



UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA E COMUNICAZIONE

Laboratorio Reti di Calcolatori

Laurea Triennale in Comunicazione Digitale

Docente: Matteo Zignani

Lezione 3

Anno accademico 2011/12

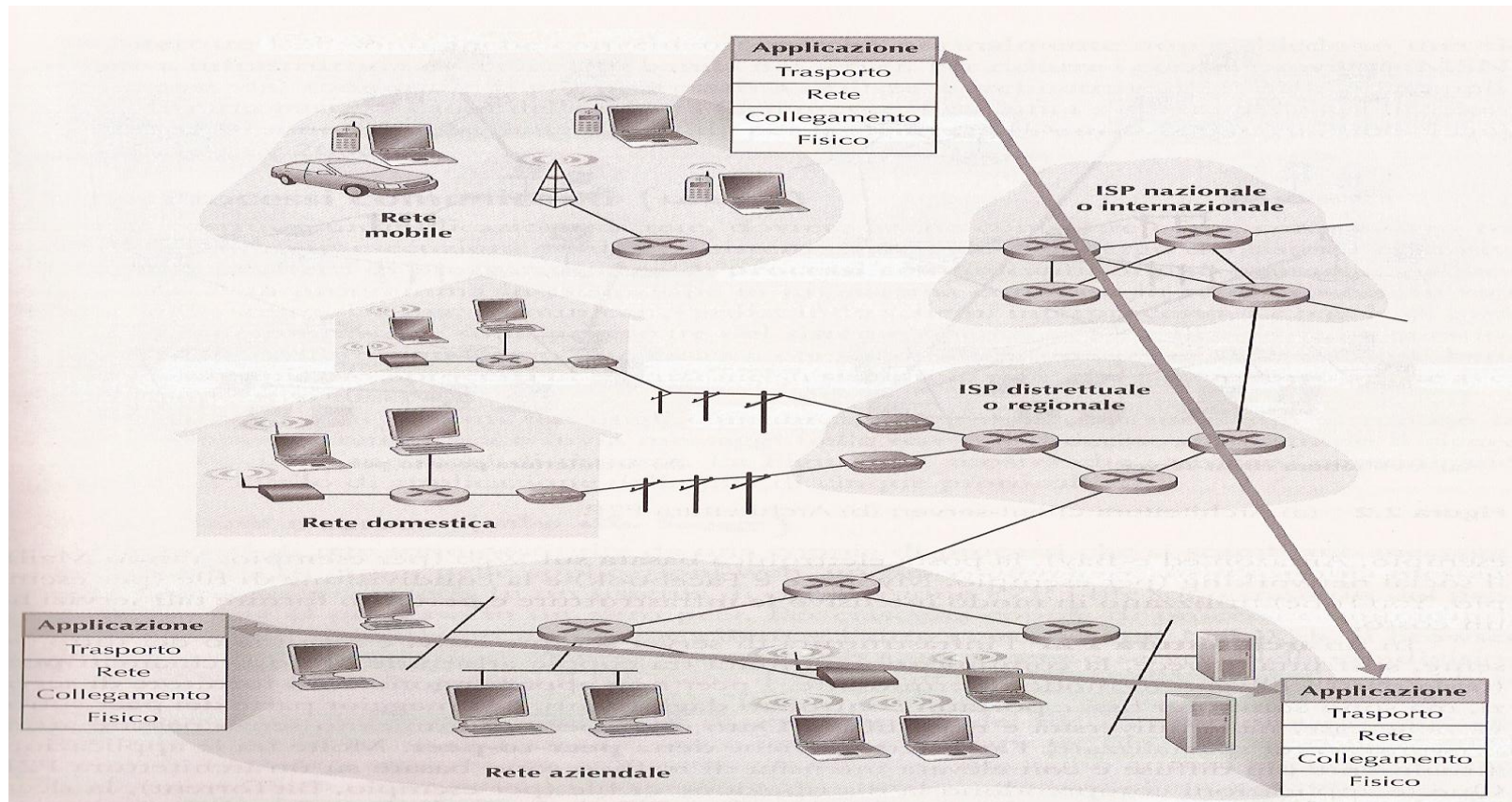
Architetture delle applicazioni di rete

Modello Client-Server



Architettura dell'applicazione

- Sviluppo di applicazioni di rete = compilazione di programmi eseguiti dai terminali che comunicano attraverso rete



