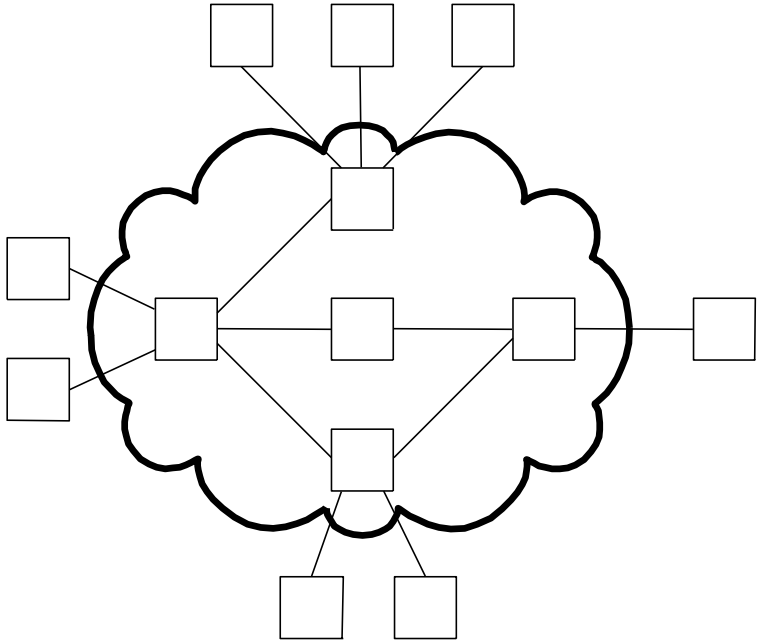
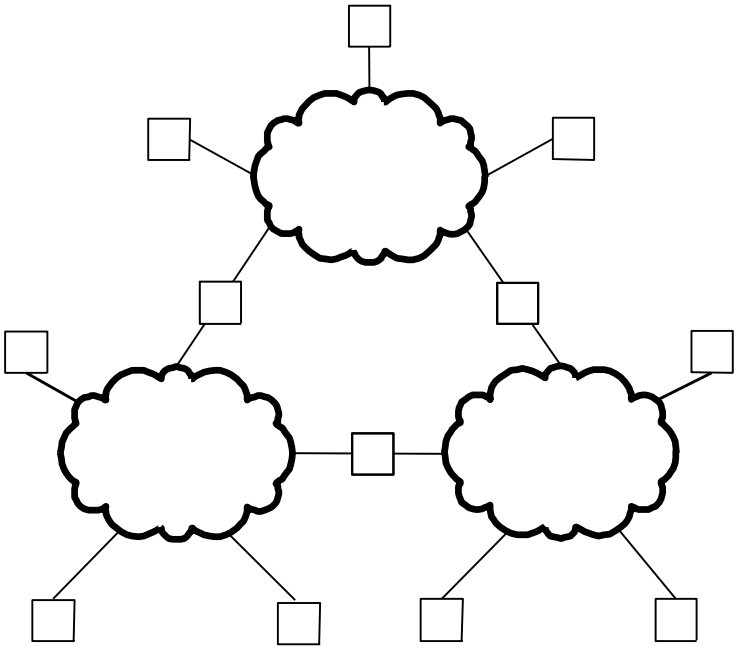
PARTE 1

INTRODUZIONE A RETI E PROTOCOLLI

Reti (*def.*)

- Una rete può essere definita ricorsivamente
  - Due o più **nodi** connessi tramite **collegamenti**
  - Due o più **reti** connesse tramite **nodi**





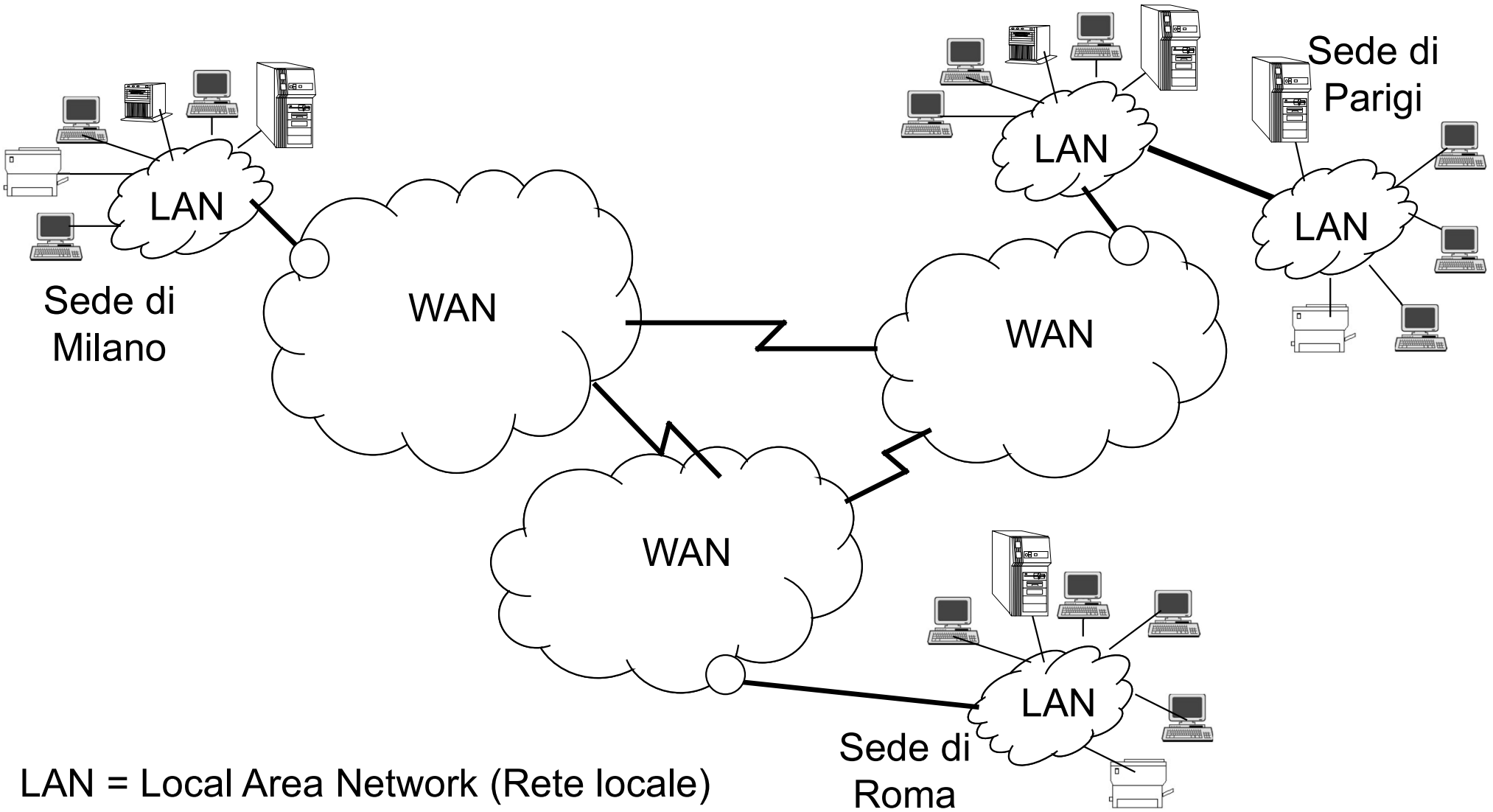
Componenti fondamentali di una rete

- **NODI (NODE)**
  - **Host**
    - termine astratto per identificare un **nodo terminale** connesso a uno dei capi della comunicazione (mittente, destinatario)
  - **Switch, Bridge, Router, …**
    - Esempi di **nodi intermedi** che abilitano la comunicazione fra host. Termini differenti indicano ruoli differenti
- COLLEGAMENTO (LINK)
  - **Cablato (Wired)**: Cavi fibra ottica, coassiali, …
  - **Non cablato (Wireless)**: WiFi, Bluetooth, …

Astrazione di Reti

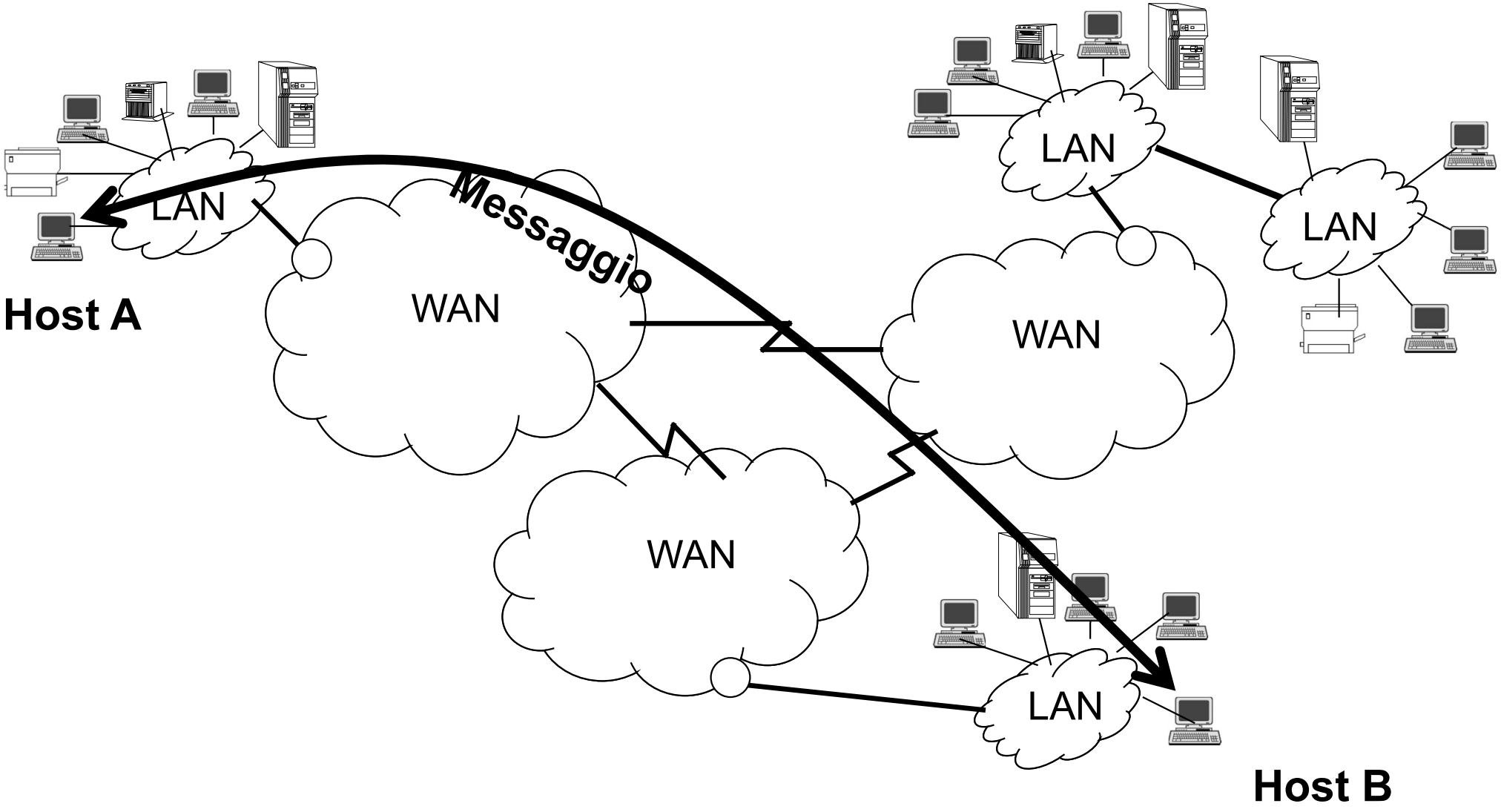
- La definizione ricorsiva permette di utilizzare paradigmi di **astrazione**
  - Nascondere dettagli non necessari per focalizzarsi sulle funzionalità di interesse
- L'approccio ricorsivo può essere applicato più volte

Esempio



LAN = Local Area Network (Rete locale)
WAN = Wide Area Network (Rete geograifca)

