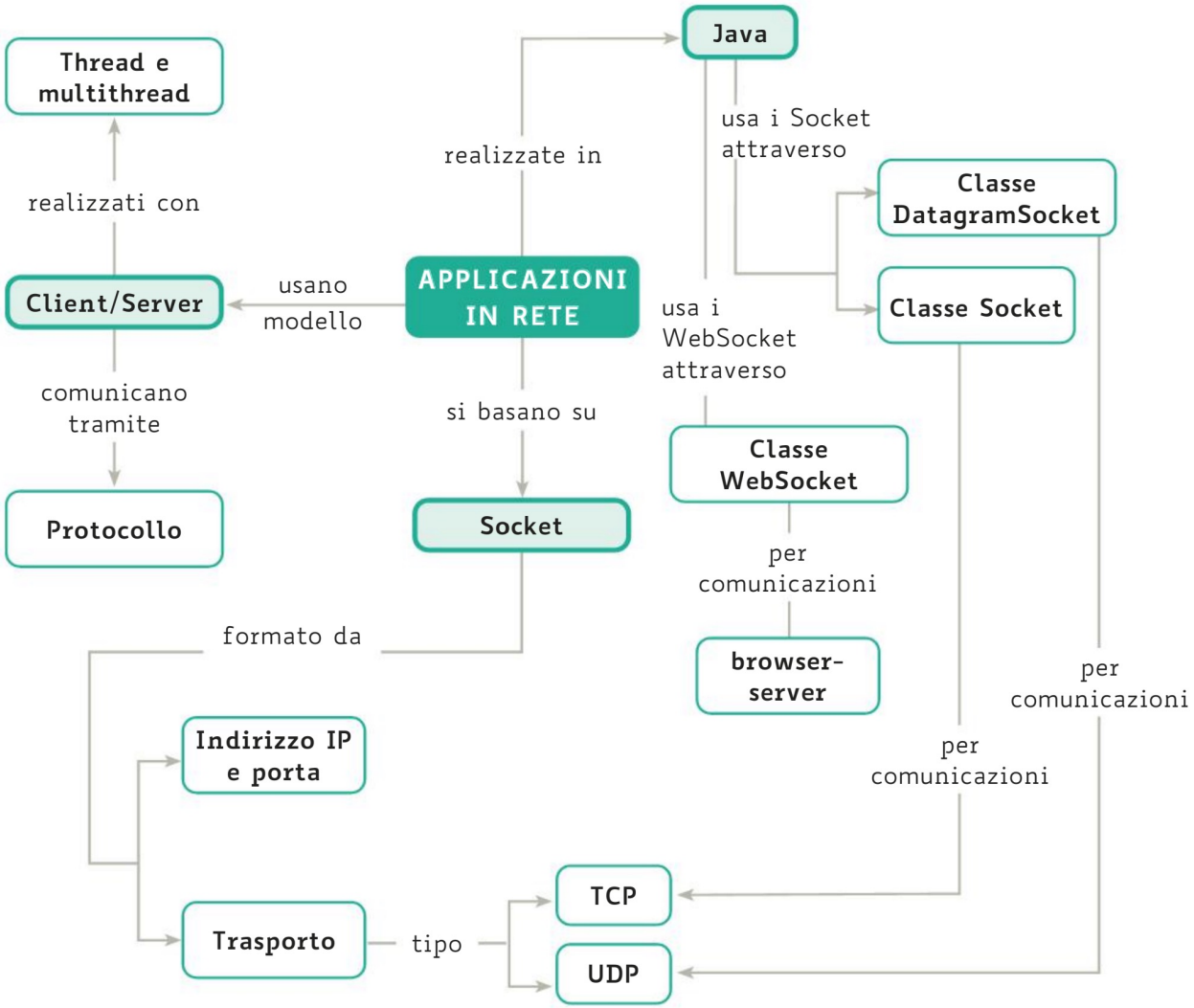
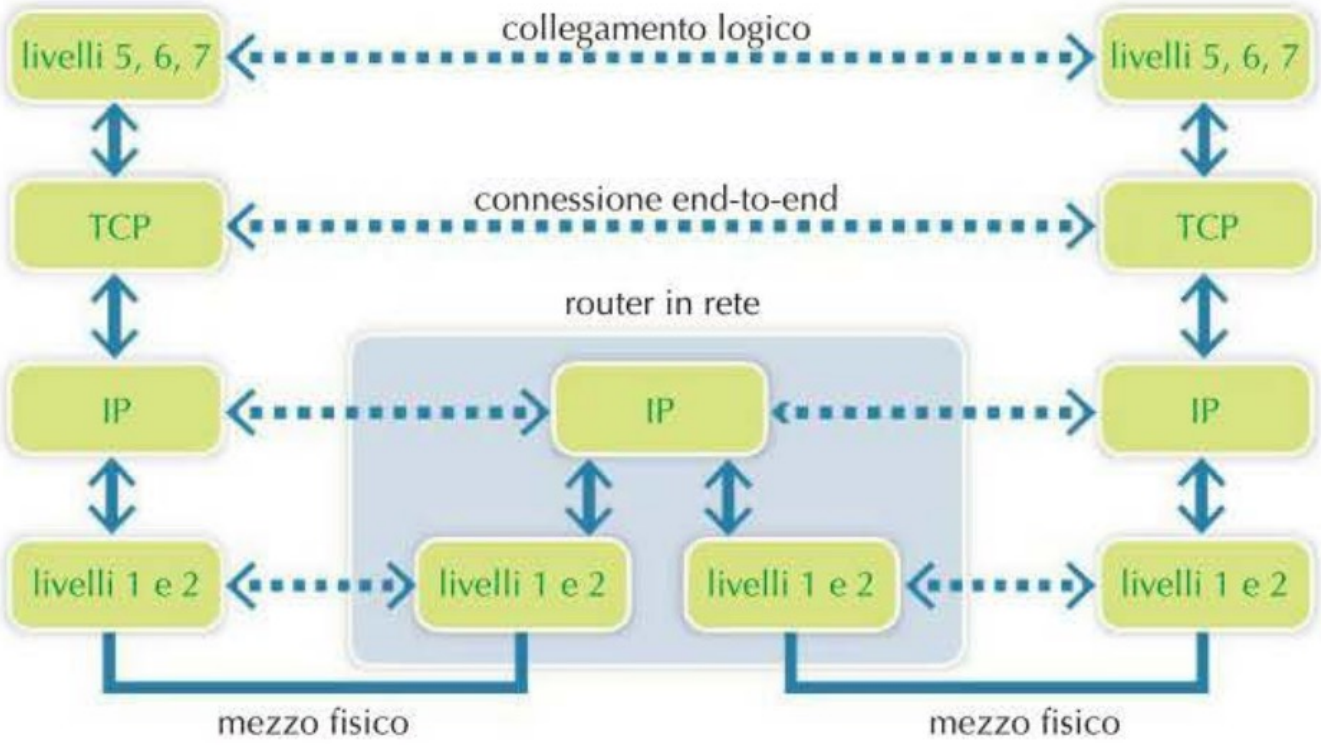

Thread e multithread
realizzati con
Client/Server
usano modello
APPLICAZIONI IN RETE
realizzate in
Java
usa i Socket attraverso
Classe DatagramSocket
Classe Socket
usa i WebSocket attraverso
comunicano tramite
Protocollo
si basano su
Socket
Classe WebSocket
per comunicazioni
browser-server
formato da
Indirizzo IP e porta
Trasporto
tipo
TCP
UDP
per comunicazioni
per comunicazioni