Architettura dell'applicazione

- Prima di scrivere codice devo pensare alla sua architettura != architettura di rete
- Architettura di rete è FISSATA e fornisce un insieme di SERVIZI
- Si deve scegliere un MODELLO = definizione funzionale del comportamento del sistema + ORGANIZZAZIONE dell'applicazione sui vari terminali

Architetture più utilizzate

Client-Server

- Host sempre attivo = SERVER, risponde alle richieste di servizio di altri host = CLIENT (non sempre attivi)
- Client non comunicano direttamente tra di loro
- Server ha un indirizzo diffusamente conosciuto

Peer-to-peer (P2P)

- Comunicazione diretta tra coppie arbitrarie di host = PEER
- Peer non appartengono ad un fornitore di servizio
- Peer comunicano senza passare attraverso un server specializzato
- Si possono ibridizzare con modello client-server
- Scalabili = aggiunta di peer richiedo file ma aggiungono capacità di servizio.

Architettura Client-Server

Il saggio dice «Chiedi e ti sarà dato»

Servizi in rete seguono il consiglio del saggio
domanda/risposta

Formalizzazione: architettura client-server

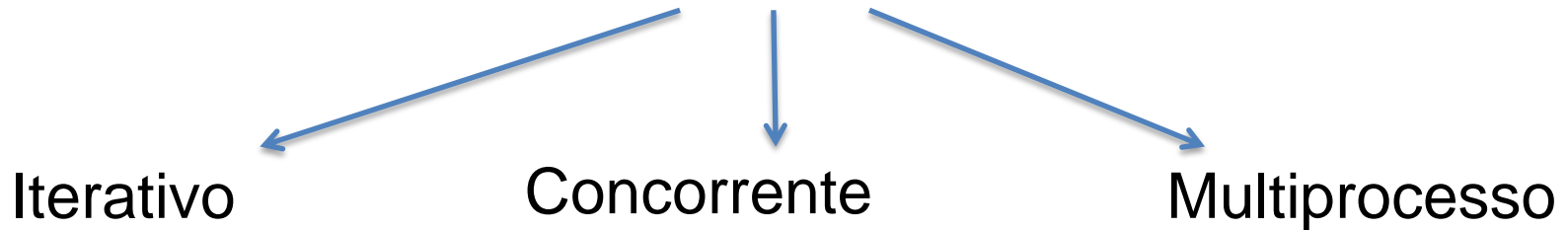
CLIENT: richiede servizio

SERVER: offre e fornisce servizio

Entità definite in base alla relazione

MODELLO DI SERVIZIO

Comportamento di un server al fine di erogare il servizio



Server iterativo

Server fornisce servizio a un singolo client.

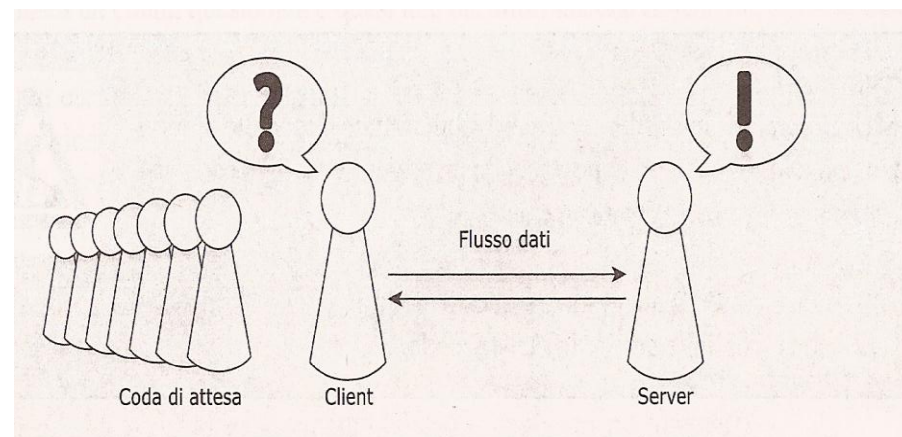
Il client chiede e ...