Comunicazione logica tra due host

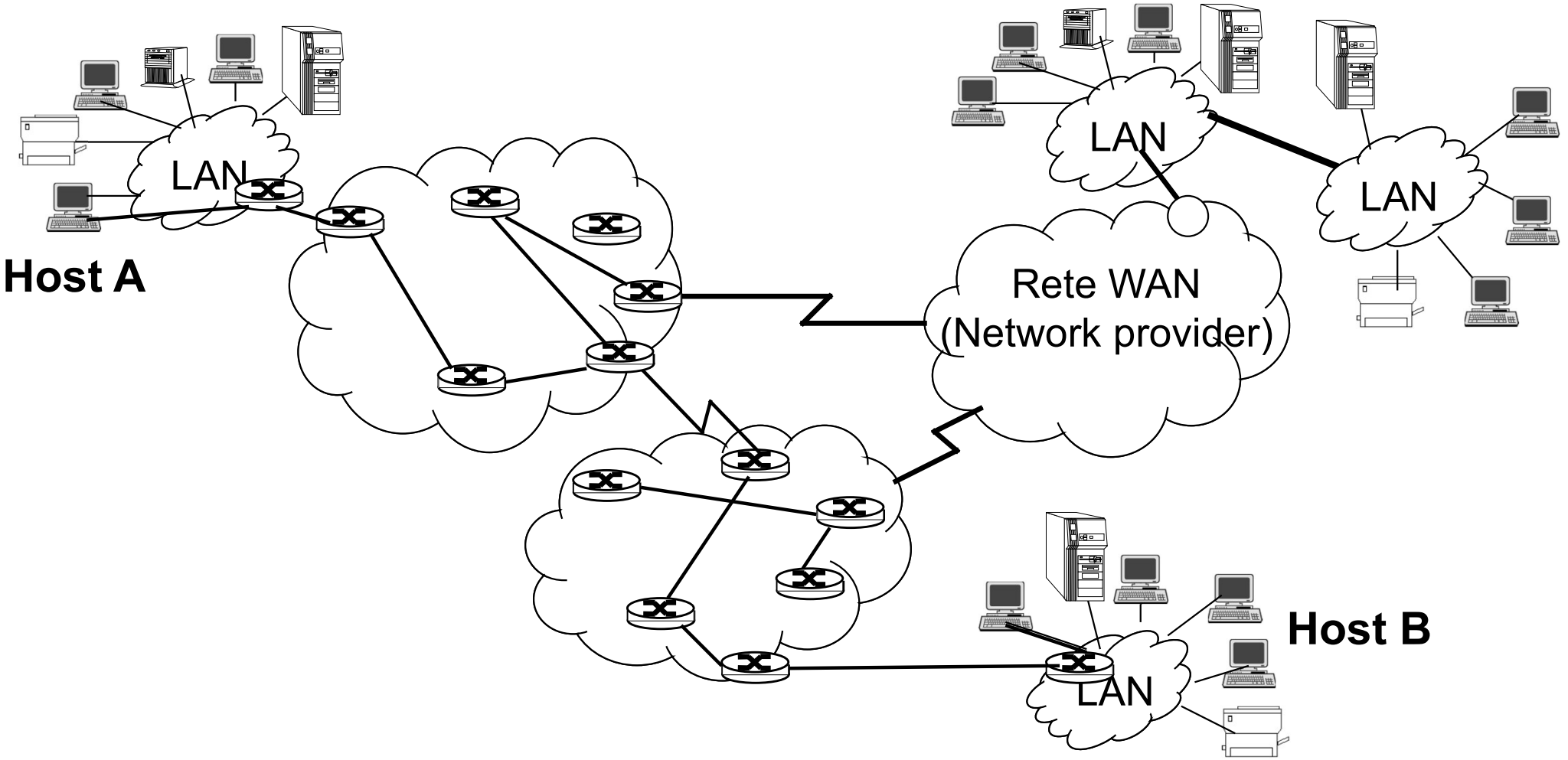


Logicamente comunicano i due host terminali

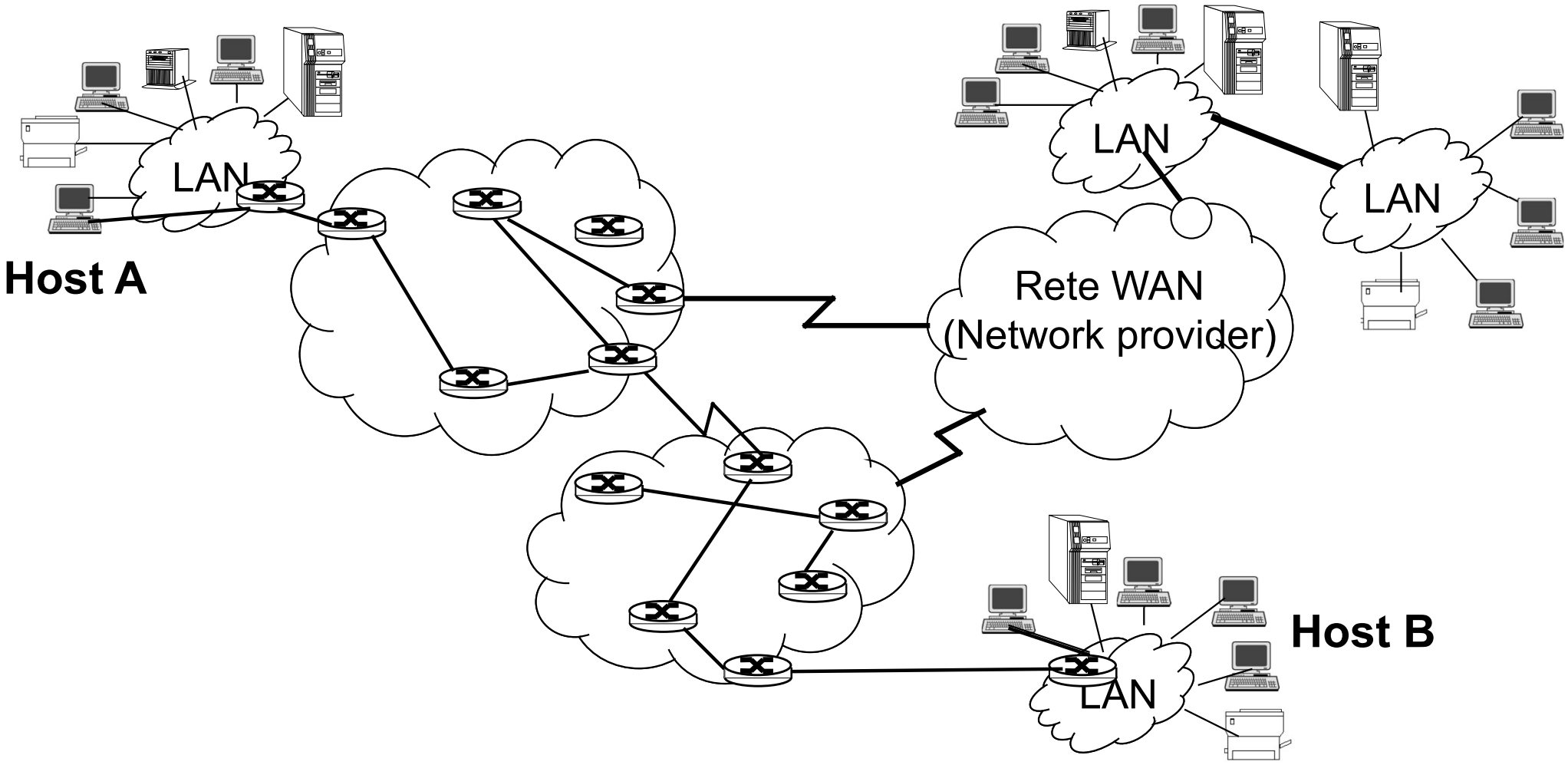
Comunicazione reale

In realtà, le informazioni attraversano tutti i nodi e i collegamenti

- Problemi di ***instradamento*** e di ***condivisione delle risorse***



Comunicazione reale



OBIETTIVO → Trasferire un Messaggio (*insieme di bit*) da un host all'altro

Sembra banale. DOV'E' IL PROBLEMA?

Alcune caratteristiche fondamentali

- Host, nodi e collegamenti sono **eterogenei**
- La rete **può cambiare** nel tempo (nelle tecnologie, negli scopi, …)
- Compromessi per ottenere i massimi benefici
  - Costi vs **Prestazioni** vs **Affidabilità** vs Sicurezza
  - La rete (collegamenti, nodi) è **condivisa**

Estrema eterogeneità

- Quali caratteristiche hw/sw ha l'host?
- Come è interconnesso l'host?
- Quale mezzo trasmissivo utilizza?
- Quale modalità di trasmissione del messaggio (=insieme di bit)?
- Come si gestisce il transito dei messaggi attraverso i nodi intermedi?
- Di quali servizi può usufruire l'utente?
- …

Cosa può non funzionare?

- Interferenze elettriche (errori a livello di bit)
- Congestioni (errori a livello di messaggi)
- Guasti di link e di nodi intermedi
- Problemi software di nodi mittente/destinazione
- …

- Ritardi nei messaggi
- Consegna dei messaggi fuori ordine
- “Ascolto” dei messaggi da parte di terzi
- …

Perché le reti tra computer? (cambiamenti nel tempo)

- Collegamenti remoti a mainframe (< anni ’70)
- Informatica distribuita *vs.* informatica monolitica dei *mainframe* (anni ’70)
- Comunicazioni tra utenti (anni ’80)
- “The network is the computer” (anni ’90)
- The network is ubiquitous (anni ’10+)

Protocollo

- La comunicazione tra entità richiede cooperazione, ossia collaborazione per il conseguimento di uno scopo comune. **Tutte le comunicazioni sono regolate mediante protocolli.**

- ***Protocollo*: insieme di regole e convenzioni seguite da entità, dislocate su nodi distinti, che intendono comunicare per svolgere un compito comune**

- Tali regole hanno l'obiettivo di assicurare una cooperazione efficiente ed affidabile per la comunicazione tra nodi e per la realizzazione di servizi di rete, tenendo conto delle caratteristiche tipiche di un sistema distribuito (banda di trasmissione limitata, ritardi variabili, errori nella comunicazione, …)

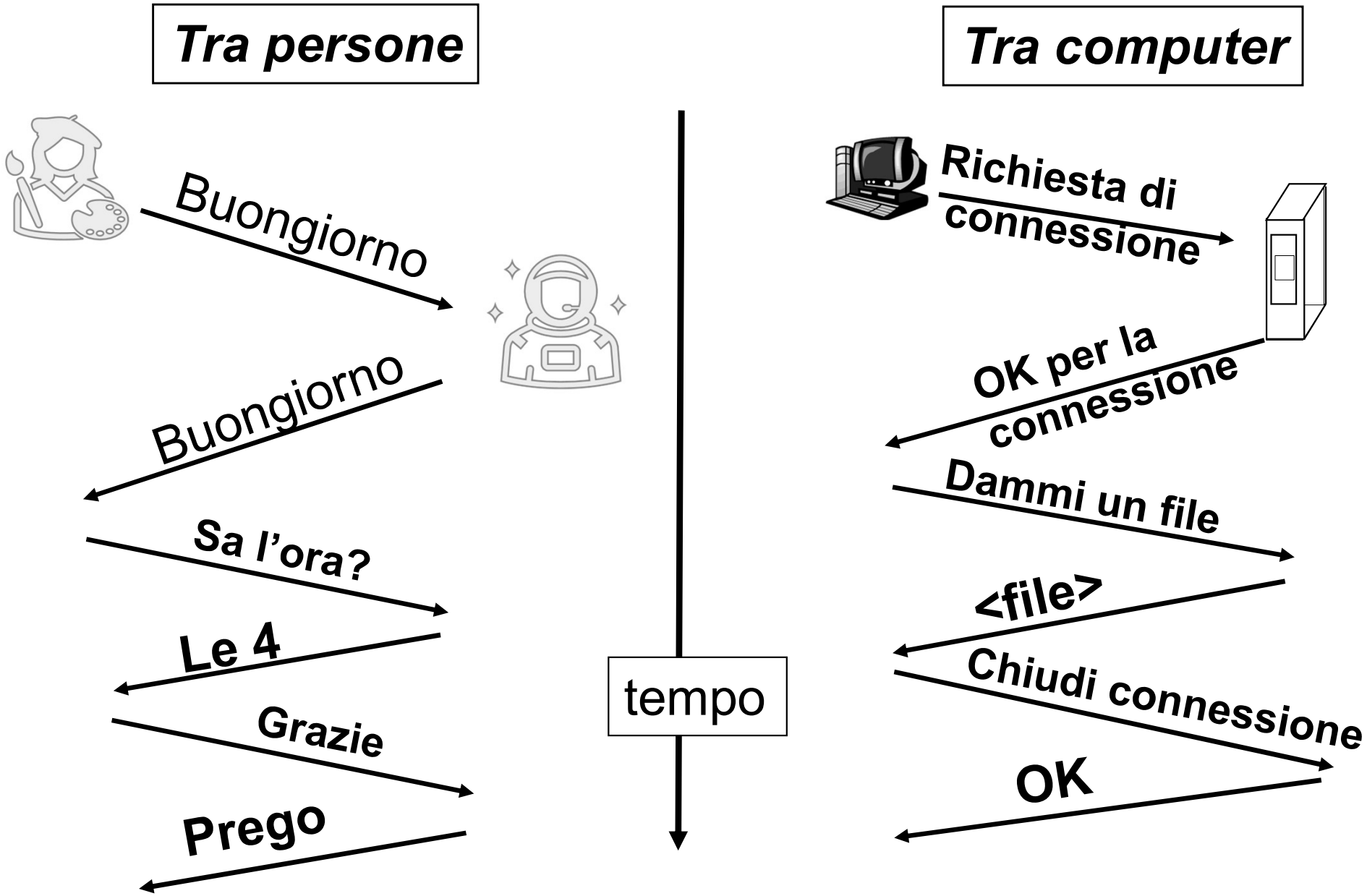
Elementi di un protocollo di comunicazione

- ***Sintassi***: insieme e struttura dei comandi e delle risposte, formato dei messaggi

- ***Semantica***: significato dei comandi, delle azioni, delle risposte da effettuare al momento della trasmissione e ricezione dei messaggi

- ***Temporizzazione***: specifica delle possibili sequenze temporali di emissione dei comandi e dei messaggi, nonché delle eventuali risposte

Esempio di protocollo tipico



Le 4 domande fondamentali

1. **Architettura hardware**
   - "Quale hardware esegue il protocollo? Quale hardware consente di comunicare? Dove viaggia l'informazione?")

2. **Schema di naming / identificazione**
   - "Come si identificano gli interlocutori?"

3. **Architettura software**
   - "Quali software esegue il protocollo?"

4. **Schema di comunicazione**
   - "Quale paradigma di comunicazione si utilizza?"

Veri obiettivi

- Trasferimento di un **messaggio** (*insieme di bit*) da un host all'altro, ma garantendo anche:
  - massima velocità possibile (**PRESTAZIONI**)
  - che si possano superare guasti o malfunzionamenti (**AFFIDABILITA'**)
  - OGGI la **SICUREZZA** della trasmissione

Questi obiettivi relativamente ad un contesto estremamente eterogeneo rendono il problema "meno" banale da risolvere

Che fare quando la complessità è molto elevata?

Metodologia

1. **Dividere il problema in sottoproblemi**
2. **Risolvere i sottoproblemi**
3. **“Collegare” le soluzioni parziali**

- Ovvero, usare **astrazioni** per mascherare la complessità globale dei problemi
- Dal punto di vista informatico, utilizzare un approccio «stratificato» (**layering**)
  - Concetto già visto in altri ambiti (es: sistemi operativi)

Stack (o suite) di protocolli

Stack di protocolli

- **Nelle architetture di rete ci sono diversi tipi di funzionalità associate a ruoli diversi**
- **Esempio concettuale (semplificato):**

Applicazioni di rete

Implementazione di logiche applicative specializzate (Web, Real Time, …)

Implementazione protocolli

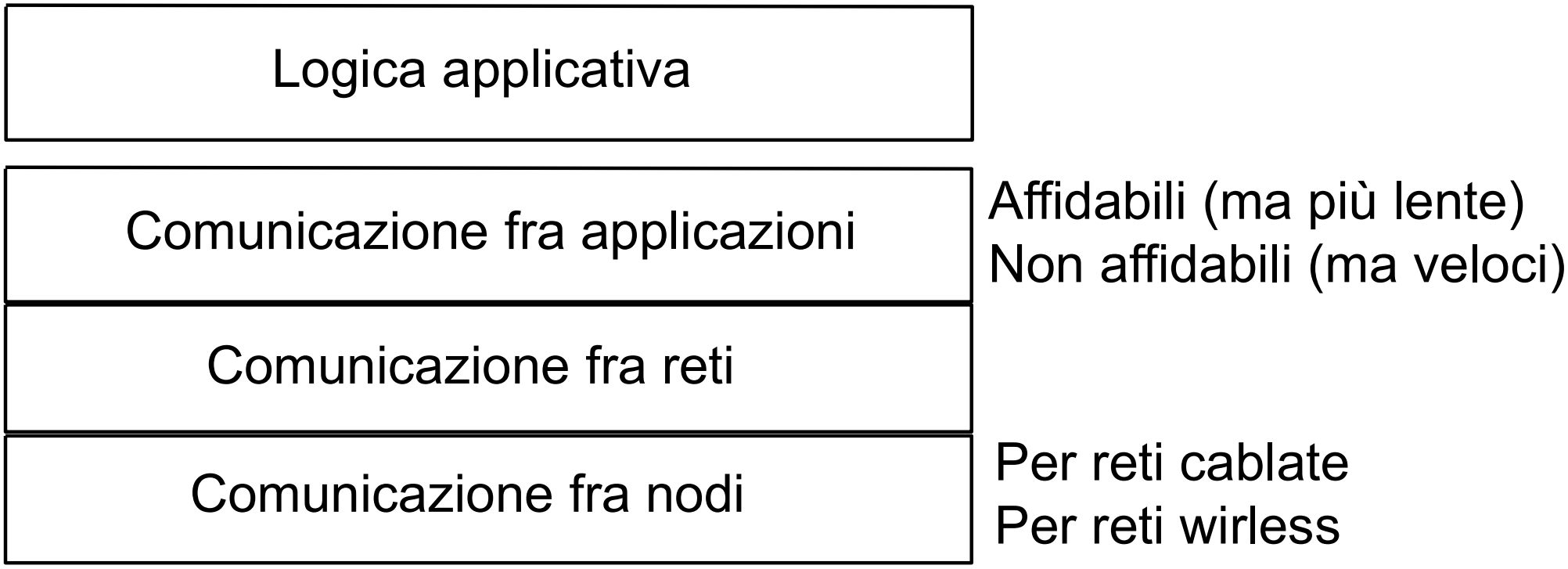
Comunicazione fra applicazioni in esecuzione sugli host