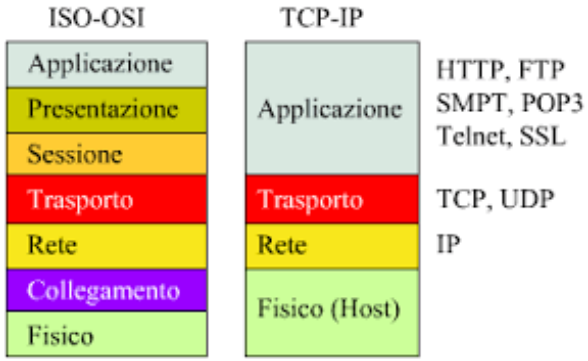
Definizione protocollo

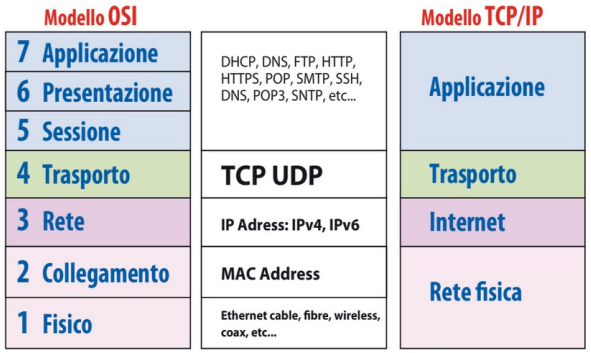
E’ l’insieme di regole utilizzate da due entità per scambiarsi informazioni, specificando cosa deve essere comunicato, in che modo e quando.

Degli esempi di protocolli sono quelli utilizzati nell’architettura di rete a strati TCP/IP (ISO/OSI).