1. Server libero e client entra in servizio immediatamente
2. Server occupato
 - I. C'è spazio in coda e il client può mettersi in attesa
 - II. La coda è piena e il servizio viene rifiutato

Coda gestita dal SO (server ignora)

Starvation

Attacchi Denial of Service (DoS)

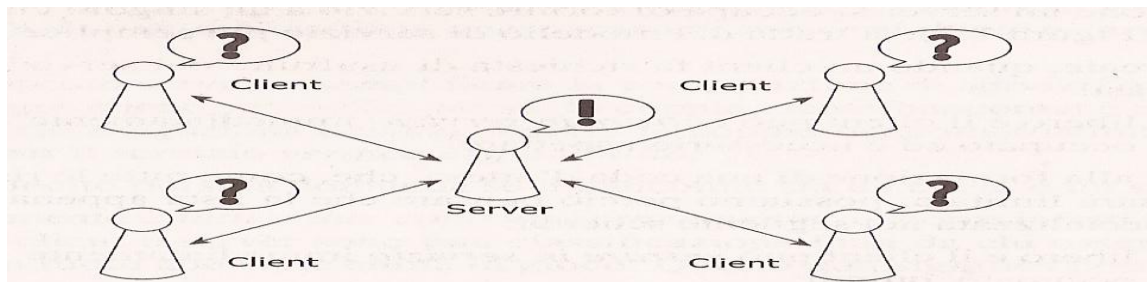


Servizio Concorrente

Server offre servizio parallelamente a più client



Parallelismo nel mantenere i rapporti, non nell'inviare o ricevere dati



Limitazioni sul numero massimo di client in servizio (dipende da SO)

Difficoltà:

- Programma che accentra su di sé tutti i canali di comunicazioni aperti con client
- Tener traccia di tutti i contesti delle diverse connessioni
- Limitazioni scalabilità