Comunicazione fra nodi di reti differenti

Comunicazione fra nodi di una stessa rete

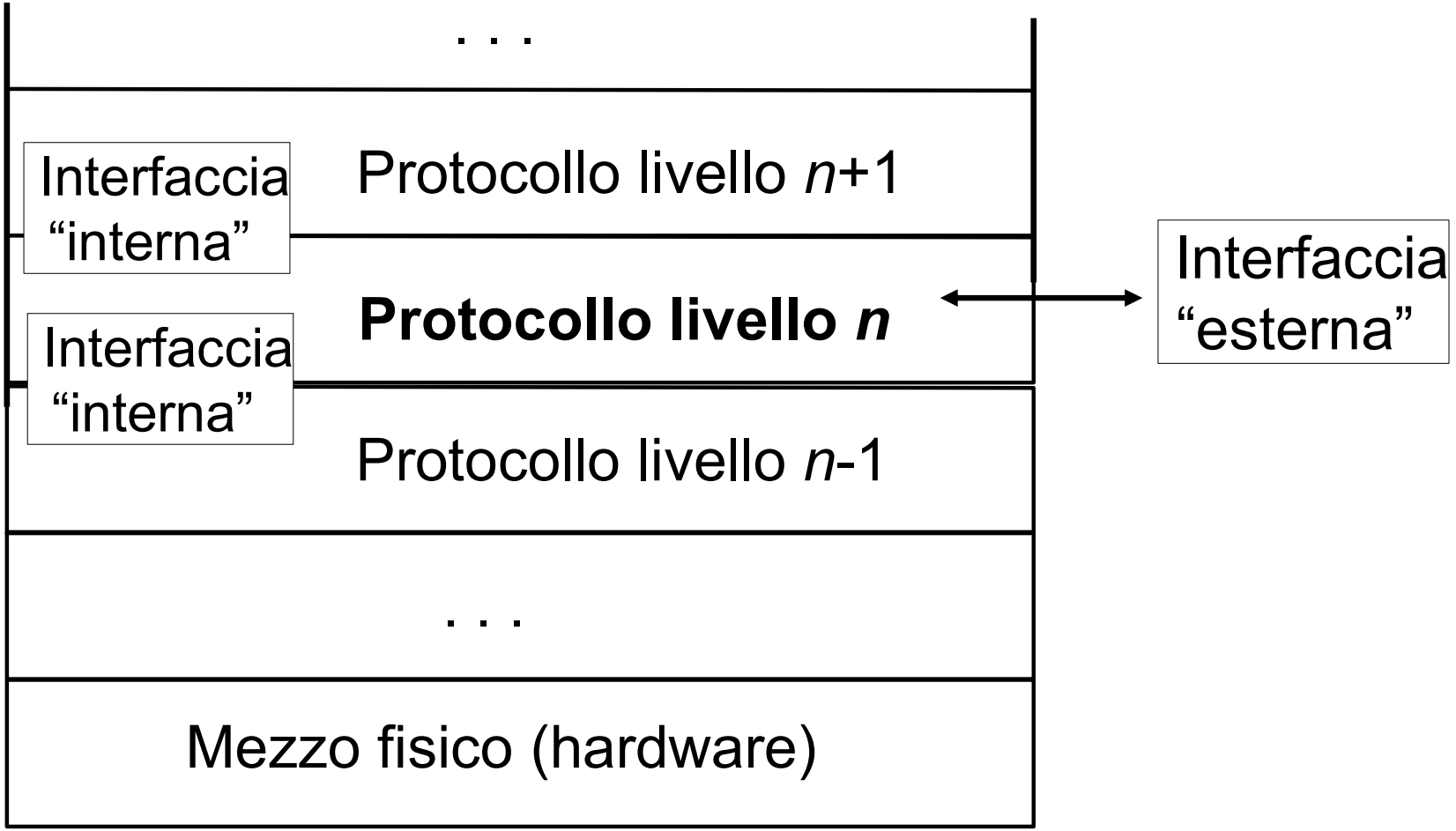
Applicazioni di rete e protocolli

Ogni layer può offrire delle «alternative»
a seconda delle necessità



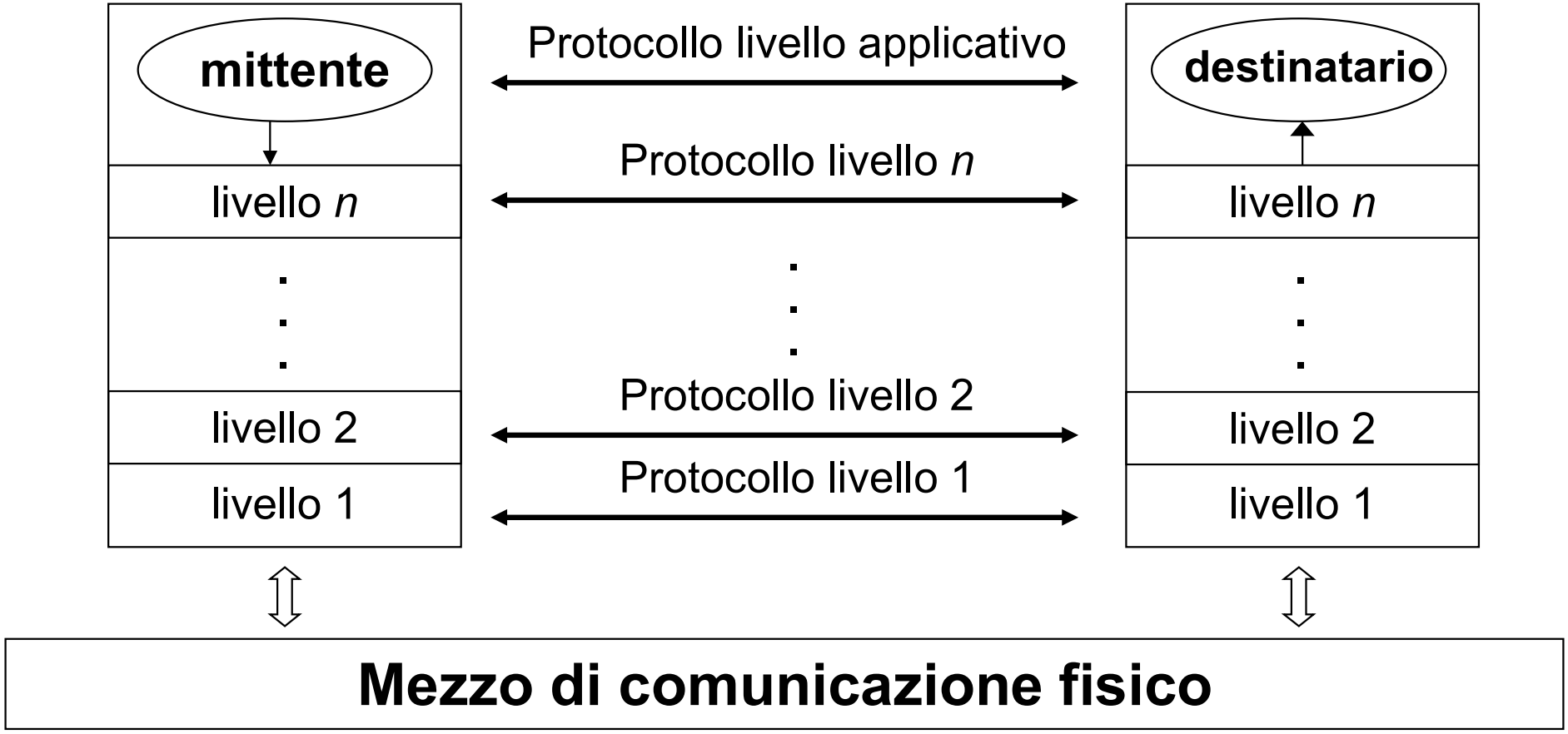
Modello “a livelli” dei protocolli

- Ciascun protocollo, ad un certo livello, ha due interfacce “interne” (verso il livello superiore ed inferiore) ed una interfaccia “esterna” verso il livello equivalente di un altro nodo



Comunicazione concettuale

- La comunicazione avviene logicamente tra **protocolli che si trovano allo stesso livello**



Comunicazione effettiva

- In realtà, la comunicazione tra avviene in modo diretto solo al livello più basso

  Per gli altri livelli, la comunicazione è indiretta

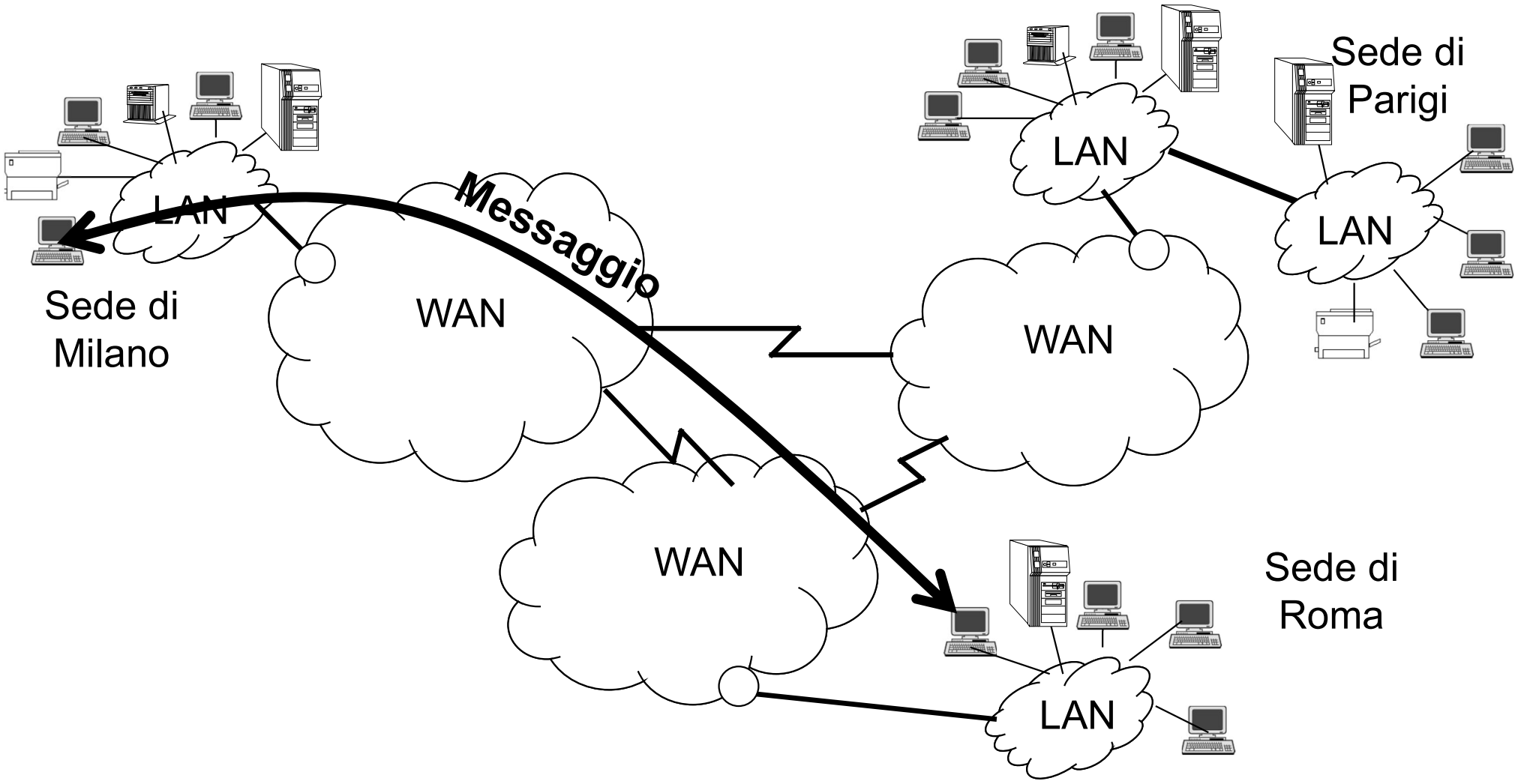
“Computer Networking: A Top-Down Approach Featuring the Internet”

- **<u>Libro di testo</u>: dagli applicativi alle interconnessioni**
- **<u>Lezioni (al contrario)</u>: dalle interconnessioni ai *dettagli* sui protocolli Internet e servizi di rete**
  - Elementi di interconnessione host-to-network
  - Protocolli, Client/server
  - Livello IP
  - Algoritmi di routing
  - Livello di trasporto: TCP/UDP
  - Naming (DNS)
  - Funzionamento applicativi di rete (Web, posta elettronica, ftp, telnet)
  - Server di rete
    - Livello applicativo (Apache Web server, posta elettronica, …)
    - Livello kernel e socket

Alcuni concetti fondamentali

- Scelte fondamentali nello sviluppo di Internet (e, in generale, dei protocolli di rete informatici)
  - Comunicazioni orientate ai pacchetti: **Packet switching**
  - Condivisione delle risorse: **Multiplexing**

Comunicazione logica tra due host

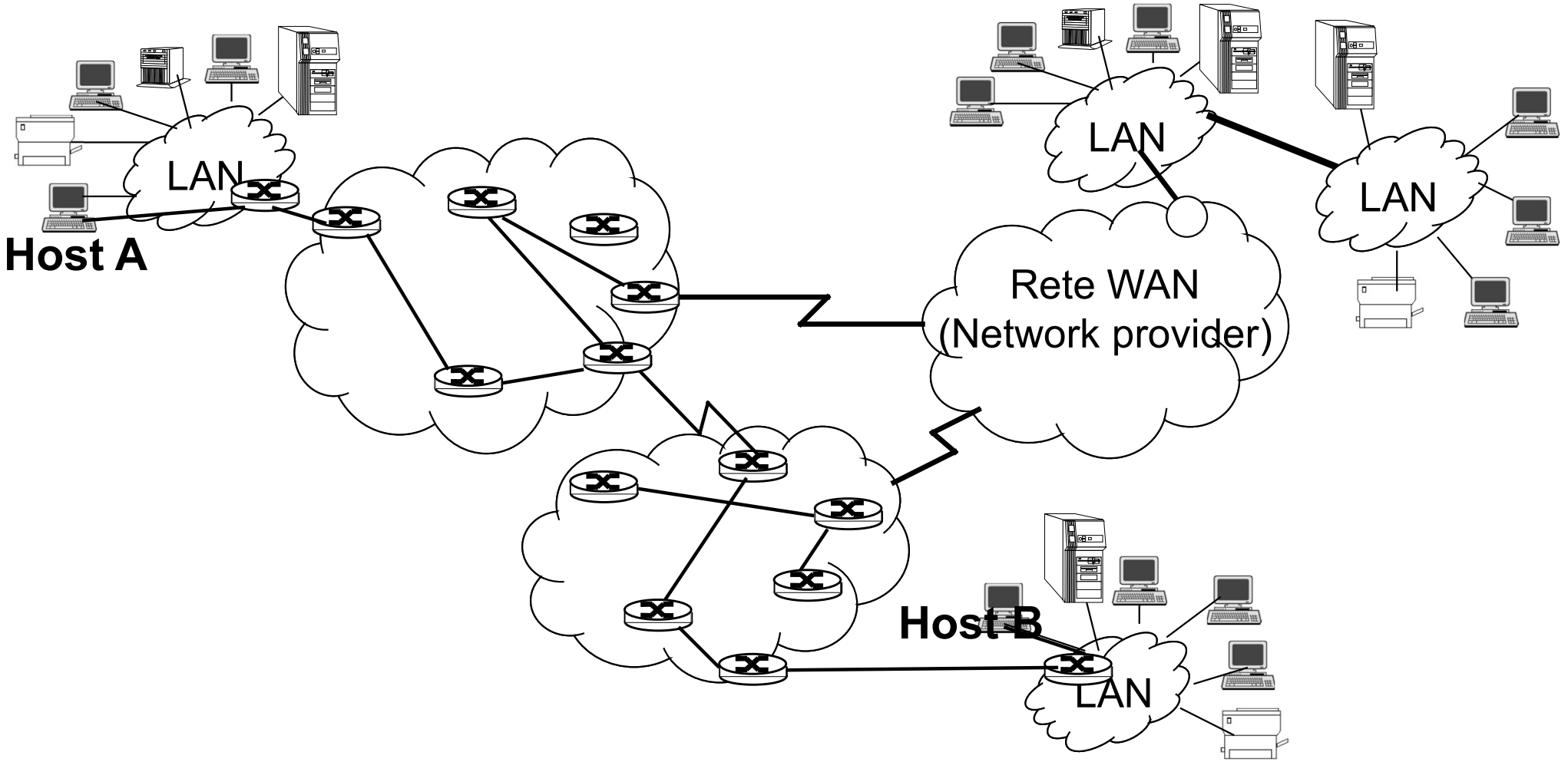


Logicamente comunicano i due host terminali

Comunicazione reale

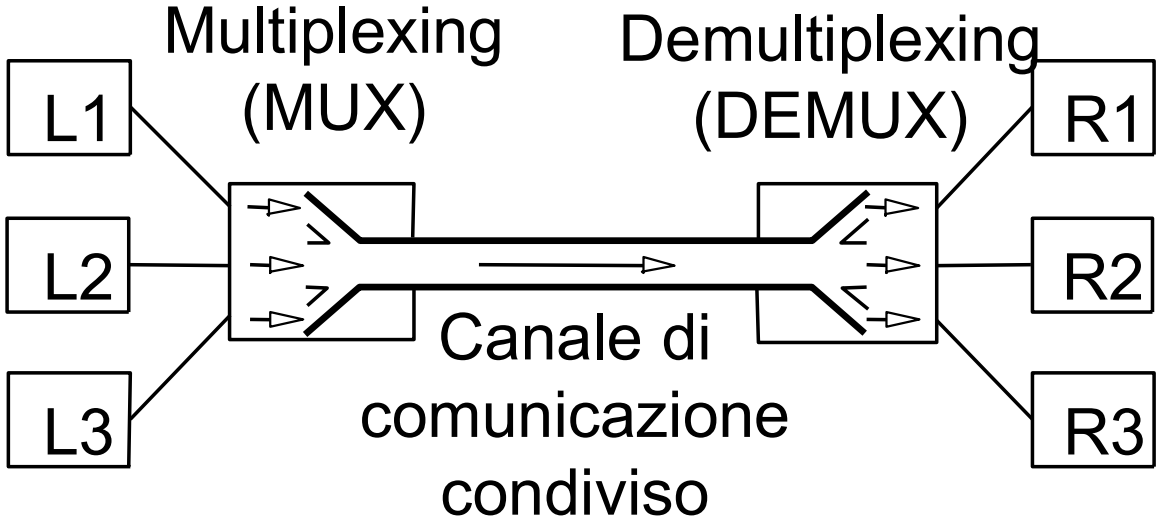
In realtà, le informazioni attraversano tutti i nodi e i collegamenti