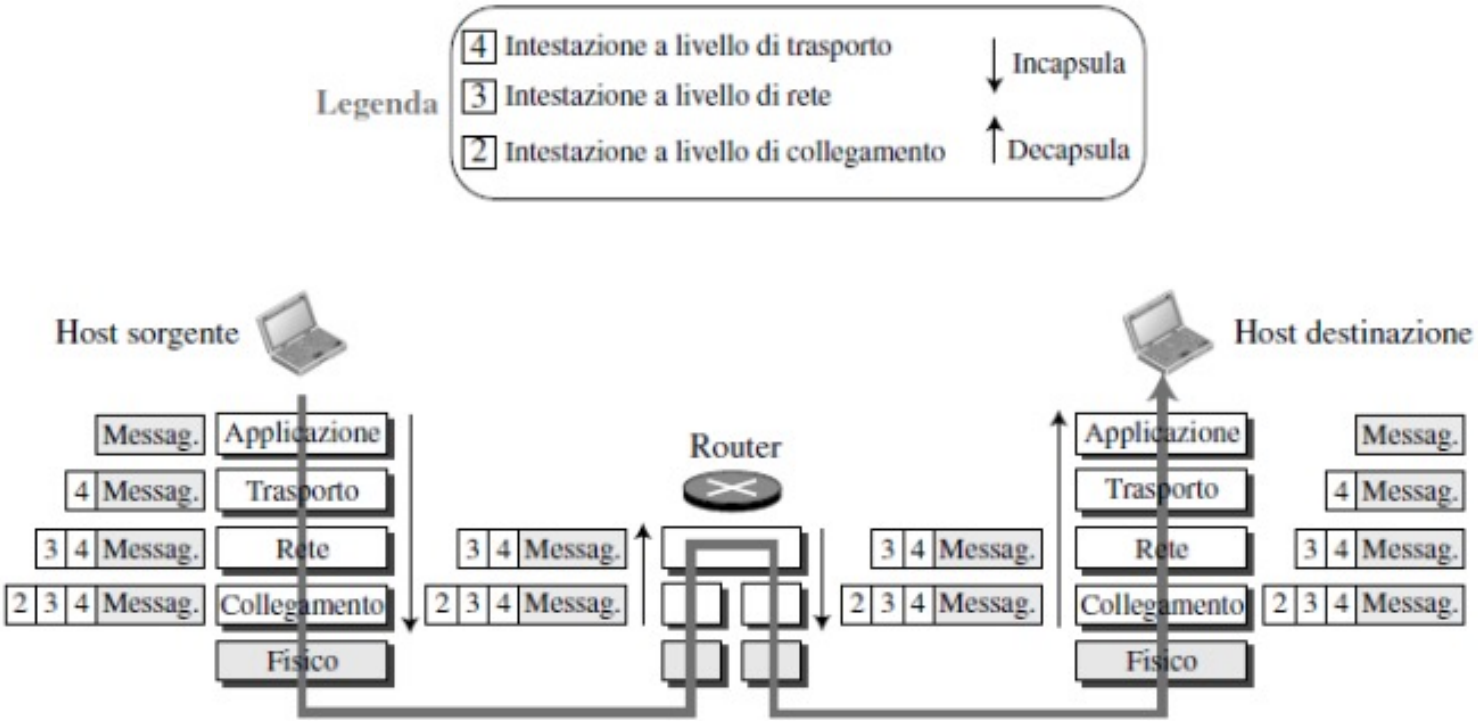
TCP/IP – ISO/OSI





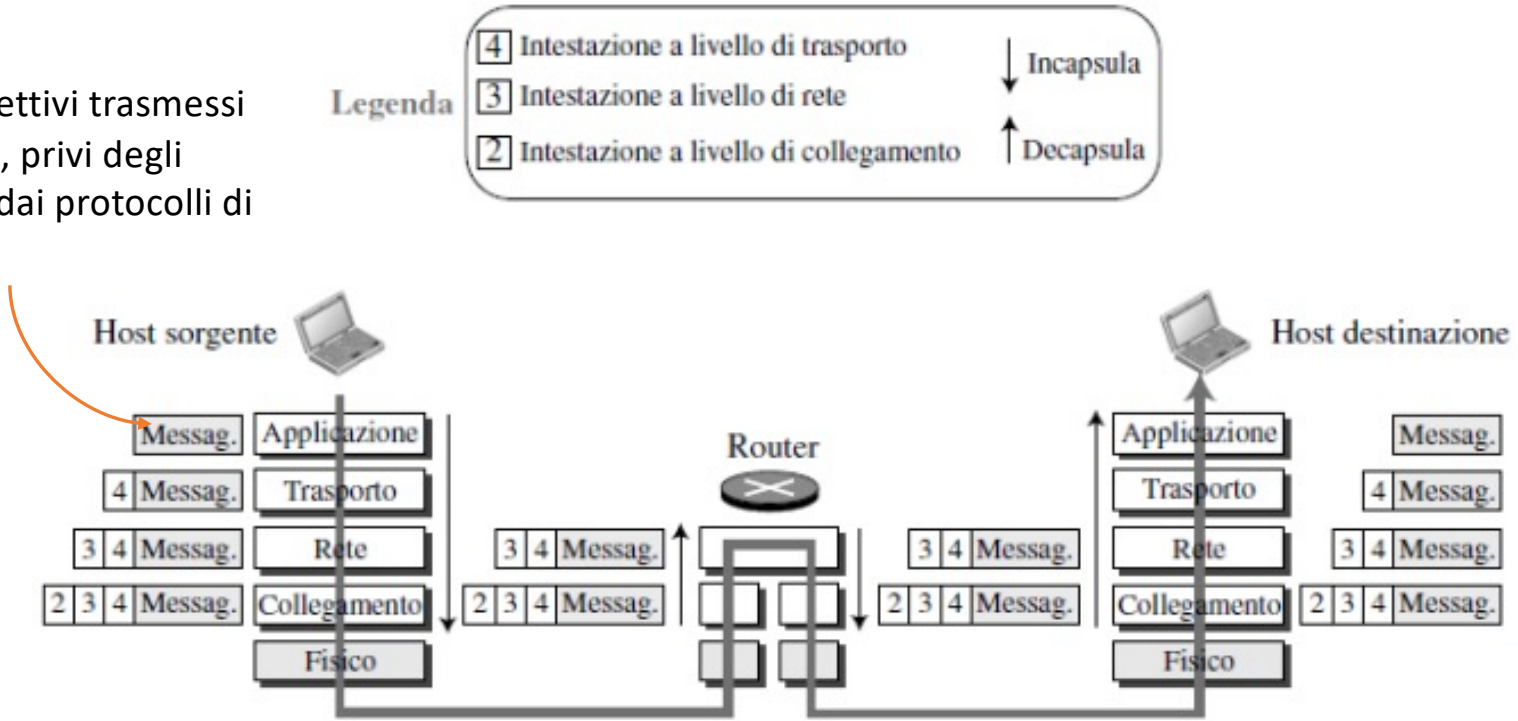


Comunicazione tra due host in rete

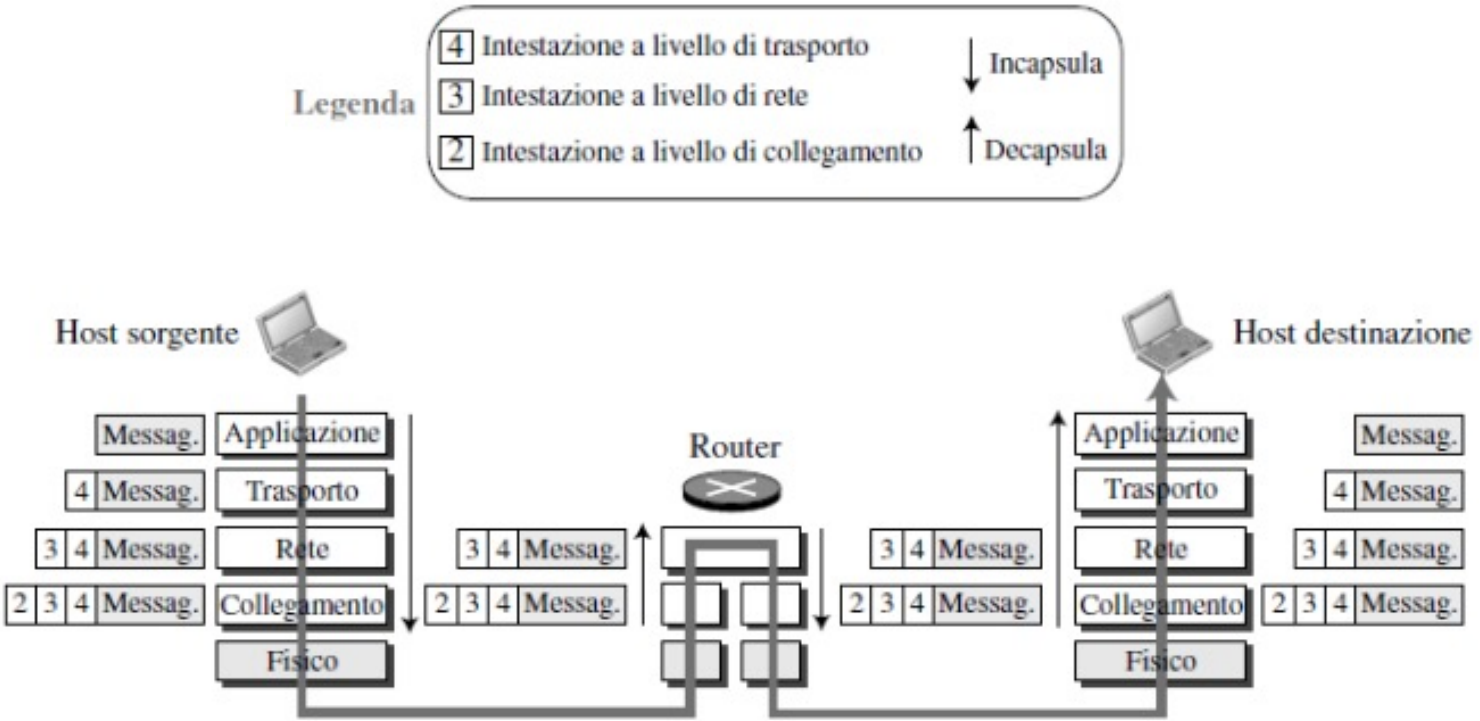


Comunicazione tra due host in rete

Payload: dati effettivi trasmessi dall'applicazione, privi degli header aggiunti dai protocolli di rete.



Comunicazione tra due host in rete



PDU

Segment o User Datagram

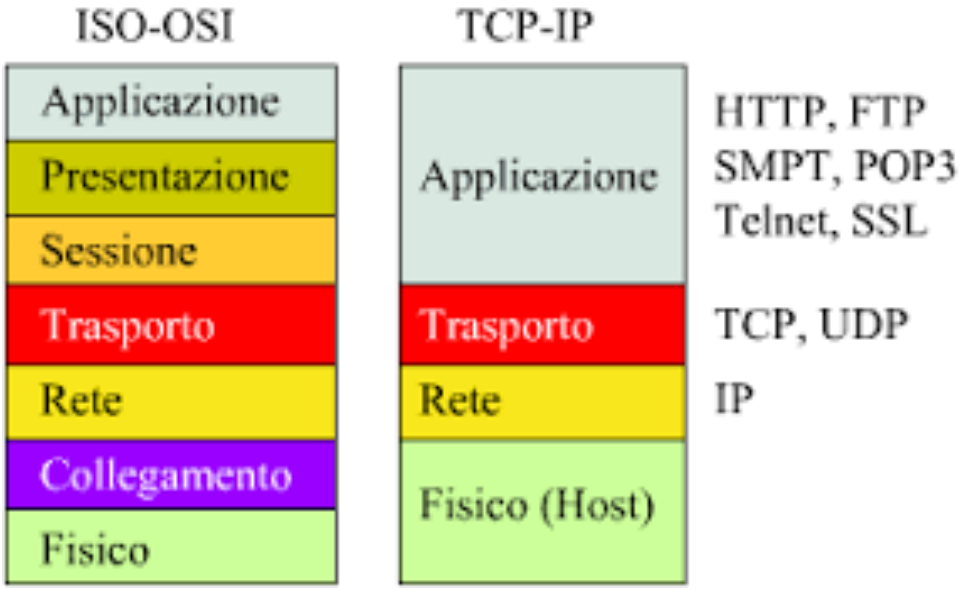
IP datagram

Frame

Segnale

Livello applicazioni di rete: si usano I protocolli applicativi già esistenti per programmare applicazioni utente

