pacchetti



# La connessione

- 4 Quando il pacchetto (1) arriva al server viene gestito dal sistema operativo che vedendo un pacchetto di “richiesta connessione” controlla se c’è un processo in ascolto
- 5 Se c’è, sblocca il processo in ascolto e manda indietro un acknowledgement (2)
- 6 L’arrivo dell’acknowledgement sblocca il client. A questo punto la connessione è stabilita.



# Dopo la connessione



- 7 Il server esegue immediatamente una RECEIVE bloccando il processo server
- 8 Il client con una SEND fa la sua richiesta (pacchetto 3) e poi subito una RECEIVE per aspettare la replica
- 9 L'arrivo del pacchetto (3) di richiesta sblocca il processo server che può gestirla. Dopo l'elaborazione della richiesta il server usa SEND per mandare la risposta (4) al client
- 10 L'arrivo del pacchetto (4) sblocca il client



# Dopo la connessione

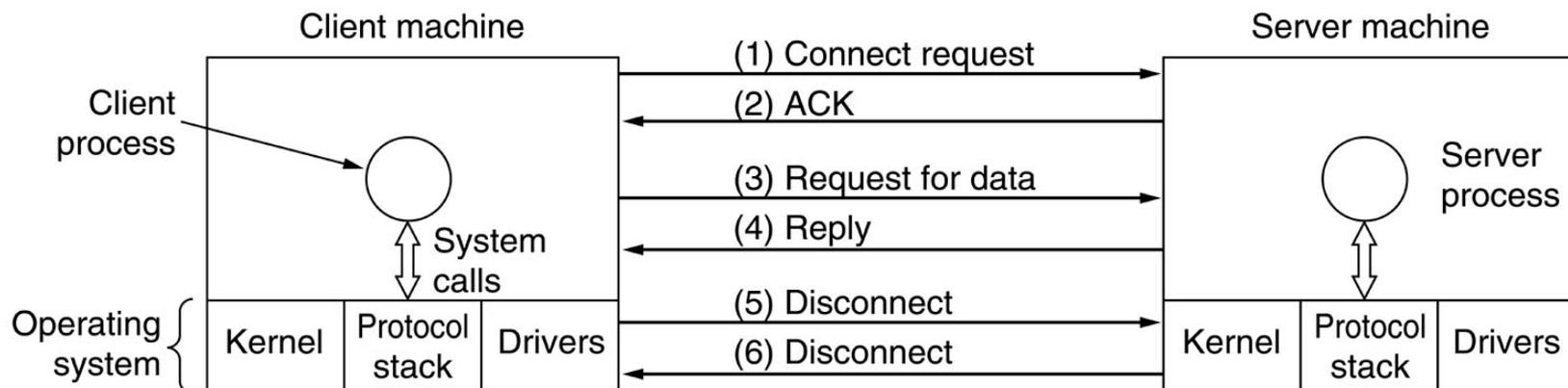


- 11 Se il client ha altre richieste le può fare oppure manda un DISCONNECT (5). Solitamente di tipo blocking
- 12 Il server riceve il pacchetto (5) e poi manda una DISCONNECT (6) e chiude la connessione.
- 13 Il client riceve il pacchetto (6) e sblocca il processo client



# Cosa può andare storto

- Cosa può andare storto?
  1. Timing sbagliato.
    - Es. CONNECT eseguita prima della LISTEN
  2. Pacchetti che si perdono





# E connectionless?

- Nell'esempio appena visto abbiamo usato **sei** pacchetti
- Se avessimo usato l'approccio **connectionless** sarebbero stati sufficienti solo **due** pacchetti!
- Buona soluzione in un mondo ideale. Ma invece di solito si usa connection oriented soprattutto in caso di:
  - **grandi messaggi in una delle due direzioni** (quindi i due pacchetti sono una minima parte del totale)
  - Linea soggetta a frequenti **errori di trasmissione**
  - Possibilità elevata di **perdita di pacchetti**
- Infatti sarebbe consigliabile non fare economia di pacchetti se vogliamo garantirci contro i problemi citati