- Problemi di ***instradamento*** e di ***condivisione delle risorse***



Multiplexing / Demultiplexing

- Il multiplexing (*condivisione*) di risorse è indispensabile per ottimizzare il loro utilizzo



- La realizzazione del multiplexing dipende dal paradigma di comunicazione e dalle caratteristiche del canale di comunicazione

Due modalità per trasferire dati

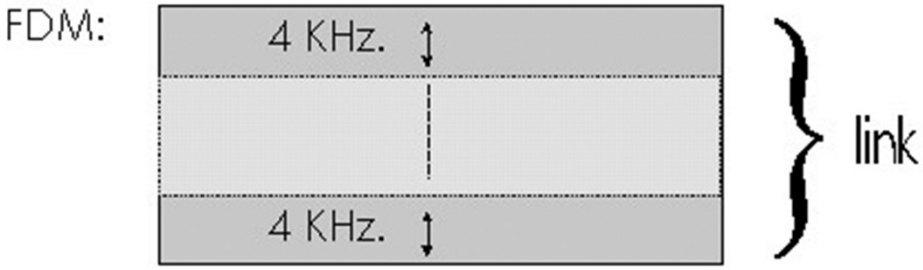
- **Circuit switching**
  - Un circuito virtuale dedicato per ogni comunicazione
    → Alla base di protocolli analogici
- **Packet switching**
  - I dati sono suddivisi in “parti” ed inviati attraverso la rete
    → L’idea alla base di Internet

Circuit switching

- Paradigma di comunicazione **orientato alla connessione**
  - L'invio di dati richiede di effettuare operazioni aggiuntive
  - Prima dell'invio: **apertura di una connessione**
  - Dopo l'invio: **chiusura della connessione**
- **Pro**:
  - in fase di apertura si può verificare la presenza delle risorse necessarie per comunicare
- **Contro**:
  - Necessità di riservare tutte le risorse **all'inizio della comunicazione**
  - Il canale creato occupa una **quantità costante di risorse** a prescindere dall'utilizzo della comunicazione

Multiplexing nel circuit switching su mezzi trasmissivi fisici

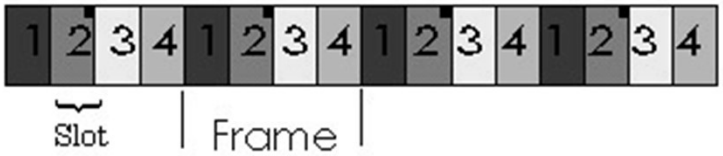
- Esempi noti sono:
  - **Metodi basati sulla frequenza (FDM)**
  - **Metodi basati sul tempo (TDM)**



(Frequency Division Multiplexing)

- FDM: apertura parallela di diversi canali dividendo la banda disponibile

TDM:



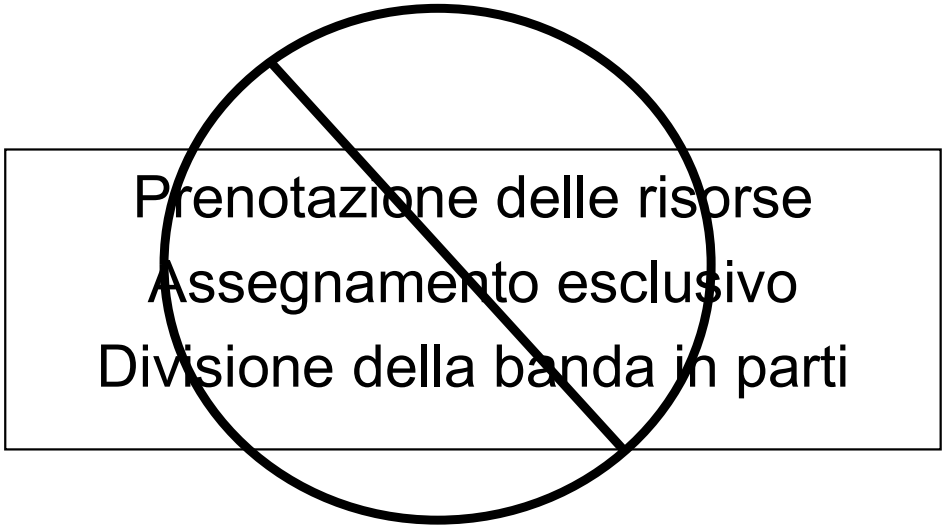
(Time Division Multiplexing)

- TDM: uso alternato di <u>tutta</u> la banda in modo **<u>stabilito a priori</u>**
- Per noi di poco interesse (<u>telecomunicazioni</u>)

Packet switching [1]

Ogni comunicazione è suddivisa in pacchetti

- Ogni pacchetto utilizza tutta la capacità trasmissiva di un link
- Accesso al mezzo **non è temporizzata a priori (!= TDM)**
- **Le risorse sono utilizzate sulla base della necessità e non della prenotazione**



Packet switching [2]

- Una decisione fondamentale nella progettazione dei protocolli informatici (e Internet)
- Le comunicazioni avvengono tramite **commutazione a pacchetto (packet switching)**
  - Efficacia dimostrata nel 1961 da Kleinrock mediante la teoria delle reti di code
  - Inizialmente considerato *senza futuro* da parte di una buona parte di telecomunicazionisti dell'epoca ("It will never work", "La storia di Internet scritta da coloro che l'hanno creata", 1997)

Packet switching [3]

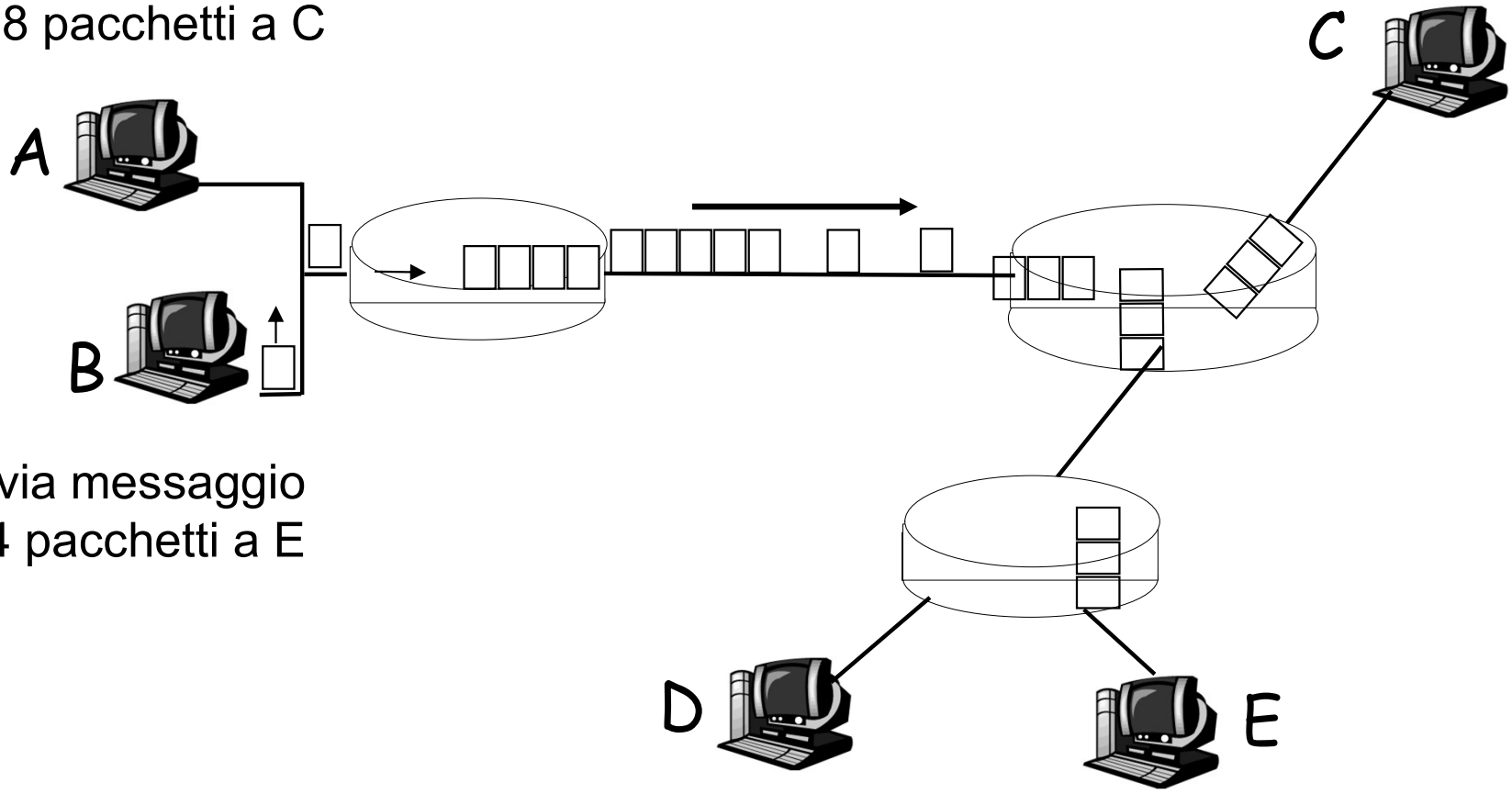
- Si può dire che il packet switching segua un principio di multiplexing statistico a suddivisione di tempo
  - Pacchetti provenienti da diverse sorgenti sono "mescolati" sullo stesso link senza slot di tempo dedicati a priori e occupando tutta la banda disponibile

Packet switching [4]

- Poiché non c'è garanzia di avere una risorsa disponibile, ci può essere conflitto e conseguenti problemi a diversi livelli (approfondiremo):
  - (Locali) **Collisioni** nelle reti locali con mezzi fisici condivisi
  - (Reti) **Congestione** nei buffer dei router
- In generale: può provocare errori o perdite di pacchetti e intaccare l'affidabilità della comunicazione «fisica»
  - I protocolli di rete devono essere progettati per porre rimedio a questo problema

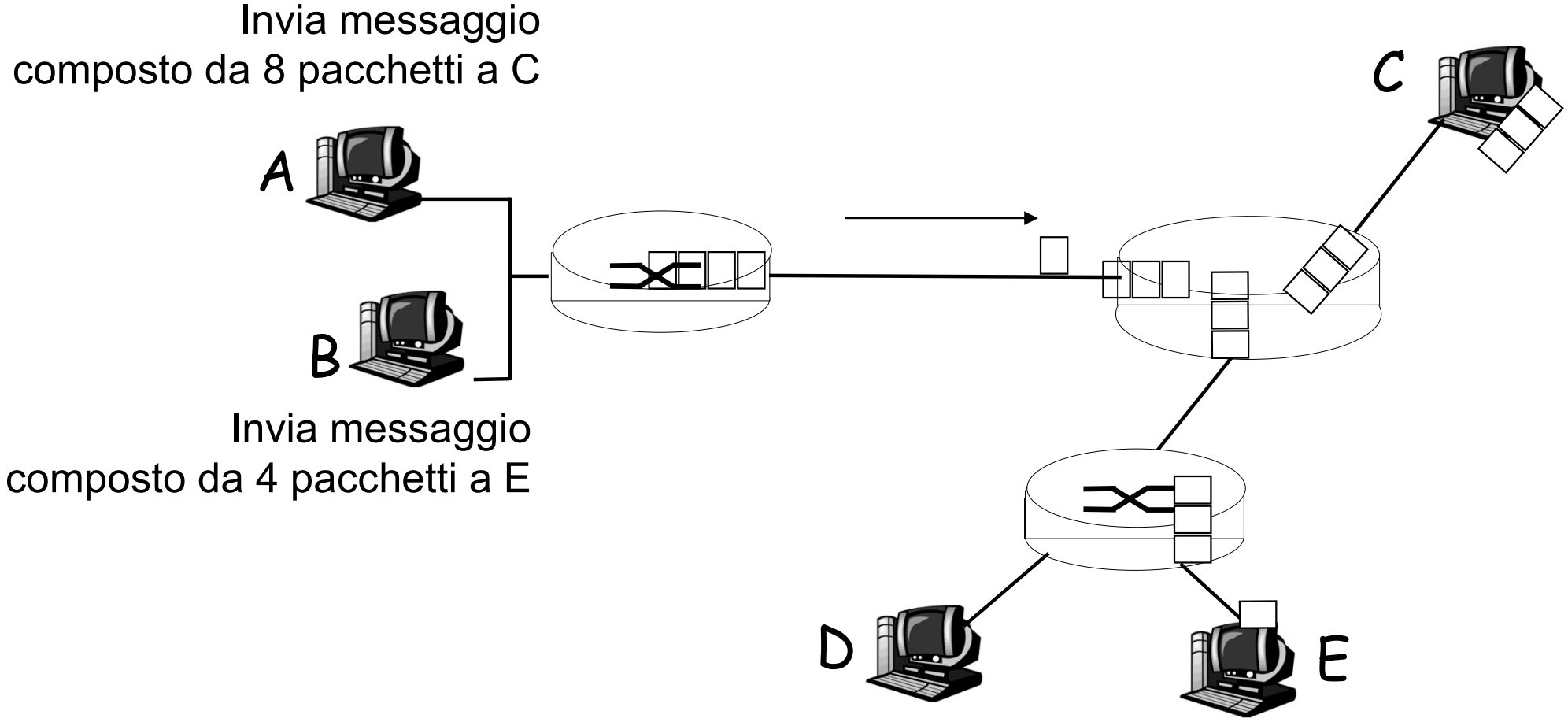
Packet switching [5]

Invia messaggio
composto da 8 pacchetti a C



Invia messaggio
composto da 4 pacchetti a E

Packet switching [6]



Metrica di prestazione

- **Bandwidth (banda di trasmissione): quantità di dati trasmessi per unità di tempo**

Tipicamente:

  - Unità di tempo = **secondo**
  - Quantità di dati trasmessi = **multipli di <u>bit</u>**

- Quindi, metriche tipiche sono:
  - **Kbps** o Kbit/s → Kilo-bit per secondo
  - **Mbps** o Mbit/s → Megabit per secondo
  - **Gbps** o Gbit/s → Gigabit per secondo

    [Notare la *b* minuscola]