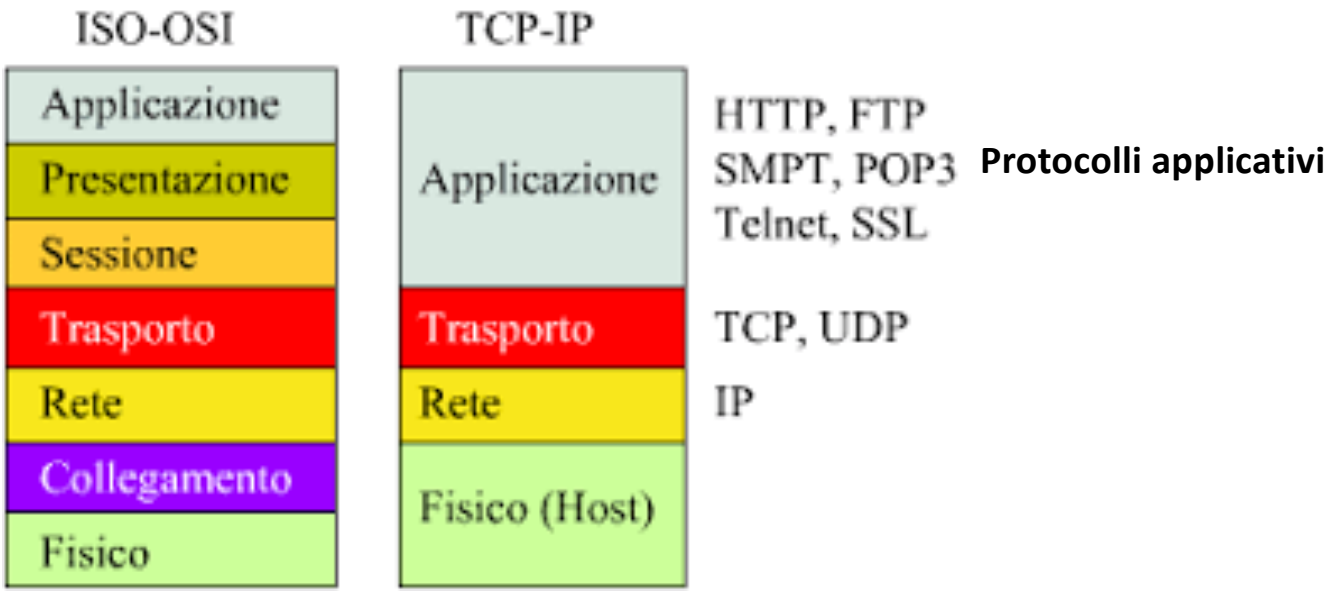
Web, Posta Elettronica, condivisione file P2P, giochi come MMORPG, autenticazione su macchina remota… sono tutti esempi di insiemi di programmi che usano i protocolli applicativi per fornire un servizio **distribuito** in rete



Protocolli applicativi

Noi prima progetteremo e svilupperemo nostri protocolli applicativi (semplici)

In seguito, studieremo e useremo protocolli applicative esistenti per progettare e sviluppare le nostre applicazioni in rete

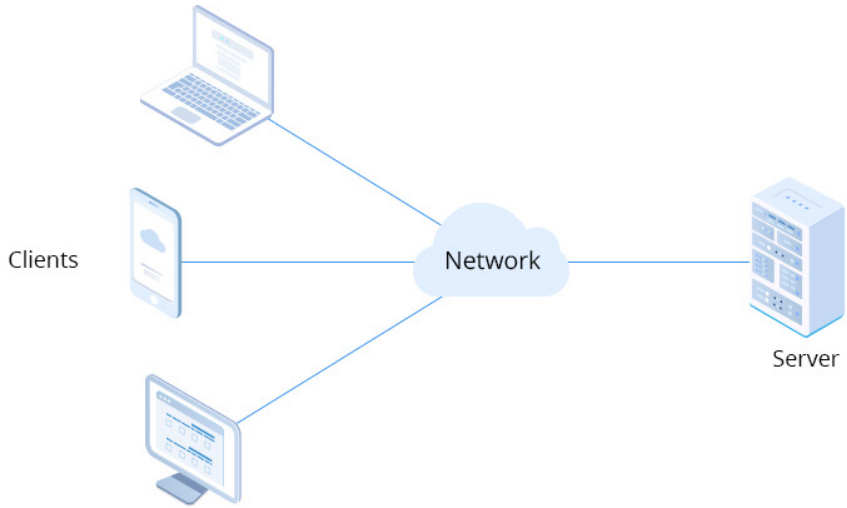


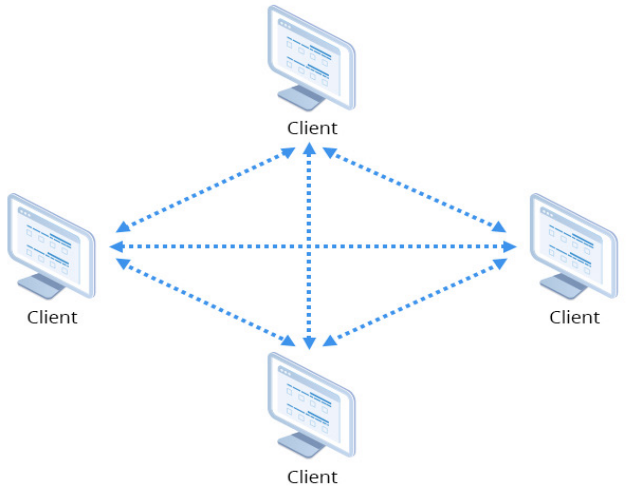
Comunicazioni client e server, peer to peer