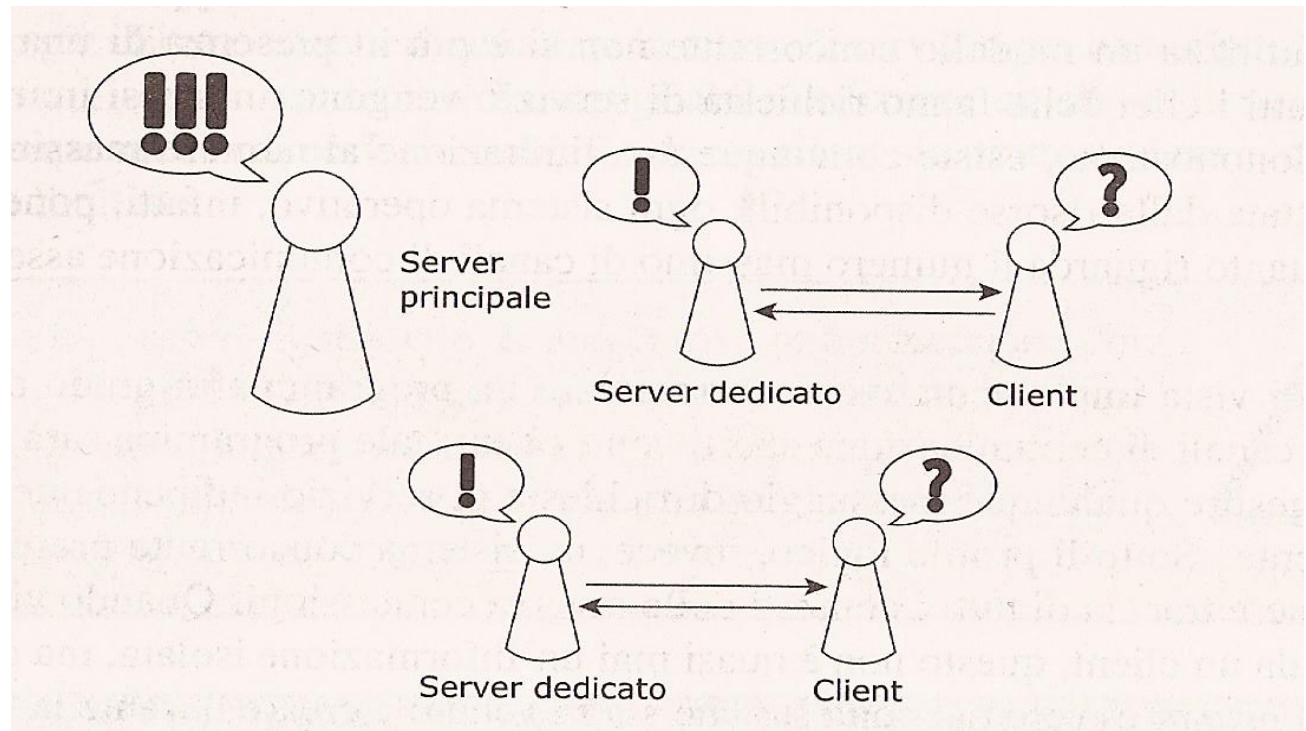
Servizio Multiprocesso/multithread

Più server entrano in gioco contemporaneamente

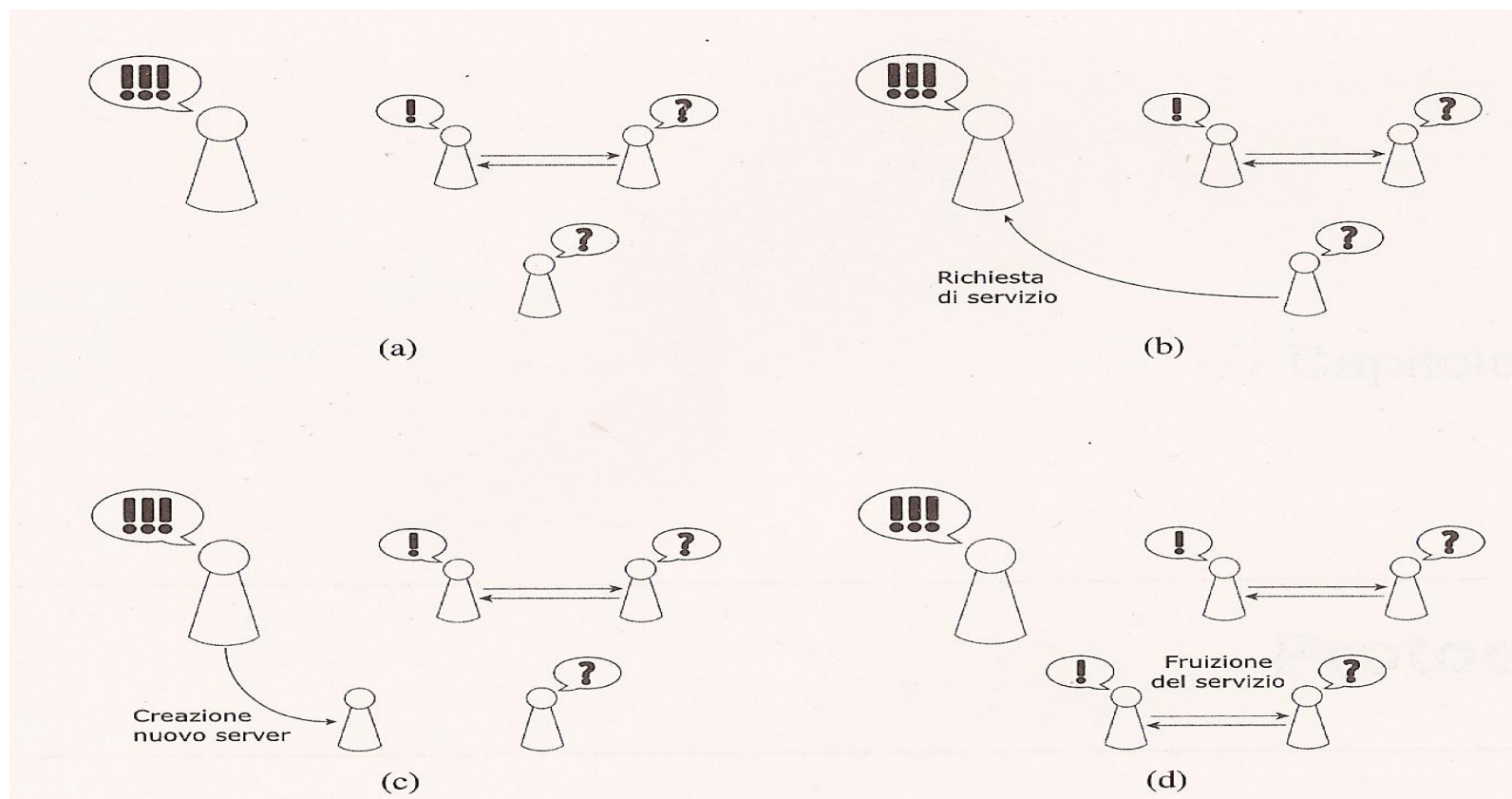


Più flussi esecutivi paralleli per la fornitura del servizio

- Processo
- Thread



Servizio Multiprocesso/multithread



Processi/thread in quantità limitata => Server principale dispone di insieme di processi dormienti che possono venire assegnati (ogni processo gestisce gli arrivi secondo iterativo o concorrente).