Vantaggi del packet switching

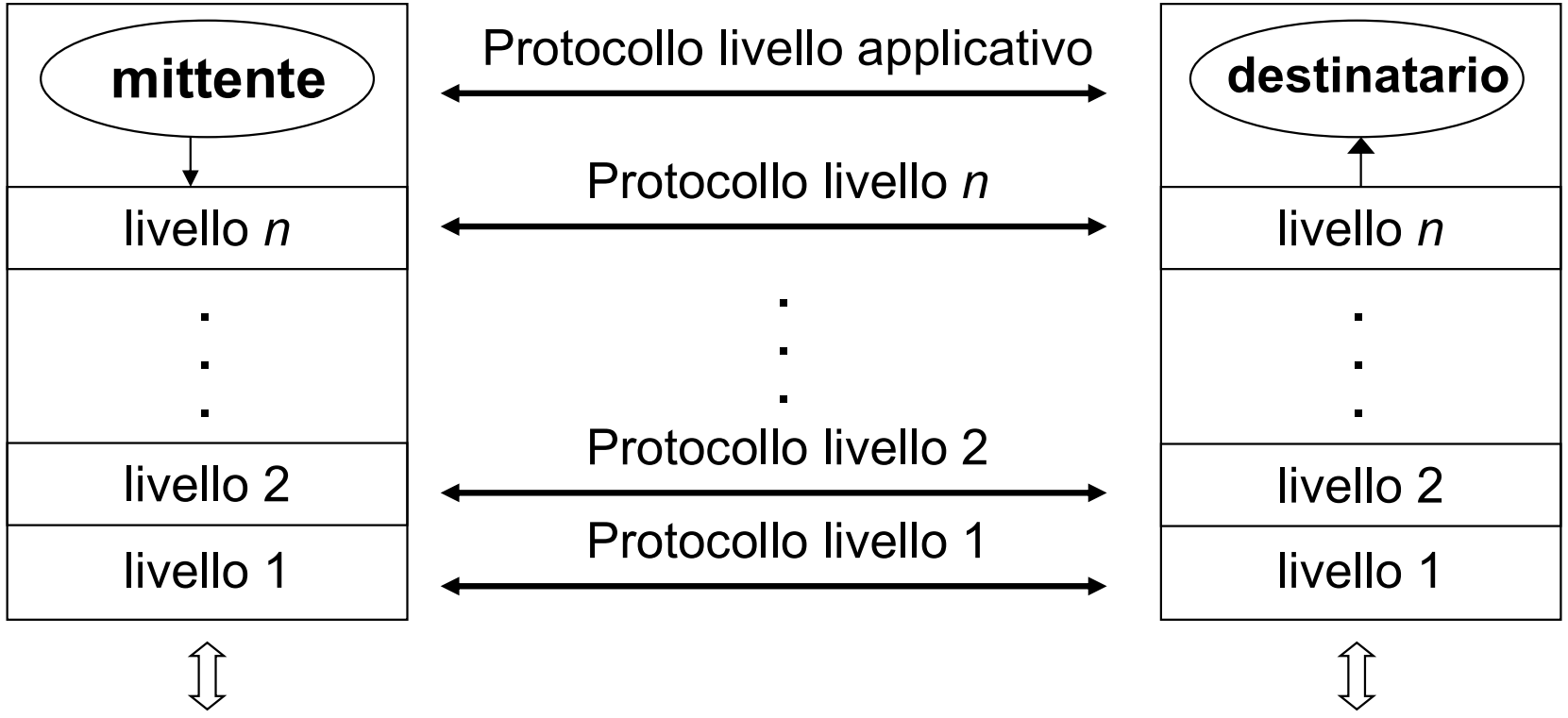
Esempio

- Link a 1 Mbps
- Ciascun utente richiede 0.1 Mbps quando trasmette, ed è attivo il 10% del tempo

- ***Circuit switching***: può supportare al più 10 utenti
- ***Packet switching***: con 35 utenti, la probabilità che più di 10 utenti trasmettano contemporaneamente è bassissima (0.0004), quindi è possibile far comunicare 35 utenti sulla stessa linea con minimi rischi di conflitti

Implementazione dello stack per Internet

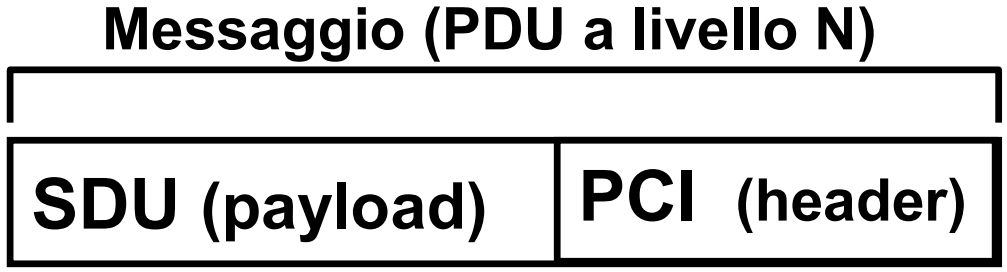
- La progettazione di Internet si basa su
  - Stack di protocolli
  - Paradigma **Packet Switching**
- Come funziona lo stack dei protocolli di Internet?



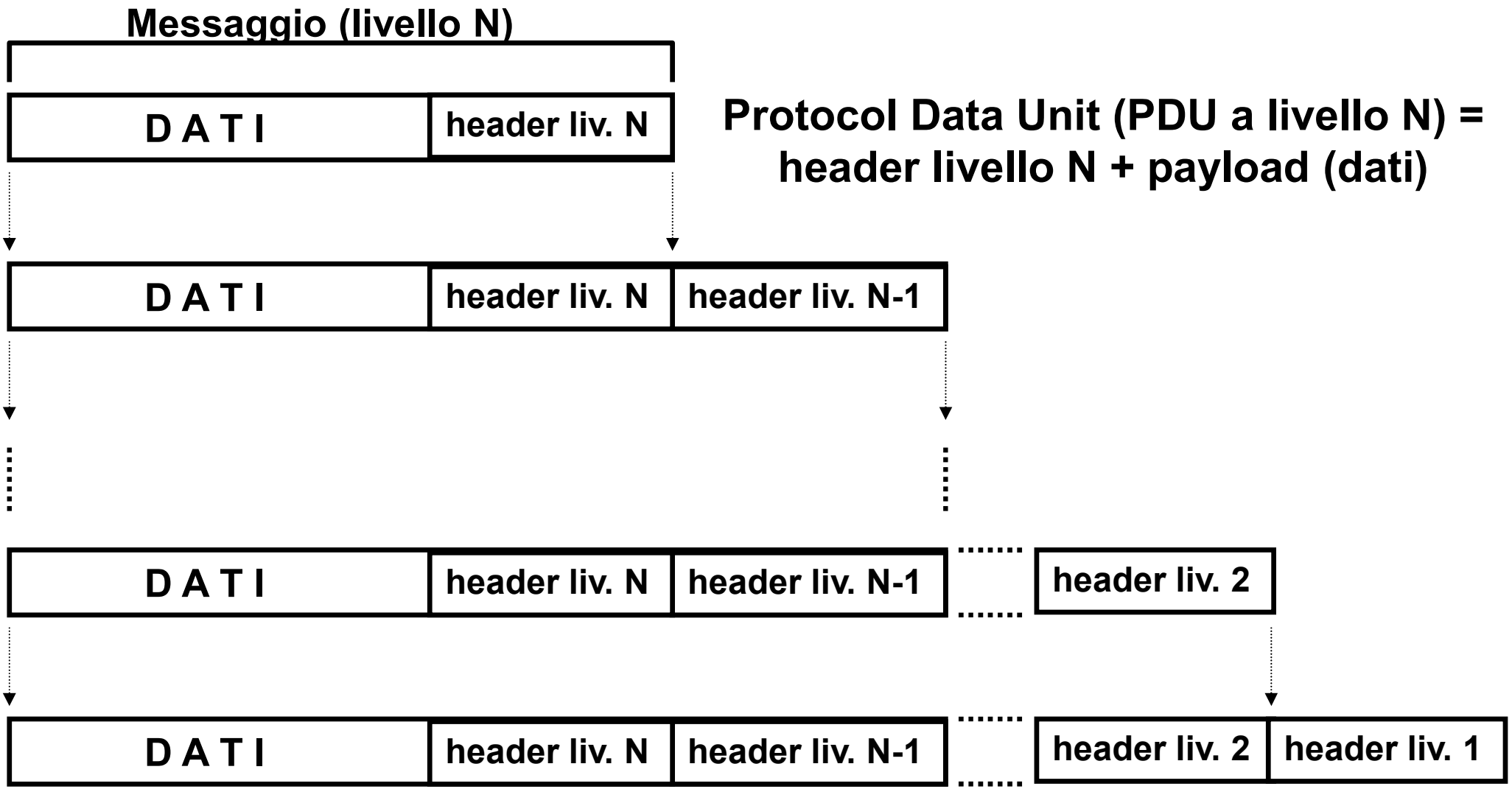
Communication Unit in Packet switching: PDU

- Ogni «pacchetto» è costituito da
  - **Protocol Control Information** (**PCI**) → ***header***
    - Insieme dei **metadati** necessari al corretto funzionamento del protocollo stesso
  - **Service Data Unit** (**SDU**) → ***payload***
    - Il vero contenuto informativo scambiato (termine utilizzato anche in ogni ambito di trasporto merci: aerei, camion, ecc.)

PCI + SDU = PDU (Protocol Data Unit)



Mezzi per realizzarla (2): Incapsulamento del messaggio

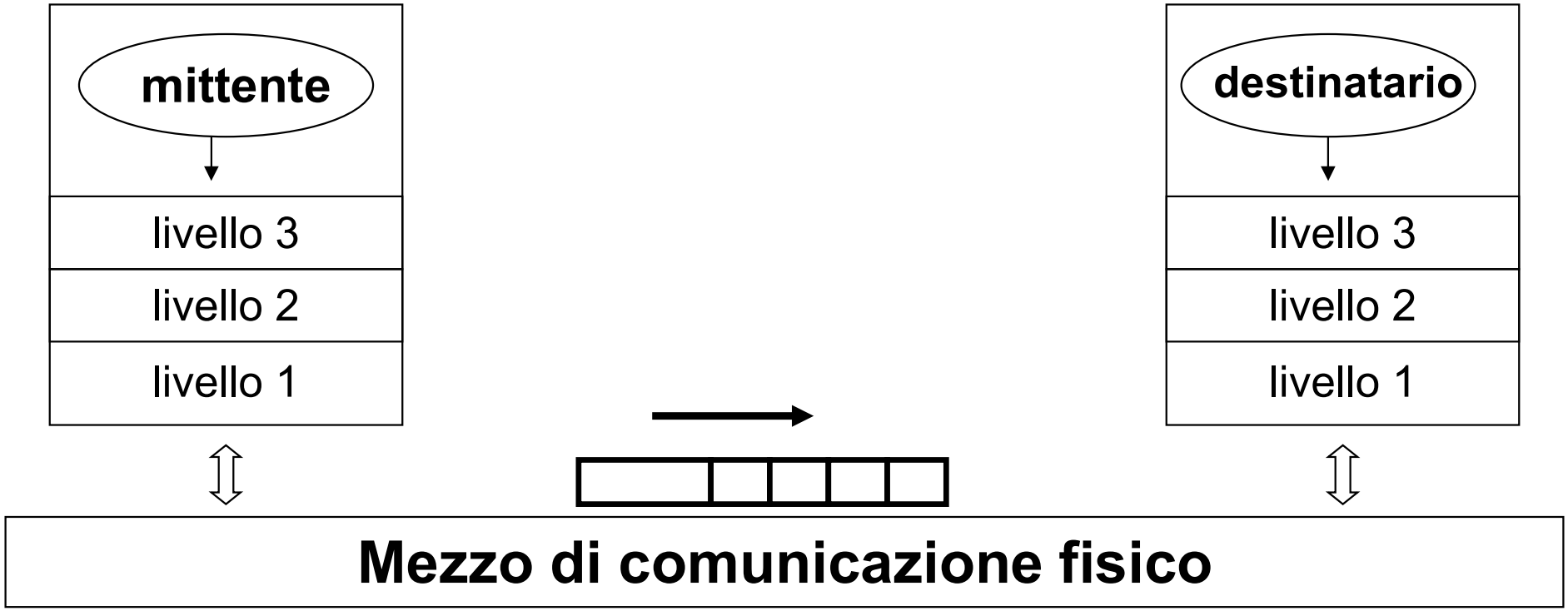


Come avviene la comunicazione

(STEP 1)

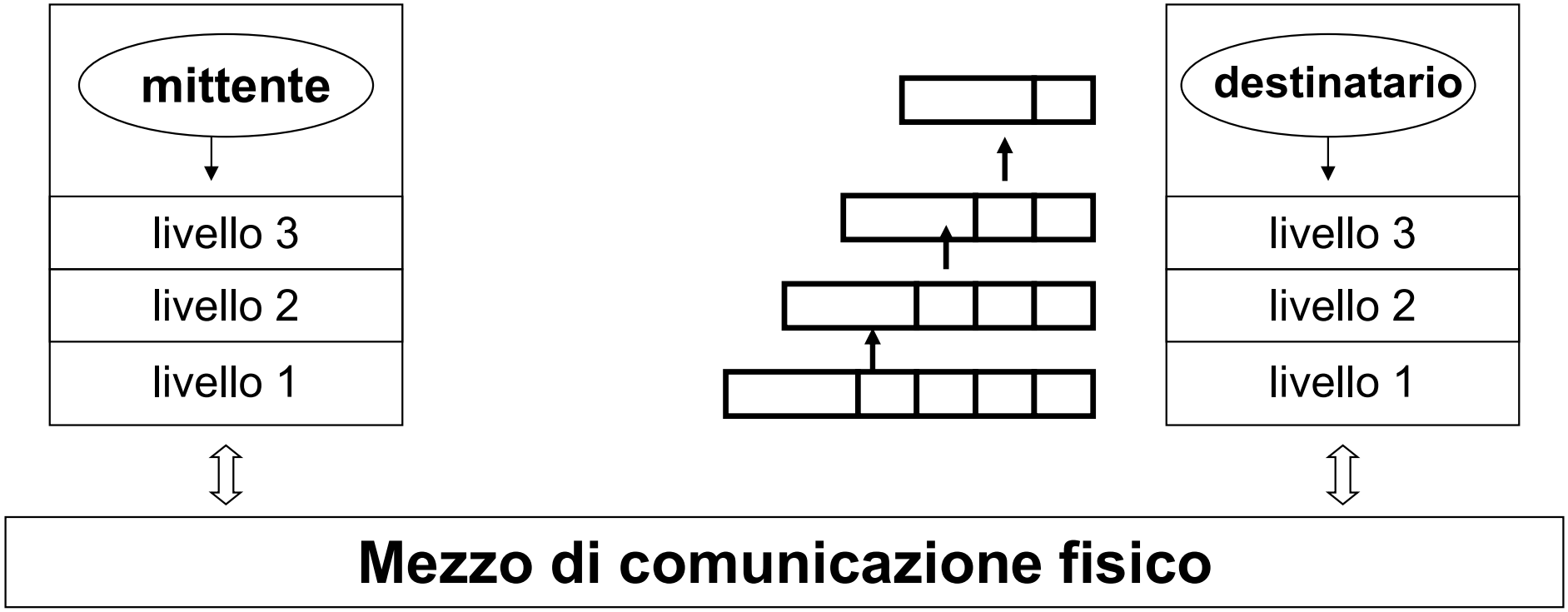
Come avviene la comunicazione

(STEP 2)



Come avviene la comunicazione

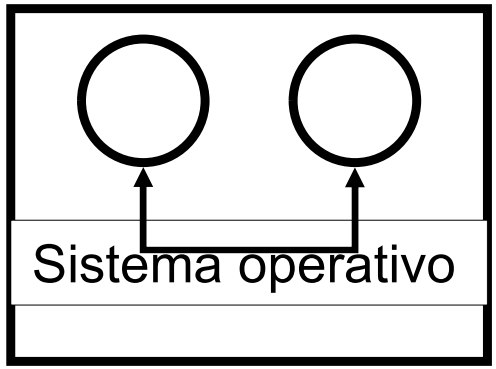
(STEP 3)



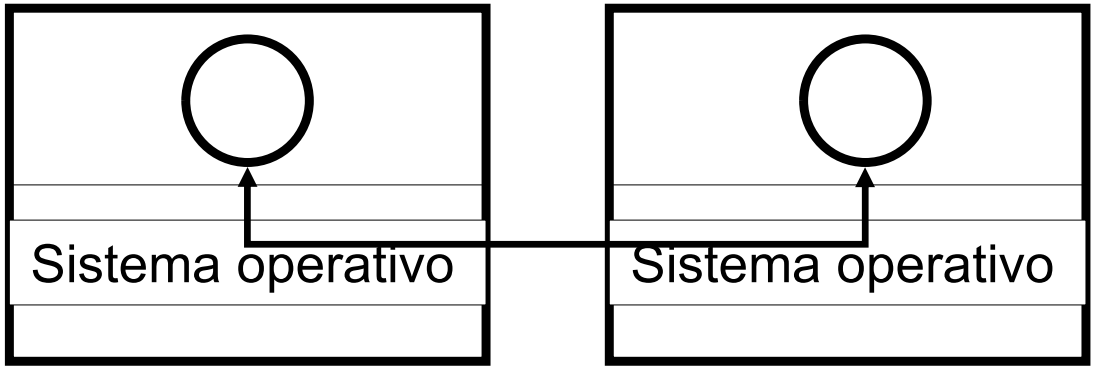
Protocolli di Rete Informatici - SINTESI