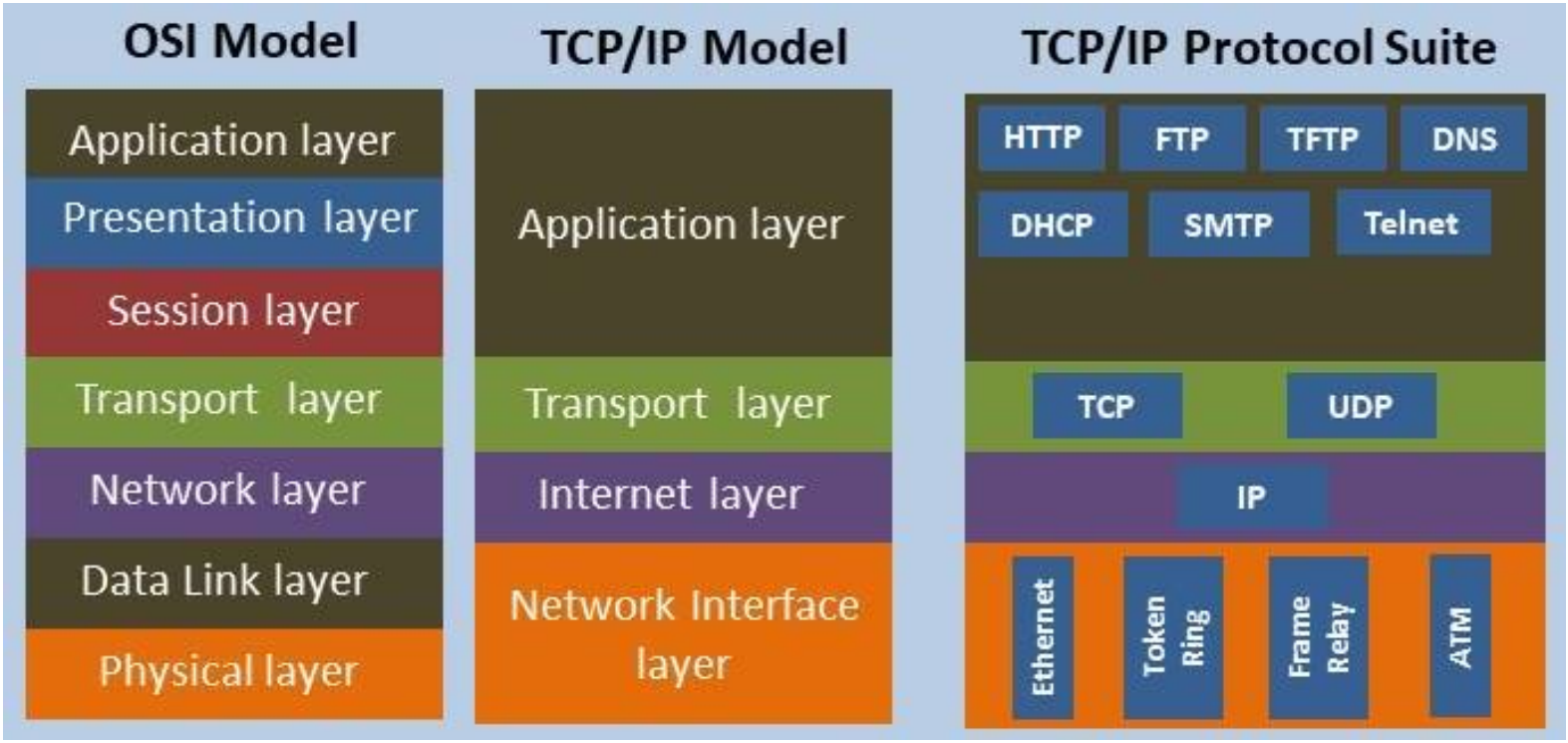
Da mittente/destinario a client/server

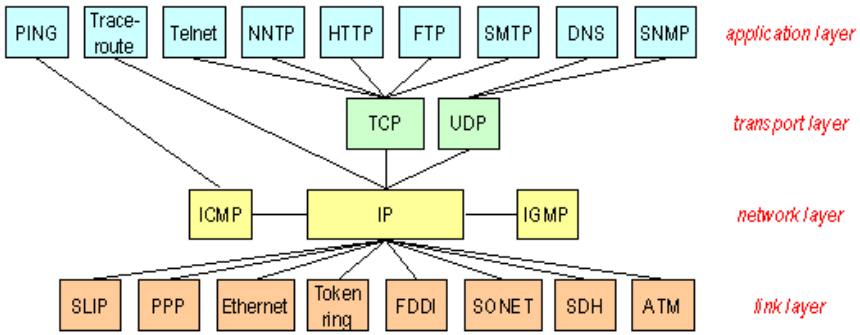
Non è unico modello di programmazione in rete

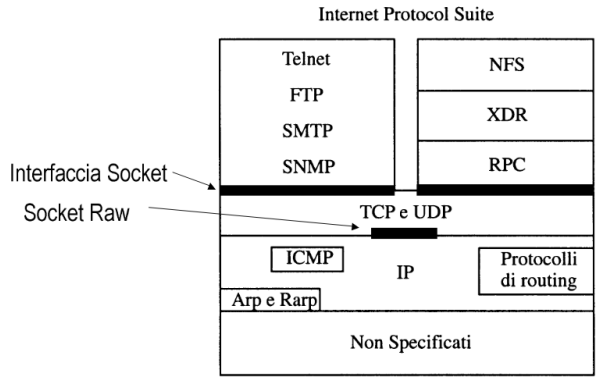
- Peer to peer
- Modelli multi-tier (estensione del modello client-server)

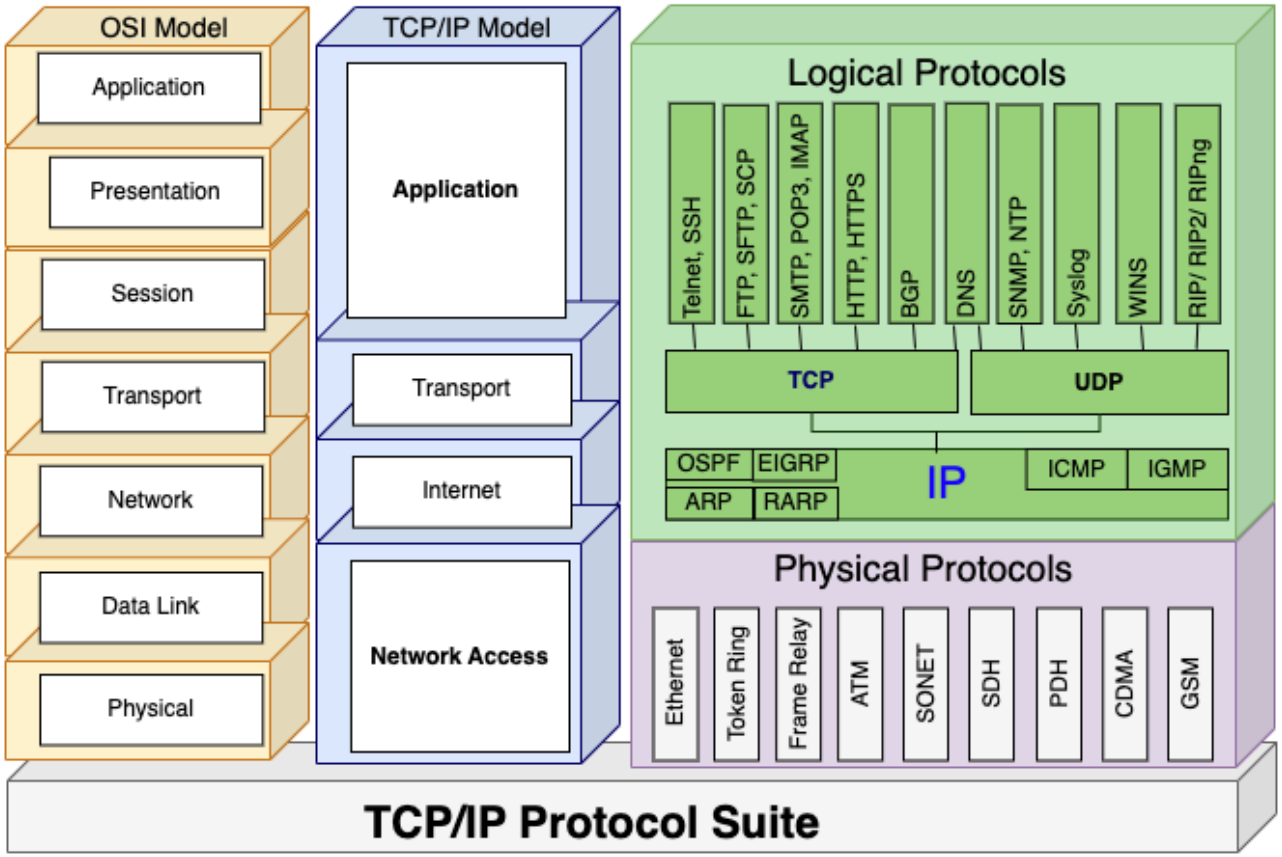





OSI Model
Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data Link layer
Physical layer
TCP/IP Model
Application layer
Transport layer
Internet layer
Network Interface layer
TCP/IP Protocol Suite
HTTP
FTP
TFTP
DNS
DHCP
SMTP
Telnet
TCP
UDP
IP
Ethernet
Token Ring
Frame Relay
ATM


PING
Trace-route
Telnet
NNTP
HTTP
FTP
SMTP
DNS
SNMP
application layer
TCP
UDP
transport layer
ICMP
IP
IGMP
network layer
SLIP
PPP
Ethernet
Token ring
FDDI
SONET
SDH
ATM
link layer


Internet Protocol Suite
Telnet
FTP
SMTP
SNMP
NFS
XDR
RPC
Interfaccia Socket
Socket Raw
TCP e UDP
ICMP
IP
Protocolli di routing
Arp e Rarp
Non Specificati


OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
TCP/IP Model
Application
Transport
Internet
Network Access
Logical Protocols
Telnet, SSH
FTP, SFTP, SCP
SMTP, POP3, IMAP
HTTP, HTTPS
BGP
DNS
SNMP, NTP
Syslog
WINS
RIP/ RIP2/ RIPng
TCP
UDP
OSPF
EIGRP
ARP
RARP
IP
ICMP
IGMP
Physical Protocols
Ethernet
Token Ring
Frame Relay
ATM
SONET
SDH
PDH
CDMA
GSM
TCP/IP Protocol Suite

Progettazione protocollo

Progettare protocollo applicativo

Cos’è RFC?