Protocolli



- Scopo di una rete: portare messaggio da host A ad host B posti in una maglia di connessioni
 - Regole che A e B devono seguire per poter dialogare tra loro
 - Ogni protocollo di rete definisce una serie di protocolli necessari al funzionamento

Le reti basano la loro struttura sui protocolli

Definizione di un protocollo

- Definizione di protocollo da Dizionario Garzanti
 1. L'insieme delle norme che regolano lo svolgimento di manifestazioni, visite, ricevimenti ufficiali;
 2. Insieme di regole che governano la successione e lo scambio di informazioni fra due dispositivi comunicanti tra loro.

Definizione di un protocollo (2)

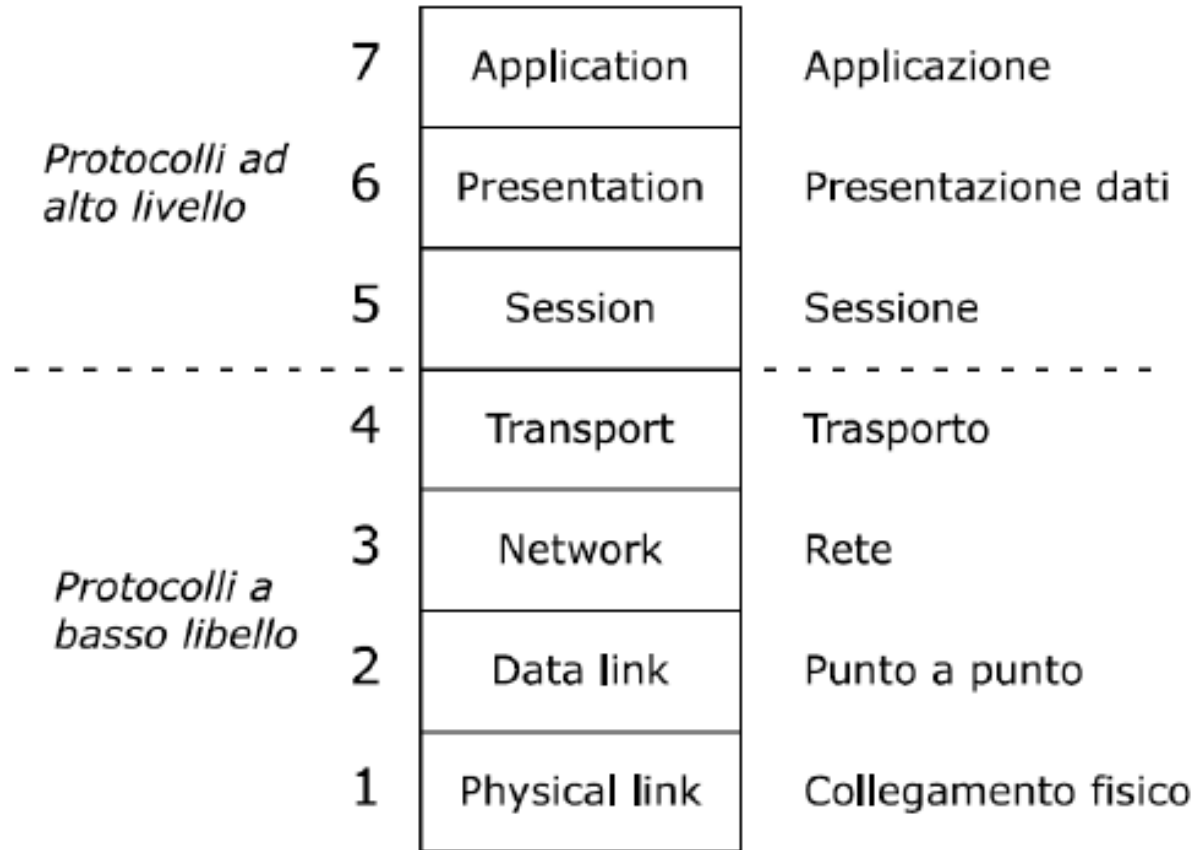
- Nelle specifiche di un protocollo identificabili:
 1. Lessico: formato dei messaggi da inviare e ricevere
 2. Sintassi: quali sono le sequenze di messaggi valide
 3. Semantica: significato delle sequenze di messaggi
 4. Sistema di turni: quando inviare e ricevere messaggi