1. Il sistema di comunicazione basato su Internet richiede un insieme di protocolli tra loro cooperanti (detti ***protocol suite*** o ***protocol stack***)
2. Si identifica una ***relazione gerarchica*** nelle funzioni che compongono un processo di comunicazione: ***Architettura a livelli (layer)***
3. **Vi è indipendenza funzionale tra i vari livelli**: il servizio fornito da un livello è definito in modo indipendente dalle procedure con cui è implementato
4. **Il livello *n*, sfruttando anche il servizio offerto dal livello *n-1*, fornisce un servizio al livello *n+1***
5. La comunicazione avviene logicamente tra *pari*, ma in realtà attraversa tutti i livelli sottostanti, mediante **incapsulamento del messaggio** a ciascun livello

Analogia con comunicazioni tra processi



Interprocess communication



Comunicazione tra processi su host diversi (necessità di un protocollo di livello applicativo)

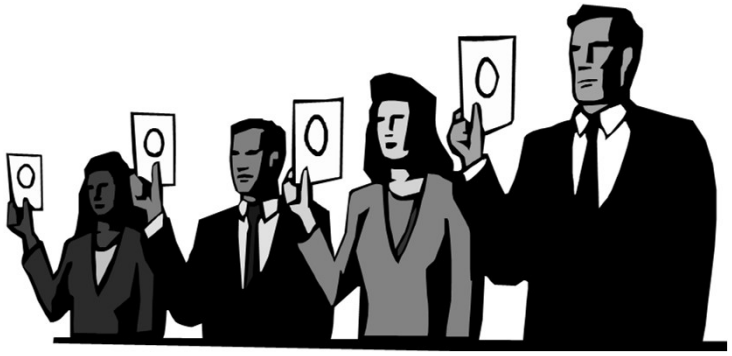
Comunicazione e standard

- La comunicazione tra nodi differenti e, possibilmente, basati su piattaforme hardware e/o software eterogenee necessita di **STANDARD**

- L'informatica, da sempre, conosce due modi per arrivare ad uno standard
  - **STANDARD *de iure***
  - **STANDARD *de facto***

- Gli standard di comunicazioni tra calcolatori offrono un esempio "storico"

Due standard in concorrenza



ISO/OSI

(*de iure*)

TCP/IP

(*de facto*)

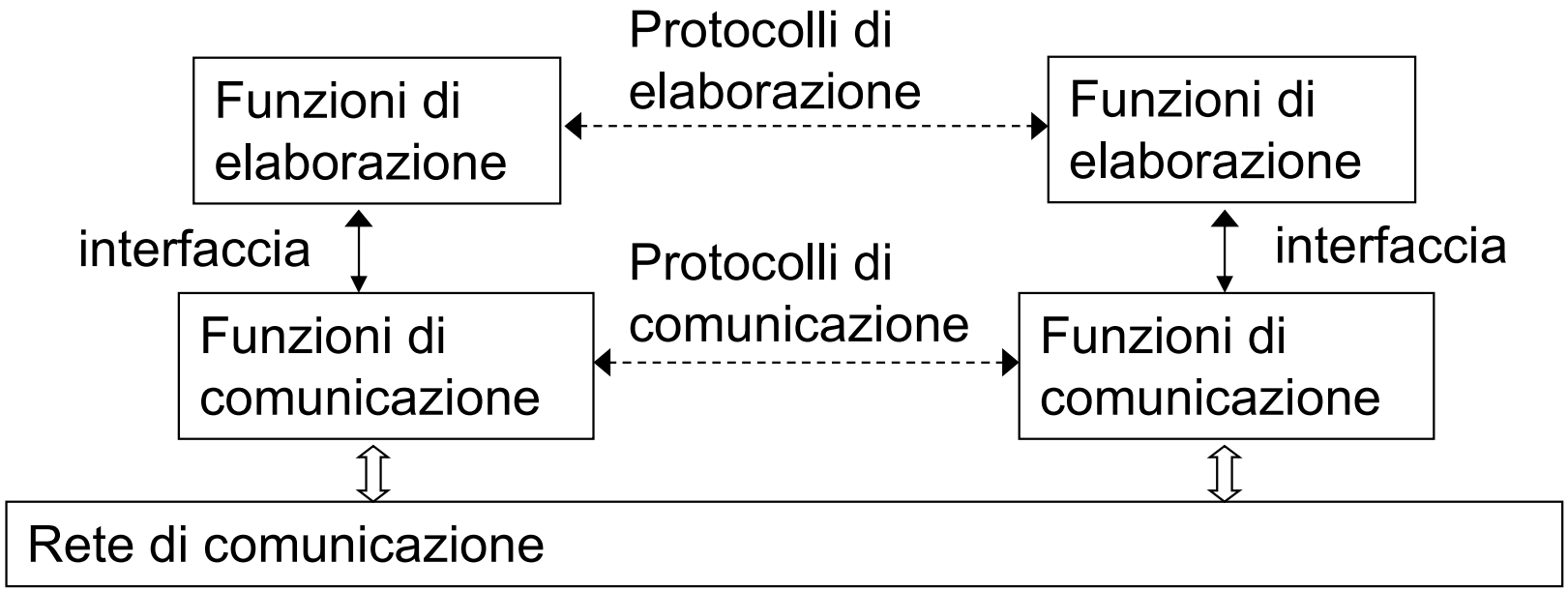
Il caso dello "standard *de iure*" ISO/OSI

- L'organizzazione **ISO** (***International Standard Organization***) ha definito le specifiche di quello che sarebbe dovuto diventare lo standard di protocolli per l'interconnessione di nodi eterogenei: **OSI** (***Open System Interconnection***)

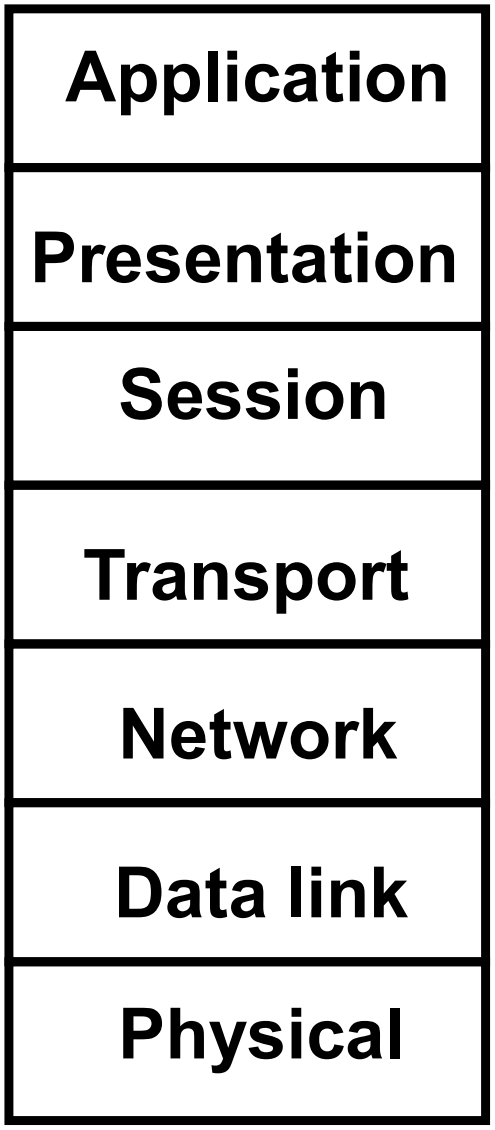
Funzionalità del modello ISO/OSI

1) ***Protocolli di comunicazione (network level)***: riguardano la comunicazione di messaggi tra nodi della rete, in modo da nascondere le caratteristiche dei mezzi fisici di trasmissione alle funzionalità di elaborazione

2) ***Protocolli di elaborazione (application level)***: insieme di meccanismi per il controllo delle applicazioni

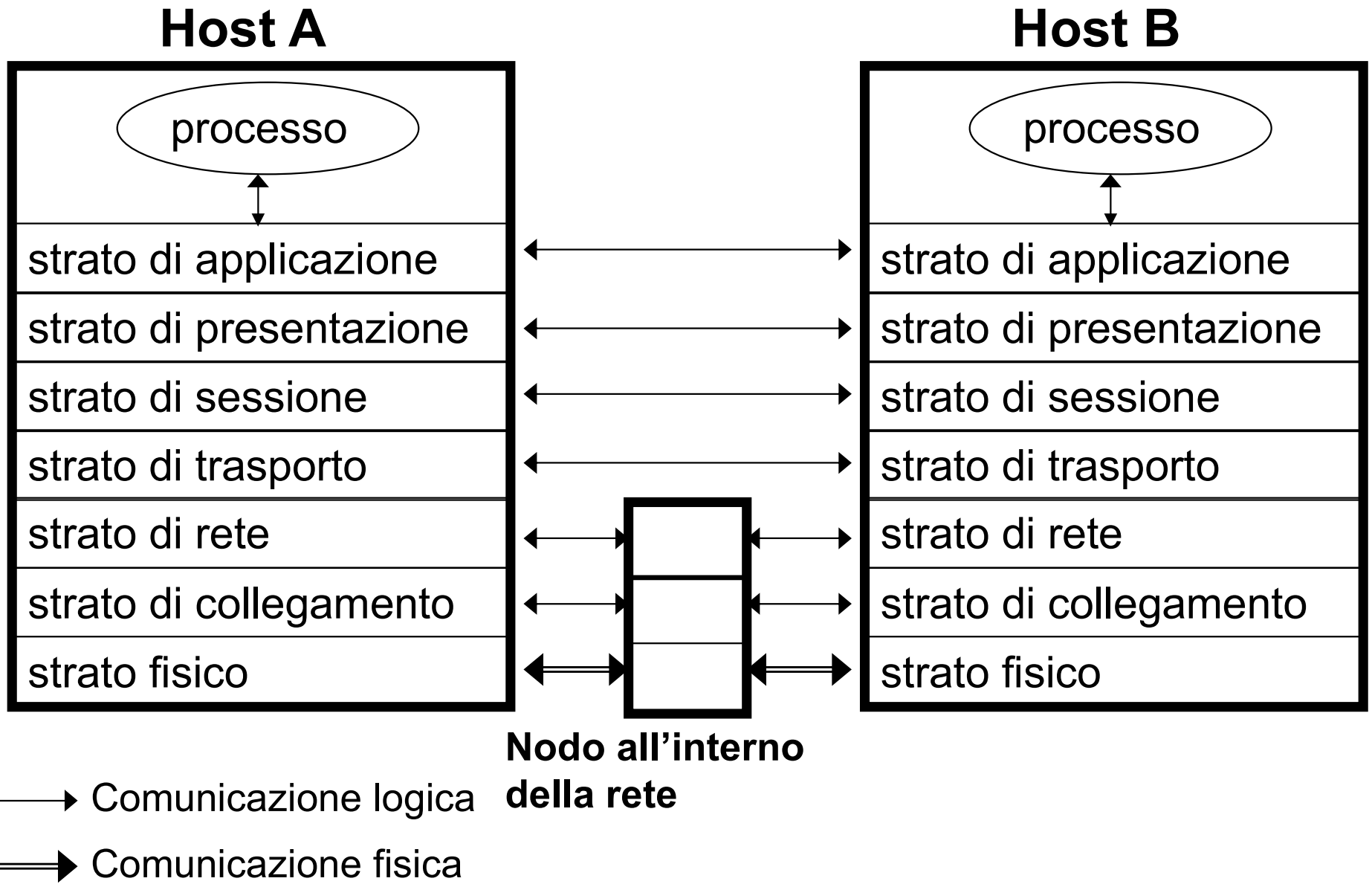


I 7 livelli dello stack ISO/OSI



← 7 livelli

I 7 livelli del modello ISO/OSI



Livelli ISO/OSI

- ***Livello fisico*** (1): Gestisce i particolari meccanici ed elettrici della trasmissione fisica di un flusso di bit
- ***Livello di collegamento dati*** (2): Gestisce i ***frame*** o i ***pacchetti*** trasformando la semplice trasmissione in una linea di comunicazione priva di errori non rilevati.
  - Gestisce l'accesso e l'uso dei canali fisici, gestisce il formato dei messaggi suddividendo (ove necessario) i dati in frame.
  - Gestisce la corretta sequenza dei dati trasmessi, comprendente l'uso di codifiche ridondanti (ad es., bit di parità) per l'individuazione e la correzione di errori che si sono verificati nello strato fisico, e la conferma dell'avvenuta ricezione
- ***Livello di rete*** (3): Fornisce i collegamenti e l'instradamento dei pacchetti nella rete, comprese la gestione dell'indirizzo dei pacchetti in uscita, la decodifica dell'indirizzo dei pacchetti in ingresso e la gestione delle informazioni di instradamento (ad es., router)

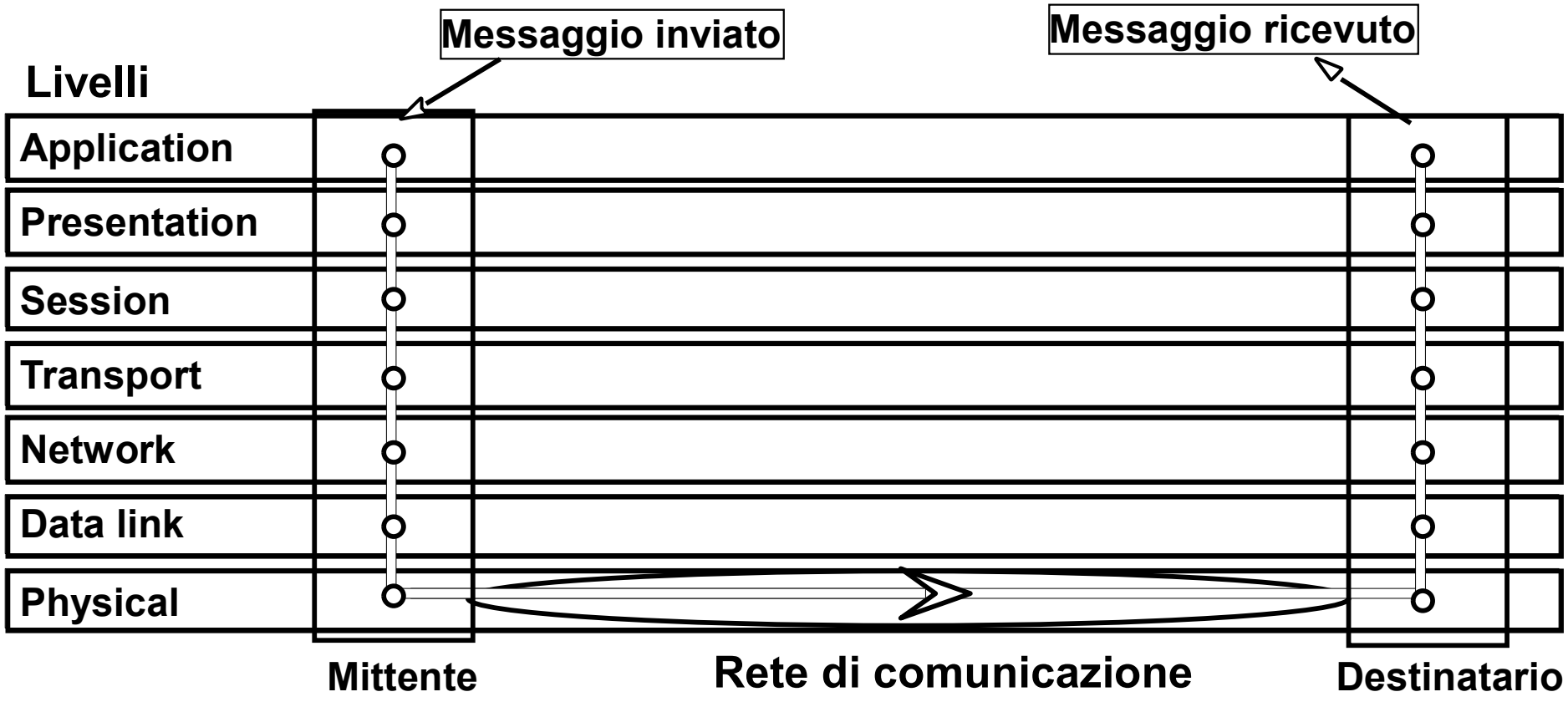
Livelli ISO/OSI (*cont.*)