- Definizione documento Request For Comments
- IETF RFC
  - RFC Editor
  - Lista di tutti gli RFC

RFC Esempi

Protocolli “famosi” più complessi

RFC 791 Internet Protocol
RFC 768 User Datagram Protocol
RFC 793 Transmission Control Protocol
RFC 783 Trivial File Transfer Protocol
RFC 913 - Simple File Ttransfer Protocol
RFC 172 File Transfer Protocol
RFC 1945 - Hyper Text Transfer Protocol
RFC 764 Telnet Protocol

Protocolli più semplici di debug e test

RFC 862 - Echo Protocol
RFC 863 - Discard Protocol
RFC 865 - Quote of the Day Protocol
RFC 867 - Daytime Protocol
RFC 868 - Time Protocol

Non solo protocolli, anche standard formati

RFC 20 - ASCII format for network interchange
RFC 8259 - The JavaScript Object Notation (JSON) Data Interchange Format

Aspetti da considerare progettazione protocollo applicativo

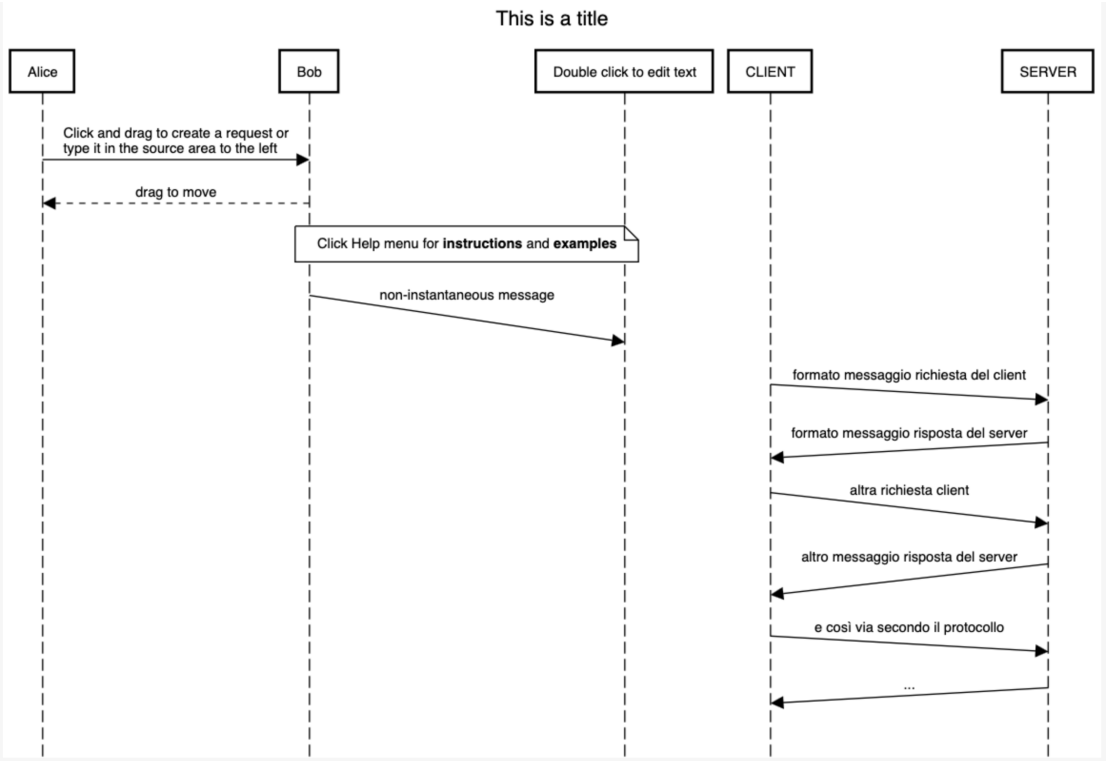
- Scelta protocollo di trasporto (UDP vs TCP)
- Definizioni di tutti o parte dei seguenti aspetti
  - **Indirizzamento**: Come identificare client e server.
  - **Frammentazione e riassemblaggio**: Gestione di dati di grandi dimensioni.
  - **Incapsulamento**: Formato dei messaggi (header, corpo, ecc.).
  - **Controllo della Connessione**: Gestione delle sessioni e stabilità delle connessioni.
  - **Servizio Confermato o Non Confermato**: Utilizzo di protocolli come TCP (confermati) o UDP (non confermati).
  - **Controllo degli Errori**: Meccanismi per garantire l'integrità dei dati.
  - **Controllo del Flusso**: Prevenzione di sovraccarichi di dati.
  - **Multiplexing e Demultiplexing**: Gestione di più flussi di comunicazione.
  - **Servizi di Trasmissione**: Affidabilità, ordine, ritrasmissione.

Echo Protocol RFC

- Fare sequence diagram echo protocol
- https://sequencediagram.org/

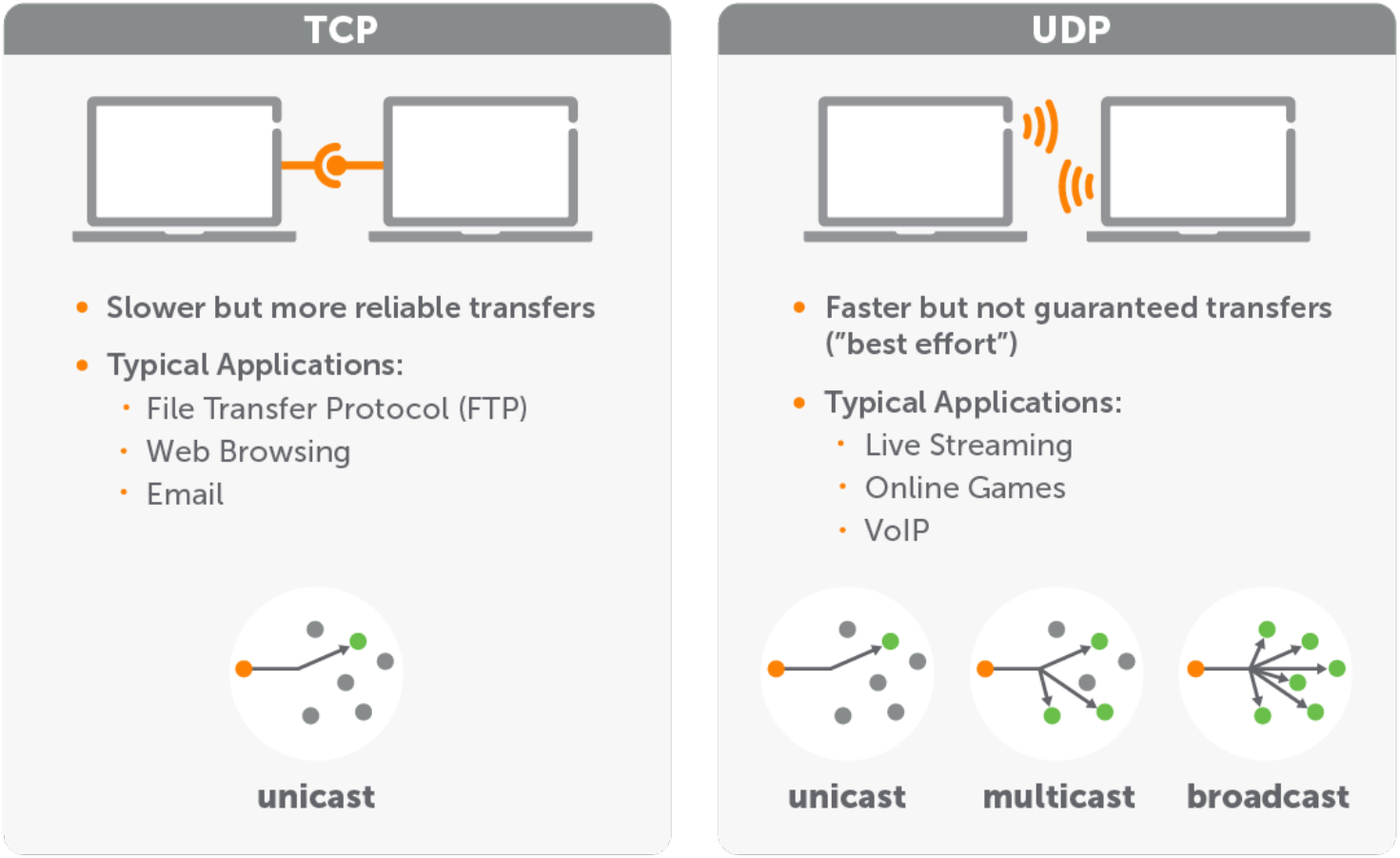
Esempio

```
title This is a title


Alice->Bob:Click and drag to create a request
or\ntype it in the source area to the left

Alice<--Bob:drag to move

note over Bob,Double click to edit text:Click
Help menu for **instructions** and **examples**

Bob->(3)Double click to edit text:non-
instantaneous message

CLIENT->(1)SERVER: formato messaggio richiesta
del client

SERVER->(1)CLIENT: formato messaggio risposta del
server

CLIENT->(2)SERVER: altra richiesta client

SERVER->(2)CLIENT: altro messaggio risposta del
server

CLIENT->(1)SERVER: e così via secondo il
protocollo

SERVER->(1)CLIENT: ...
```