# Servizi e Protocolli



- Non vanno confusi i due concetti:
  - **Servizi:** L'insieme di operazioni che un livello fornisce al livello superiore. Il servizio definisce **quali** operazioni il layer fornisce ai suoi utenti ma non specifica per nulla **come** queste operazioni sono implementare
  - **Protocolli:** Le regole che decidono il formato e il significato dei pacchetti e messaggi che si scambiano le "peer entities" all'interno del livello. Le entities possono liberamente cambiare il loro protocollo ma non possono cambiare i servizi visibili agli utenti

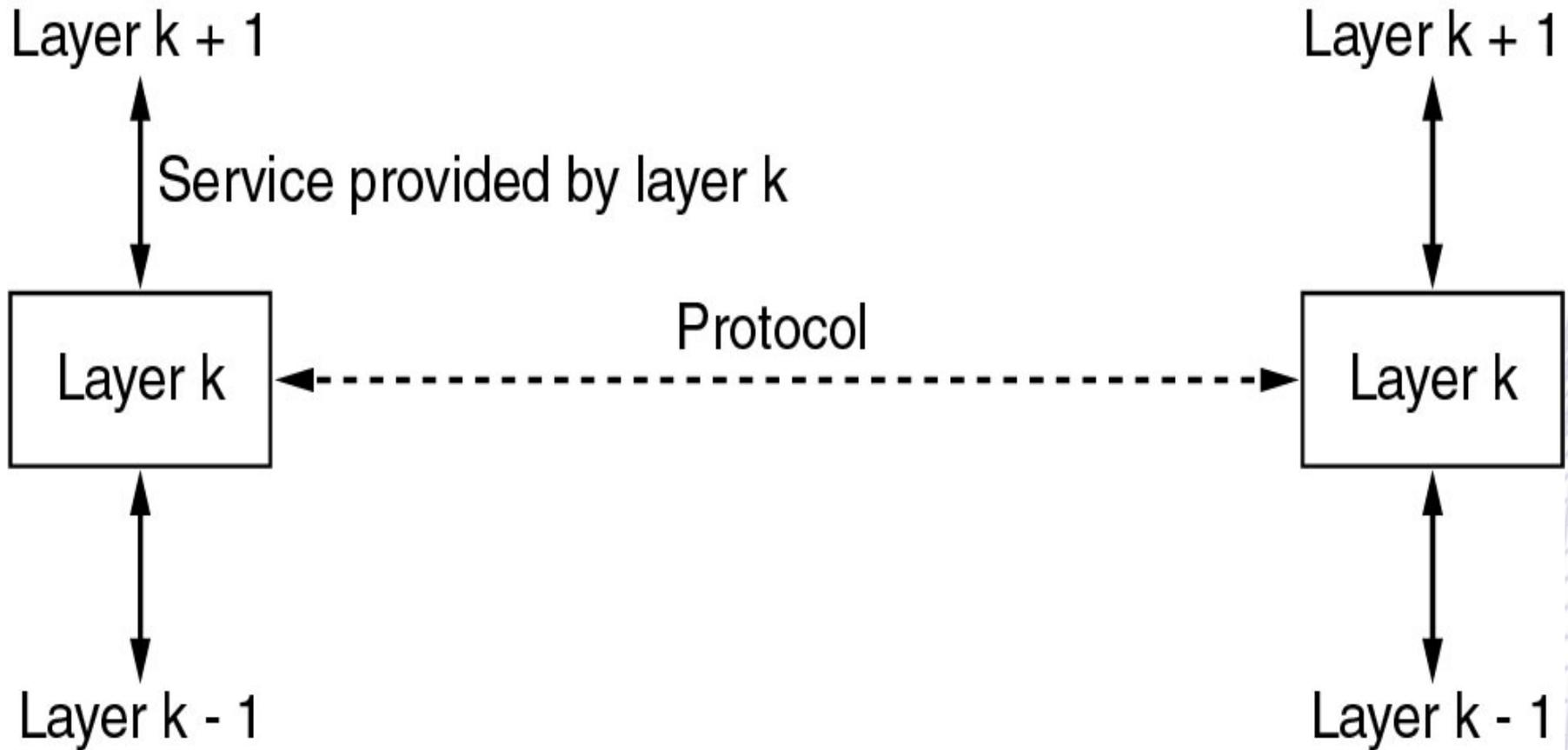


# Servizi e Protocolli

- I servizi si riferiscono alle interfacce tra i layers
- I protocolli invece si riferiscono ai pacchetti che si mandano le peer entities nelle diverse macchine
- È come la differenza tra un abstract data type (o un oggetto) che specifica **quali** sono le operazioni possibili ma non **come** vengono implementate, mentre il protocollo specifica l'implementazione del servizio che non è visibile all'utente del servizio



# Relazione servizi - protocolli





# Ripasso matematica

- Le potenze di due

- 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024

- $2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}$

- $2^{10}=1\text{K}, 2^{16}=65536=64\text{K}$

- $2^{24}=16\text{M}, 2^{32}=4\text{G}$

- $2^{10} * 2^{10} = 2^{10+10} = 2^{20}$

- $2 * 2^{10} = 2^{10+1} = 2^{11}$



# Ripasso matematica

- Le conversioni di base
  - $1111_2 = F_{16} = 0xF = 15_{10}$
  - $1111\ 1111_2 = 0xFF = 255_{10}$
- Utili per le maschere degli indirizzi IP
  - $224_{10} = 1110\ 0000$