- ***Livello di trasporto*** (4): Effettua il controllo end-to-end della sessione di comunicazione (accesso alla rete da parte del client e trasferimento dei messaggi tra i client) e garantisce l'affidabilità del trasporto
- ***Livello di sessione*** (5): Consente a utenti su macchine eterogenee di stabilire sessioni, implementando funzioni di coordinamento, sincronizzazione e mantenimento dello stato (di sessione)
- ***Livello di presentazione*** (6): Risolve le differenze di formato che possono presentarsi tra diversi nodi della rete (ad es., conversione tra caratteri ASCII, Unicode, EBCDC, conversione di codifica tra little- e big-endian), ma gestisce anche la compressione dei dati, la sicurezza e l'autenticità dei messaggi attraverso tecniche di crittografia
- ***Livello di applicazione*** (7): Fornisce un'interfaccia standard per i programmi applicativi che utilizzano la rete, mascherando le peculiarità e la complessità del sistema sottostante

Formato del messaggio inviato

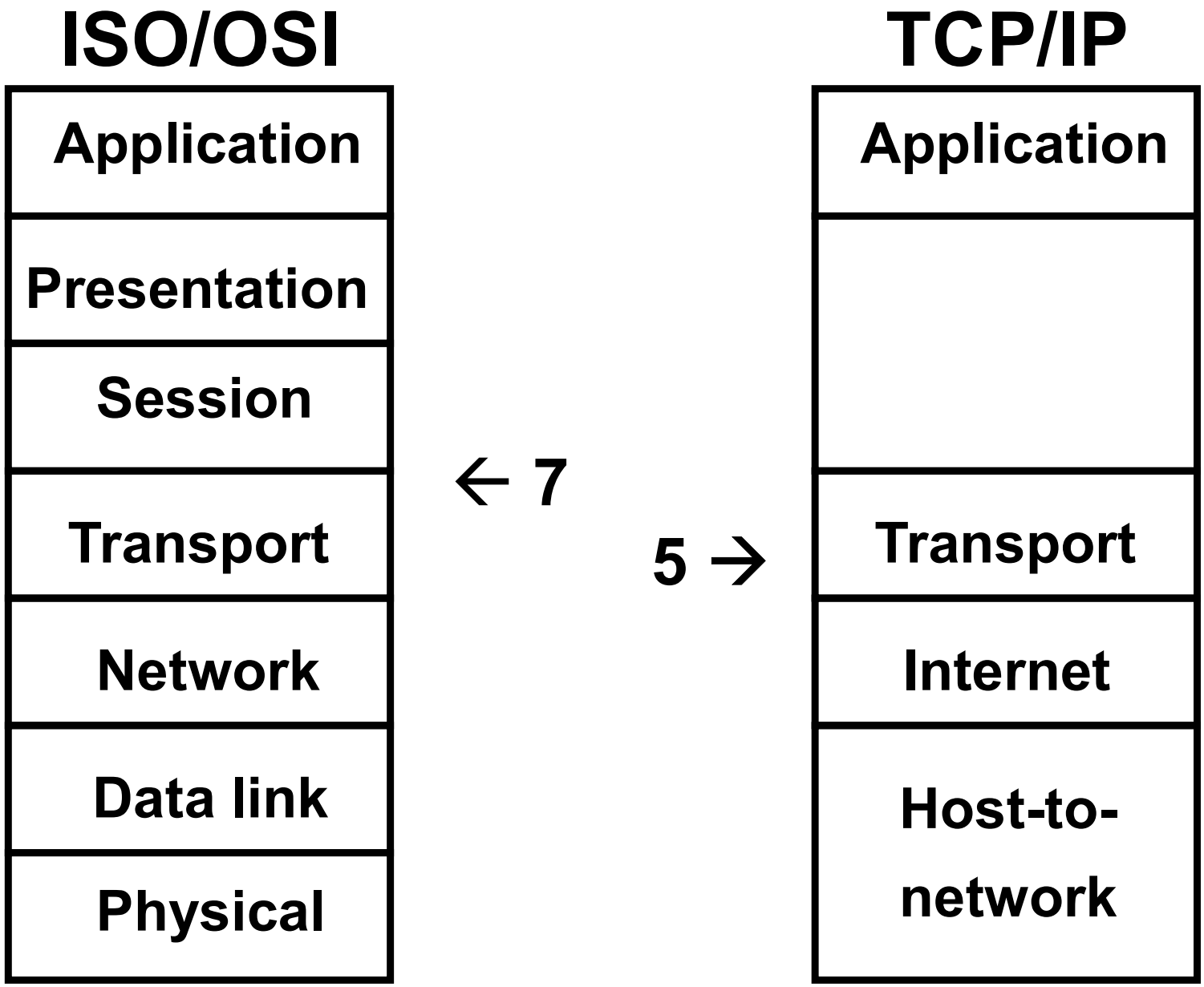
- Messaggio (PDU) composto da ***intestazione*** (*header*) e ***dati***
- Ogni livello aggiunge una propria intestazione

Comunicazione nel Modello ISO/OSI



Ma l'ISO/OSI non è riuscito ad affermarsi perché nel frattempo stava esplodendo ……

I livelli dei due Protocol Stack



Critiche

Al modello ISO/OSI

- **Cattiva tempistica**
- **Cattiva tecnologia**
  - Influenzato dal modello IBM-SNA
  - Ridondanze
- **Cattiva implementazione**
  - Complessità
  - Eccessivi 7 livelli per la tecnologia (reti-computer) del tempo
- **Pessima politica**
  - Modello imposto contro il libero TCP/IP legato a Unix

Al modello TCP/IP

- **Poco generale**
- **Meno concettuale e più orientato al funzionamento**
- **Livelli host-to-network confusi e interdipendenti**
- **Protocolli sviluppati ad hoc invece che protocolli generali**

Altro motivo del successo di TCP/IP

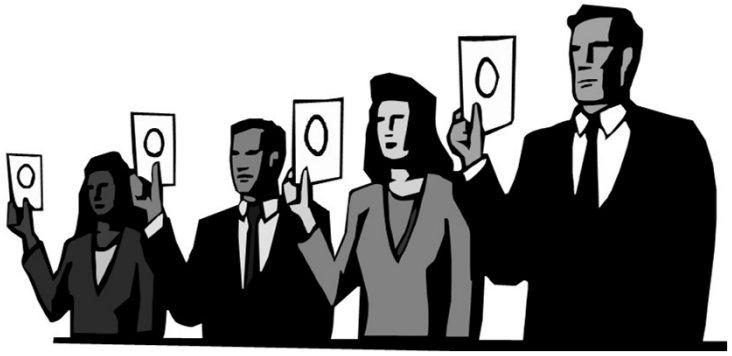
- Disponibilità di una buona implementazione dello stack in versione open source a metà degli anni ’80 su **BSD Unix**
- Disponibilità di un buon insieme di API (BSD socket API) per sviluppare applicazioni di rete: non perfette, ma funzionanti

- Al contrario,
  - il comitato ISO/OSI definì le specifiche dello stack
  - le implementazioni funzionanti delle specifiche ISO/OSI erano molto in ritardo rispetto a quelle già disponibili TCP/IP

Due standard non più in concorrenza

ISO/OSI

(*de iure*)

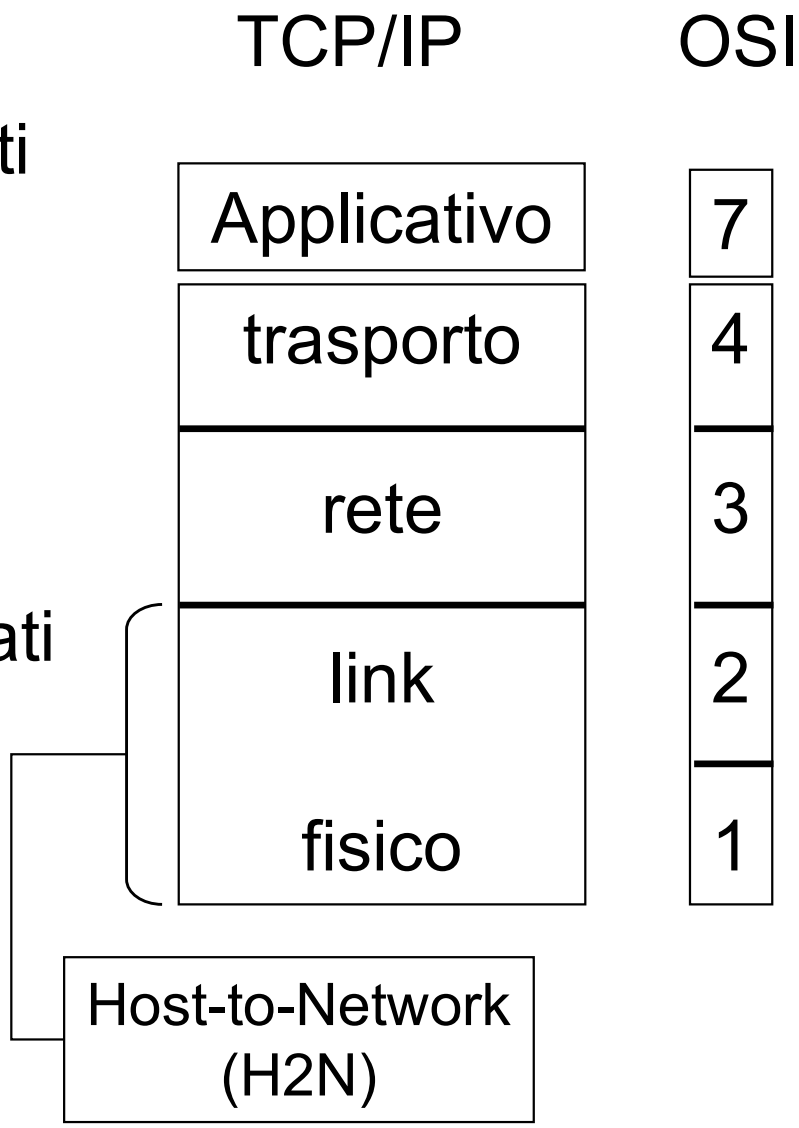


TCP/IP

(*de facto*)

Suite di protocolli Internet (TCP/IP)

- **Trasporto (4)**: supporta i trasferimenti fra processi in esecuzione su host
- **Rete (3)**: trasferisce i pacchetti dal nodo mittente al destinatario
- **Link (2)**: effettua i trasferimenti dei dati tra componenti della rete confinanti
- **Fisico (1)**: trasferisce bit “sul cavo”



Livello 1-2 (*host-to-network*)

- I primi due livelli (*fisico* e *data link*) non sono separati, nel senso che **connessione fisica** e protocollo **data link** sono interdipendenti

- Pertanto, nel caso dello stack TCP/IP è più corretto parlare di un livello **host-to-network** (**h2n**) che comprende i primi due livelli

- Esempi di protocolli h2n:
  - Protocollo per LAN: ***Ethernet***, ***token-ring***
  - Protocollo per connessioni via modem: ***PPP***
  - Protocollo per connessioni LAN wireless: ***802.11***

Livello 3 (*network*): Protocollo IP

- **Protocollo per la consegna dei pacchetti da un host mittente ad un host destinatario**
- ***Servizi aggiuntivi rispetto a h2n***
  - ***identificativo univoco di ciascun host (indirizzo IP)***
  - ***comunicazione logica tra host***
- ***Ma***
  - ***privo di connessione***: ogni pacchetto è trattato in modo indipendente da tutti gli altri
  - ***non affidabile***: la consegna non è garantita (i pacchetti possono essere persi, duplicati, ritardati, o consegnati senza l'ordine di invio)
  - ***consegna con impegno***: tentativo di consegnare ogni pacchetto (l'inaffidabilità deriva dalle possibili congestioni della rete o guasti dei nodi/router)

Livello 4 (*transport*)

- **Il livello transport estende il servizio di consegna con impegno proprio del protocollo IP tra due host terminali ad un servizio di consegna a due processi applicativi in esecuzione sugli host**
- ***Servizi aggiuntivi rispetto a IP***
  - ***multiplazione e demultiplazione messaggi tra processi***
  - ***rilevamento dell'errore*** *(mediante checksum)*
- Esempi di protocolli *transport*
  - **UDP (*User Datagram Protocol*)**
  - **TCP (*Transmission Control Protocol*): offre servizi aggiuntivi rispetto a UDP**

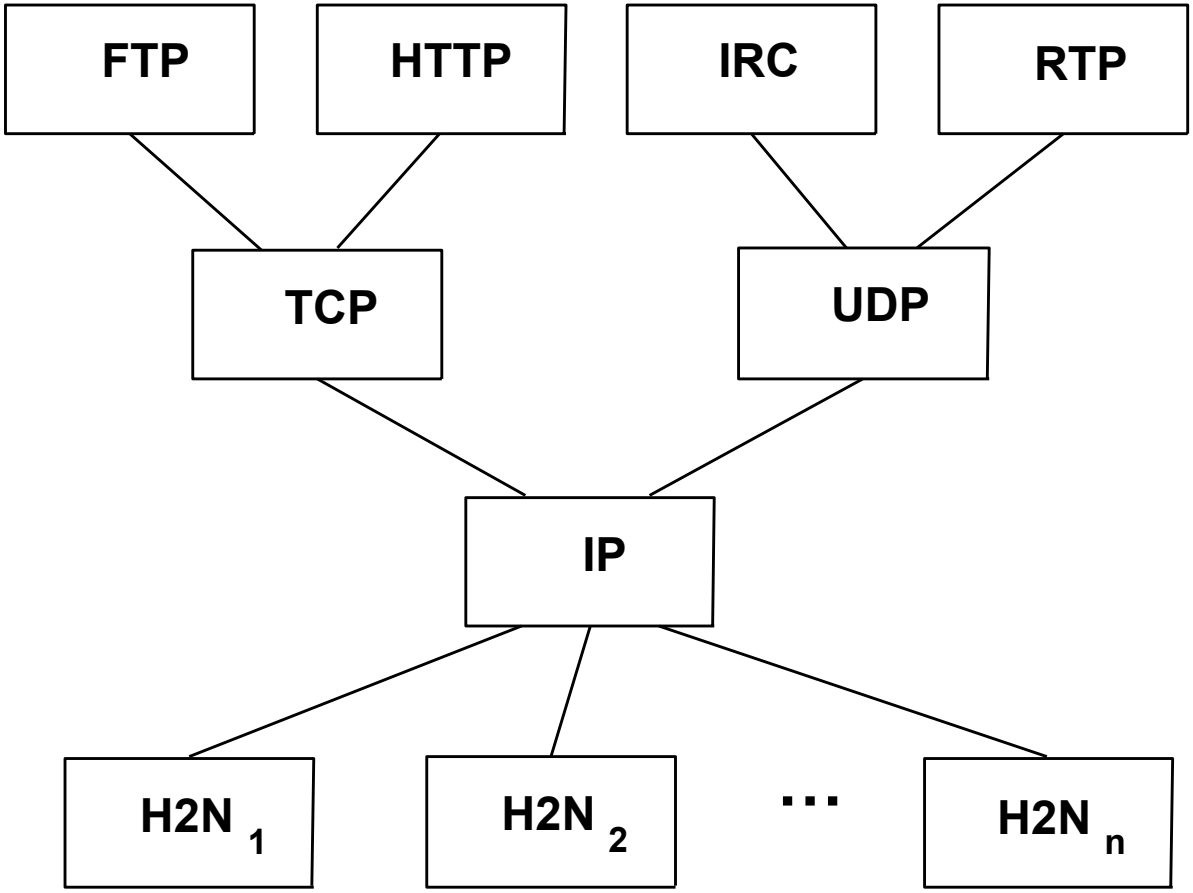
Livello 7 (*application*)

- **Il livello application utilizza il livello di trasporto dell'informazione tra processi in esecuzione su host terminali per realizzare applicazioni di rete**