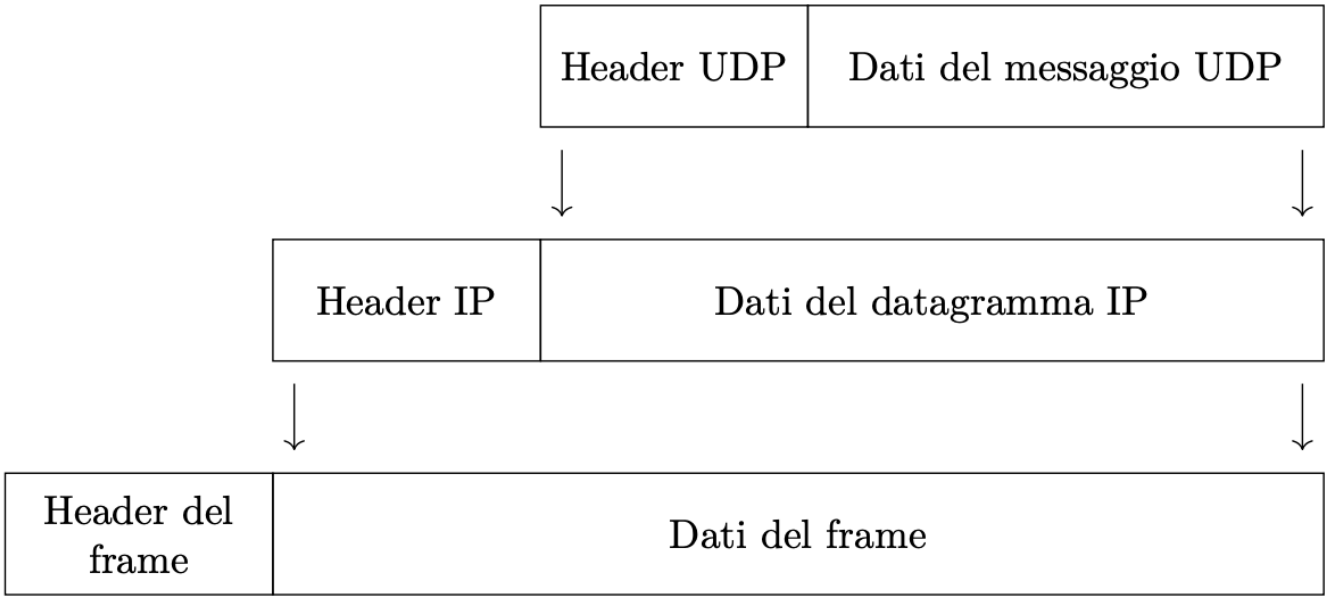


Ripasso

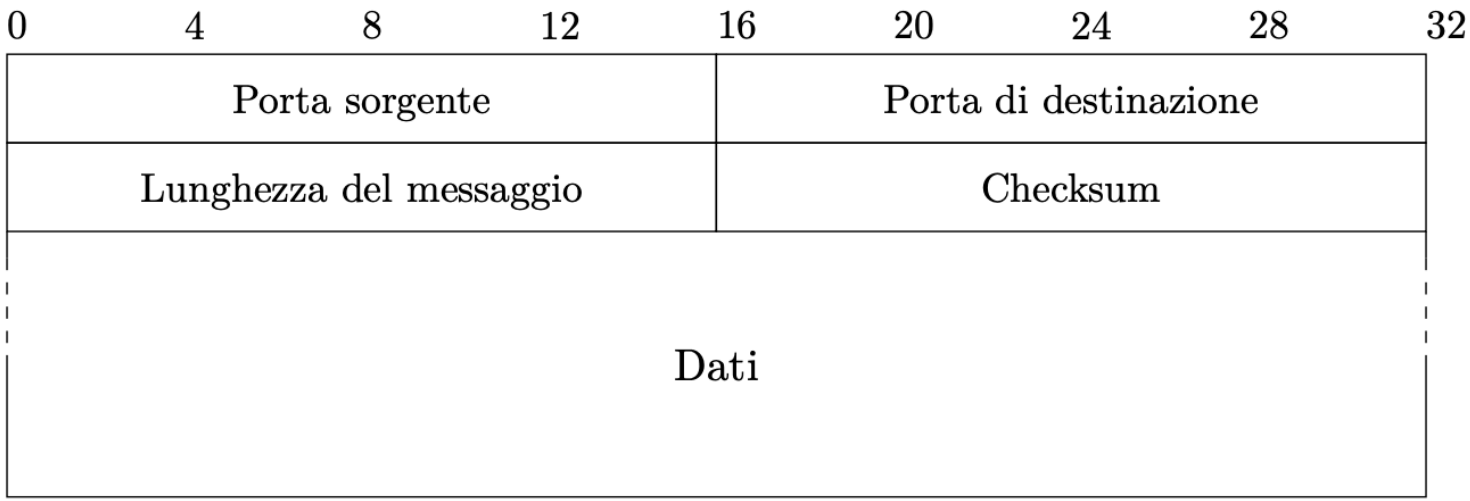
Livello di trasporto – TCP vs UDP



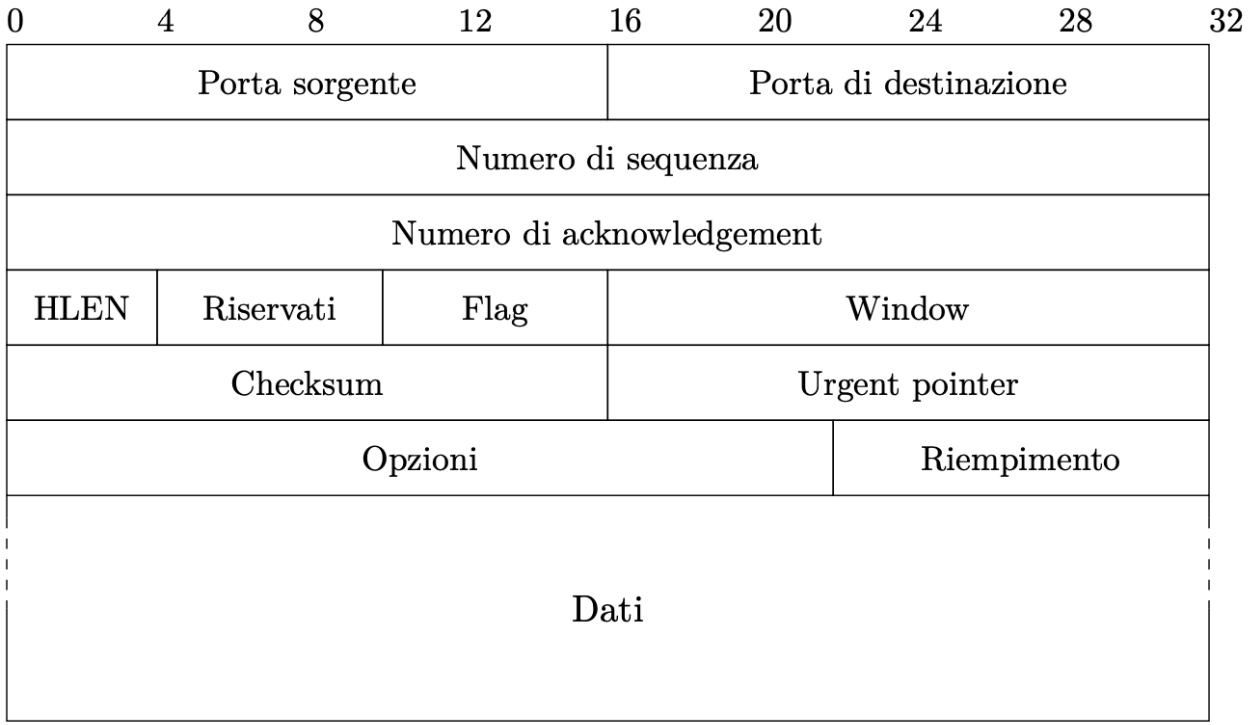
Incapsulamento



UDP


0
4
8
12
16
20
24
28
32
Porta sorgente
Porta di destinazione
Lunghezza del messaggio
Checksum
Dati

TCP


0
4
8
12
16
20
24
28
32
Porta sorgente
Porta di destinazione
Numero di sequenza
Numero di acknowledgement
HLEN
Riservati
Flag
Window
Checksum
Urgent pointer
Opzioni
Riempimento
Dati

Connessione end to end, socket e porte

Connessione end to end: connessione logica bidirezionale tra due processi applicativi su host diversi, che permette lo scambio affidabile di dati attraverso la rete utilizzando il protocollo TCP.

Socket: Endpoint di comunicazione identificato da una coppia <indirizzo IP, numero di porta>. Rappresenta l'interfaccia tra il livello applicativo e il livello di trasporto.