  - $252_{10} = 1111\ 1100$
  - $7_{10} = 0000\ 0111$



# Ripasso matematica



## ● Logaritmi

- $\text{Log}_{10} 1\ 000 = 3$
- $\text{Log}_{10} 10\ 000\ 000 = 7$
- $\text{Log}_2 4096 = 12$
- $\text{Log}_2 8 = 3$



# Ripasso Fisica

- Velocità = spazio/tempo
- Tempo = spazio/velocità
- Spazio = velocità \* tempo
- Velocità luce nel vuoto =  $3 * 10^8 \text{ ms}^{-1}$
- Velocità segnale in un cavo =  $2/3$  di  $3 * 10^8 \text{ ms}^{-1}$



# Esempio Fisica

- Quanto ci mette un pacchetto per muoversi in un cavo lungo 1km?
- $10^3\text{m} / (2 \cdot 10^8\text{m/s}) = 5 \cdot 10^{-6}\text{s} = 5 \mu\text{s}$
- Quanta strada percorre un fotone in un millisecondo?
- $S = c \cdot t = 3 \cdot 10^8\text{m/s} \cdot 10^{-3}\text{s} = 3 \cdot 10^5\text{m} = 300\text{ km}$
- E in 1 ns?
- $S = 3 \cdot 10^8\text{m/s} \cdot 10^{-9}\text{s} = 3 \cdot 10^{-1}\text{m} = 30\text{ cm}$



# Tempi vs Hertz

- 1 Hz = 1 volta al secondo =  $1 \text{ s}^{-1}$
- 1 kHz = 1000 volte al secondo =  $10^{-3} \text{ s}^{-1}$  quindi clock di 1 ms
- 1 MHz =  $10^6$  volte al secondo =  $10^{-6} \text{ s}^{-1}$  quindi clock di 1  $\mu\text{s}$
- 1 GHz =  $10^9$  volte al secondo =  $10^{-9} \text{ s}^{-1}$  quindi clock di 1 ns



# Prestazioni

- Introduciamo la terminologia per definire senza ambiguità le prestazioni
- Bandwidth o Larghezza di banda
  - Unità di misura?
- Throughput
  - A quale velocità riesco a spedire i dati
- Latenza o Delay
  - Quanto tempo ci mette un messaggio per arrivare a destinazione (tempo di propagazione + tempo di trasmissione + tempo di attesa + tempo di inoltra)