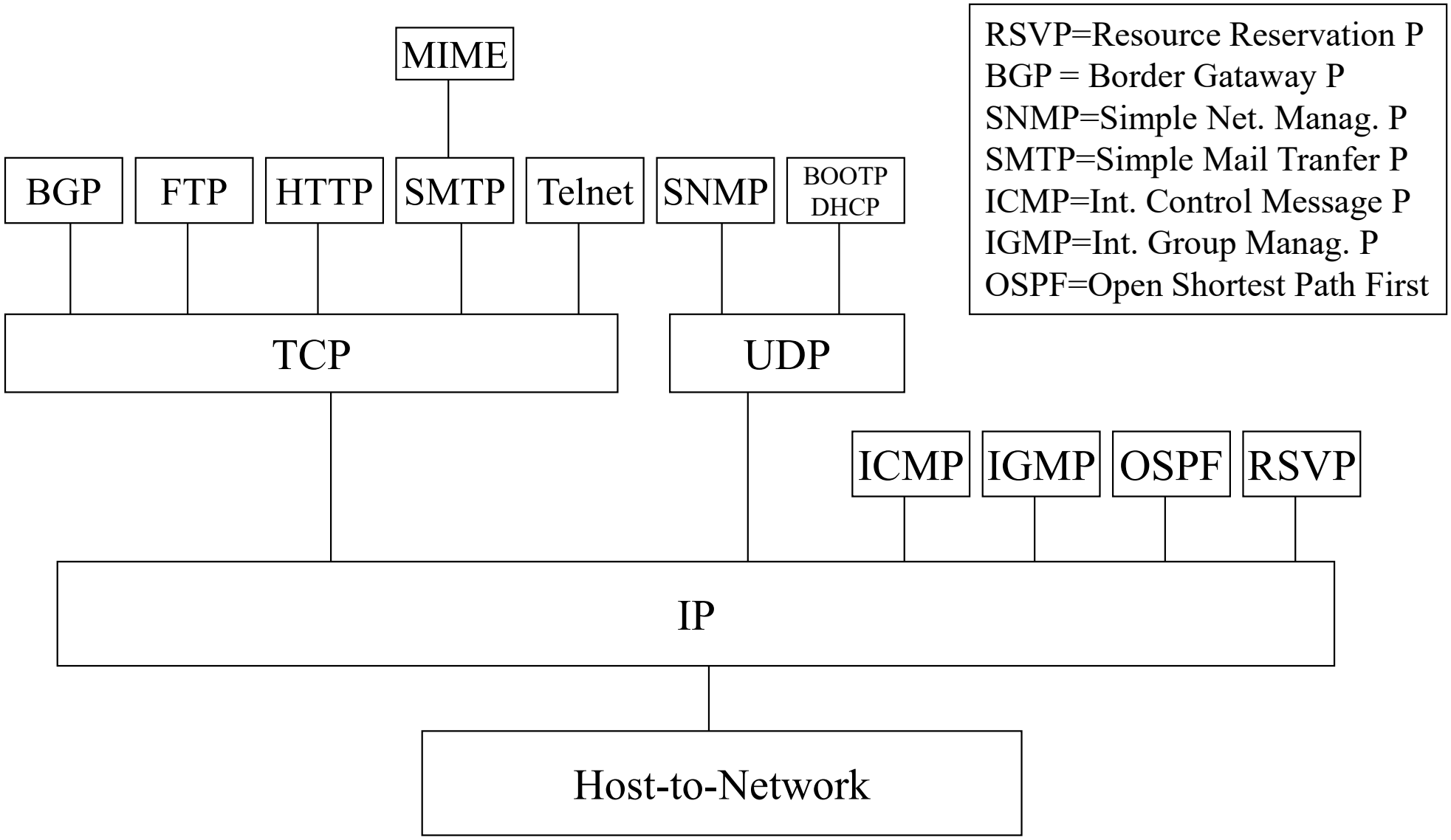
- Esempi protocolli applicativi
  - **ftp**
  - **telnet**
  - **http**
  - **smtp**
  - **irc**
  - …

NOTA: Applicazioni di rete≠protocolli applicativi

Progetto Internet “a clessidra”

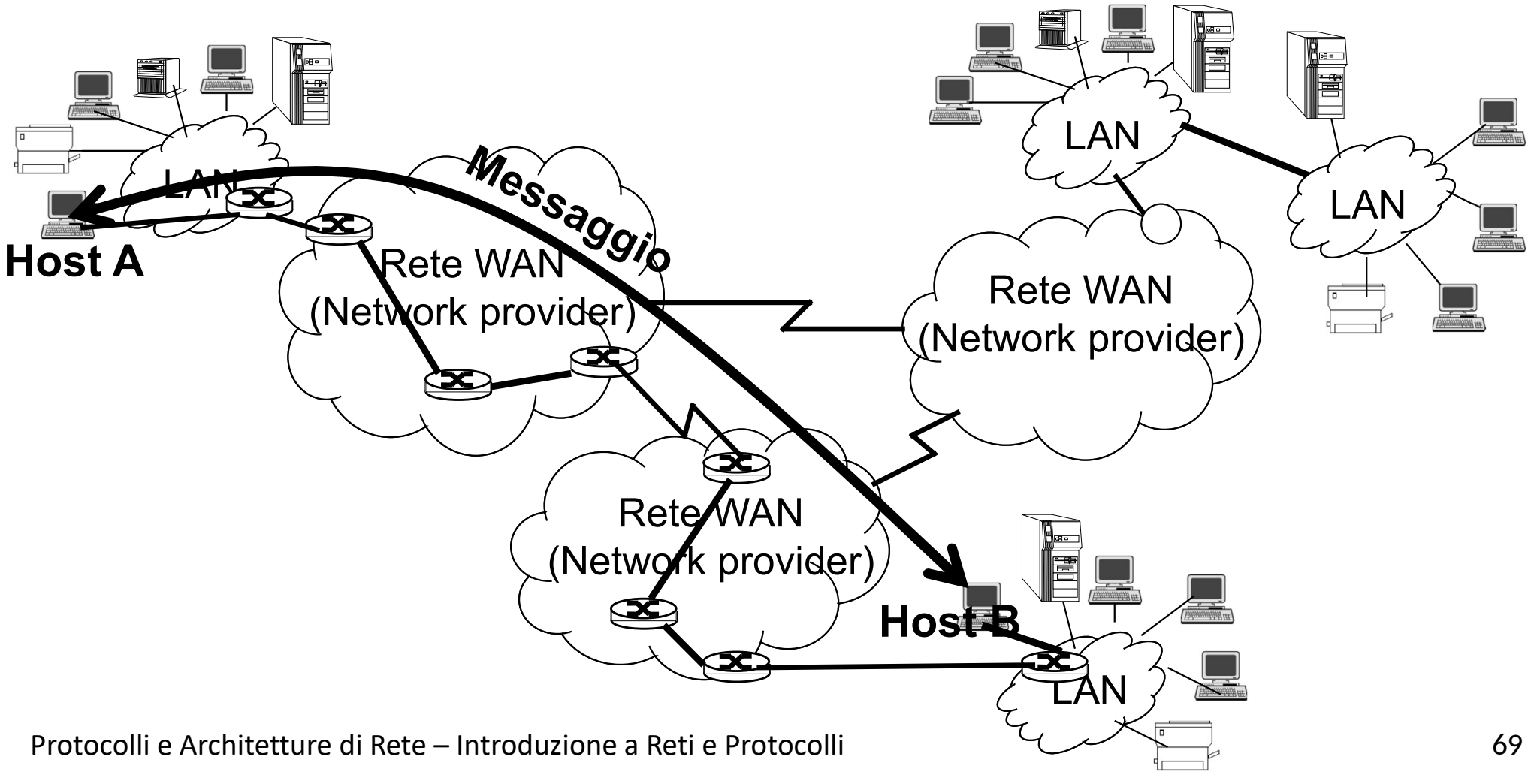


Molti protocolli, ma non a tutti i livelli



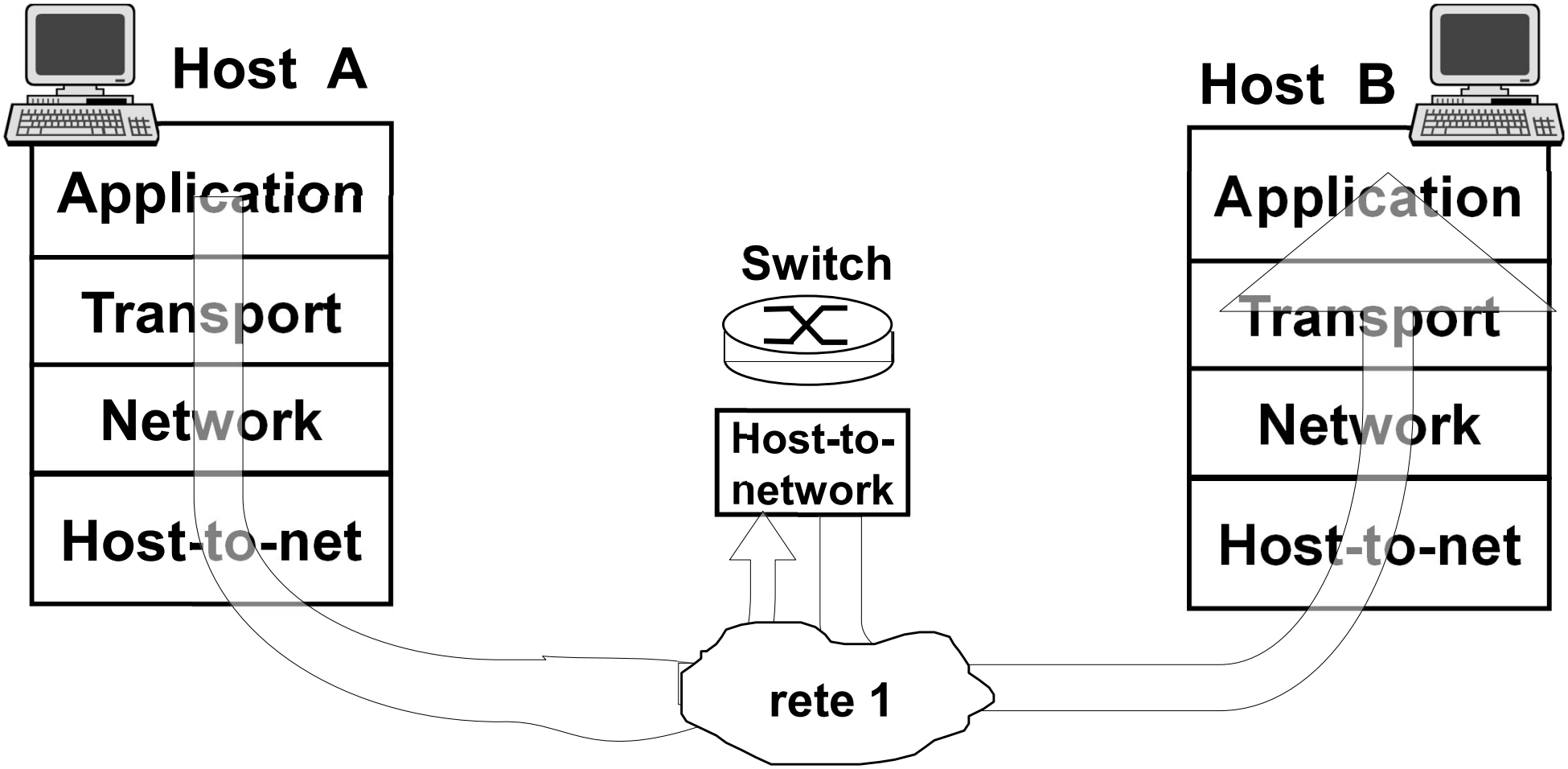
Comunicazione in Internet

Ogni pacchetto deve attraversare nodi che analizzano il pacchetto a «profondità» differenti



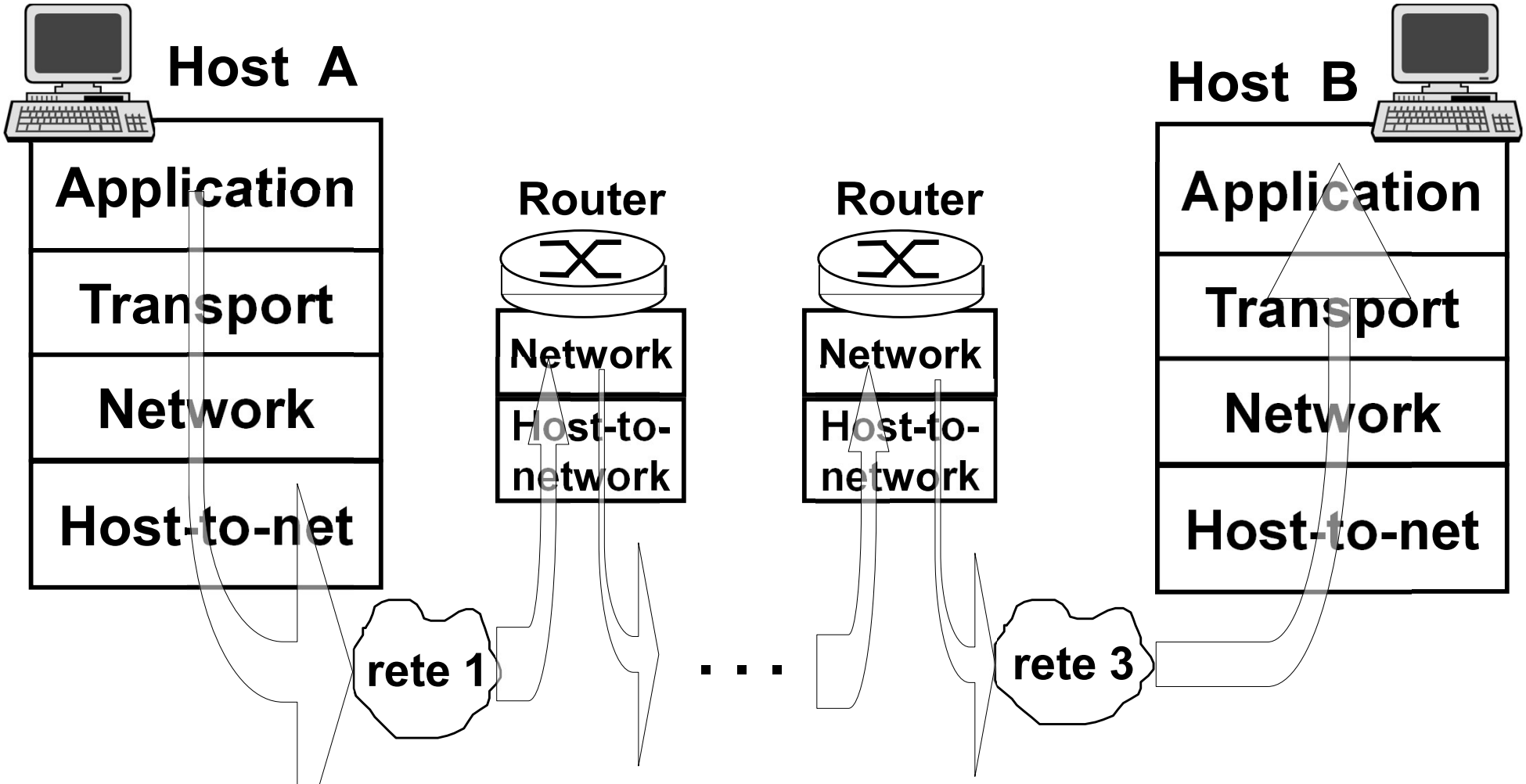
Comunicazioni in Internet [*vista 3*]

Un host implementa tutto lo stack
Switch e bridge sono nodi di livello 2 – H2N



Comunicazioni in Internet [*vista 4*]

Un router è un nodo di livello 3: legge fino all'header IP



Norma ed eccezioni

Quindi di norma, **un dispositivo si dice di «livello N» per indicare fino a quale header dello stack dei protocolli è in grado di ispezionare il pacchetto informativo**

- **Switch, Bridge**: nodi di livello 2 (host-to-network)
- **Router**: nodi di livello 3 (IP)

Esistono delle **eccezioni**, ad esempio (approfondiremo):

- **Switch di livello 3** : uno switch che è in grado di leggere alcune informazioni dell'header IP
- **Router con funzionalità di port address translation per effettuare NAT,** leggono anche informazioni del livello 4
- **Deep packet inspection**: router (di solito con funzionalità di firewall) che analizza anche il livello applicativo