Porta: Numero a 16 bit che identifica un processo specifico su un host. Consente il multiplexing delle connessioni, permettendo a un singolo host di gestire multiple comunicazioni simultanee.

Connessione Logica a Livello di Trasporto

- Una connessione logica end to end tra due processi su una rete è identificata da una quintupla composta da cinque elementi fondamentali:

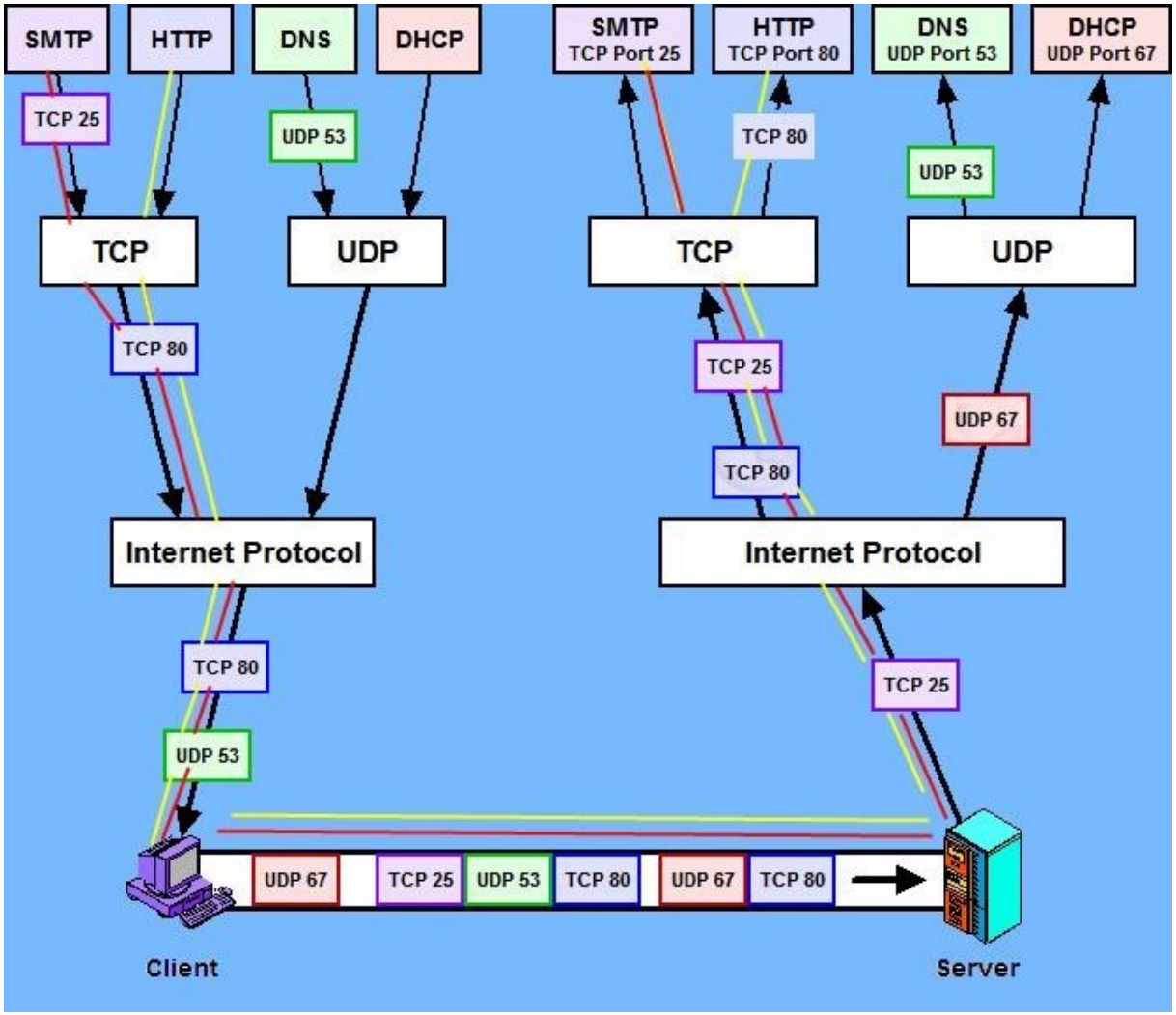
  <protocollo, IP_sorgente, porta_sorgente, IP_destinazione, porta_destinazione>

- Elementi simmetrici della comunicazione:

  Mittente → identificato da <IP_sorgente, porta_sorgente>

  Destinatario → identificato da <IP_destinazione, porta_destinazione>

Multiplexing/demultiplexing


SMTP
HTTP
DNS
DHCP
SMTP
TCP Port 25
HTTP
TCP Port 80
DNS
UDP Port 53
DHCP
UDP Port 67
TCP 25
UDP 53
TCP 80
UDP 53
TCP
UDP
TCP
UDP
TCP 80
TCP 25
UDP 67
TCP 80
Internet Protocol
Internet Protocol
TCP 80
TCP 25
UDP 53
UDP 67
TCP 25
UDP 53
TCP 80
UDP 67
TCP 80
Client
Server