# Bandwidth

- Due tipi di larghezza di banda – **Bandwidth**
- In Herz
  - Vedremo che rappresenta l'intervallo di frequenze contenute in un segnale
- In bit per secondo
  - Indica la velocità con cui possiamo spedire i bit sul canale (o sulla rete, utilizzando diversi canali)
- Relazione tra le due.
  - Un aumento della banda in Herz mi permette di aumentare la banda in bit/sec, dipende se trasmetto il segnale di base o un segnale modulato



# Throughput

- Il Throughput **T** a prima vista equivale alla banda **B**
- Invece pur avendo un canale con banda **B** possiamo trasmettere solo **T bps** perché non riusciamo ad utilizzare tutta la banda
- Viene influenzato anche dalla capacità dei nodi di smistare i dati ma anche dal traffico della rete
- Es Ho una rete con **B=1000 Mbps** ma riesco a trasferire in media solo **120000 frame al minuto** e ogni frame contiene in media **10kbit**. Calcolare **T**
- $T = (120000 \text{ fr/m} \times 10000 \text{ bit/fr}) / 60 \text{ s} = 20 \text{ Mbps}$  quindi ho solo 1/50 della banda disponibile



# Delay



- Latenza o Ritardo o Delay misura il tempo perché l'intero messaggio arrivi a destinazione
- Somma di quattro componenti
  - Propagation Delay (Tempo di Propagazione)
  - Transmission Delay (Tempo di Trasmissione)
  - Queueing Delay (Tempo di Accodamento)
  - Forwarding Delay (Tempo di Inoltro)



# Tempo di propagazione

- Misura il tempo necessario al segnale per viaggiare da mittente a destinatario
- Distanza / Velocità di propagazione del segnale
- Es. Calcolare tempo di propagazione tra due punti distanti 12000 km. Considerare la velocità di propagazione di  $2.4 * 10^8$  m/s
- $T_{prop} = (12 * 10^6 \text{ m}) / 2.4 * 10^8 \text{ m/s} = 50\text{ms}$
- Quindi 1 bit ci mette circa 50 millisecondi ad attraversare l'Atlantico



# Tempo di trasmissione

- Se spediamo tanti bit (un messaggio), il primo arriva dopo un tempo pari al tempo di propagazione.
- Non dobbiamo aspettare che il primo arrivi per spedirli tutti, li spediamo tutti in sequenza
- Il tempo per mettere tutti i bit sul canale è detto tempo di trasmissione = dimensione del messaggio / larghezza di banda
- Es Calcolare tempi di propagazione e trasmissione di un email di 2.5 KB in una rete a 1Gbps, assumendo distanza 12000 km e velocità di propagazione  $2.4 * 10^8$  m/s
- $T_{prop}$  calcolato nella slide precedente **50 ms**
- $T_{trasm} = 2500 * 8 \text{ b} / 10^9 \text{ bps} = 20 * 10^3 / 10^9 = 0.02 \text{ ms}$



# Queueing e Forwarding

- Il tempo di attesa nelle code dei nodi intermedi (o comunque in quello di destinazione)
  - Dipende dal carico delle rete. Un dispositivo intermedio, per esempio un router, deve accodare i messaggi per poterli inoltrare. Se ci sono molti messaggi in attesa ogni messaggio avrà un  $T_Q$  delay molto elevato
- Tempo di forwarding.
  - $T_F$  Dipende dalla velocità di elaborazione del router (che deve per esempio fare bit error checking e trovare la porta di uscita). Non dipende dal traffico della rete ma dalle caratteristiche hw dei nodi intermedi



# Queueing delay

- $R$ =link bandwidth (bps)
- $L$ =packet length (bits)
- $a$ =average packet arrival rate

Intensità del traffico =  $La/R$

- $La/R \sim 0$ : **delay medio è basso**
- $La/R \rightarrow 1$ : **delay diventa grande**
- $La/R > 1$ : arrivano più pacchetti di quanti possano essere elaborati, **delay medio infinito!**