Modello ISO-OSI

- Le reti standardizzate dalla *International Standard Organization* (ISO) usando il modello *Open System Interconnection* => ISO-OSI
- Modello != Architettura
 - Modello: sistema come insieme di funzionalità che deve assolvere. ISO-OSI definisce i livelli funzionali e le relazioni interne
 - Architettura: realizzazione pratica. TCP/IP ‘implementa’ ISO-OSI

Modello ISO-OSI

- Funzioni logiche in sette strati

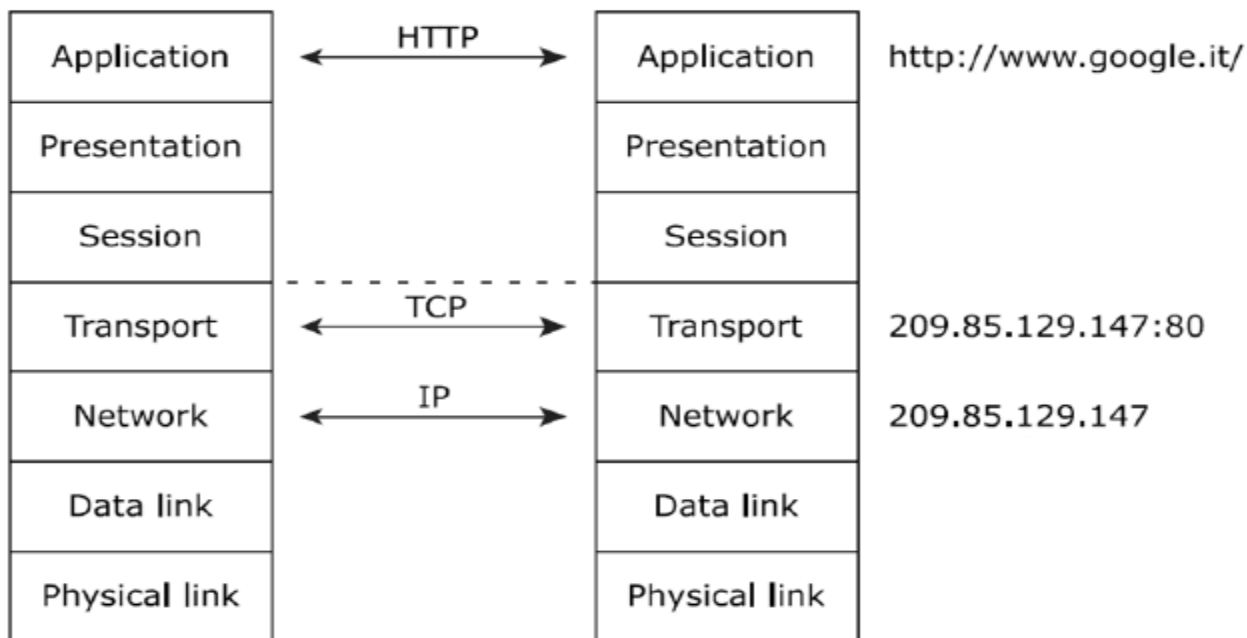


Modello ISO-OSI

- **Livello fisico:** trasmissione di sequenze di bit sul canale di comunicazione
 - Codifiche dei bit, cavi e connettori
- **Data link:** permettere il trasferimento affidabile di dati tra nodi adiacenti.
 - Invia frame di dati con sincronizzazione ed effettua un controllo degli errori
- **Rete:** *rende i livelli superiori indipendenti dai meccanismi e dalle tecnologie di trasmissione usate per la connessione. Scambio di dati tra entità non connesse fisicamente*
 - Routing, gestione della congestione, conversione dei dati tra reti con diverse caratteristiche
- **Trasporto:** trasferimento di dati trasparente e affidabile tra due host .
Livello end-to-end
- **Sessione:** gestisce sessioni logiche e sincronizzazione tra applicazioni
- **Presentazione:** trasformare i dati forniti dalle applicazioni in un formato standardizzato e offrire servizi di comunicazione comuni, come la crittografia, la compressione del testo
- **Applicazione:** interfacciare utente e macchina

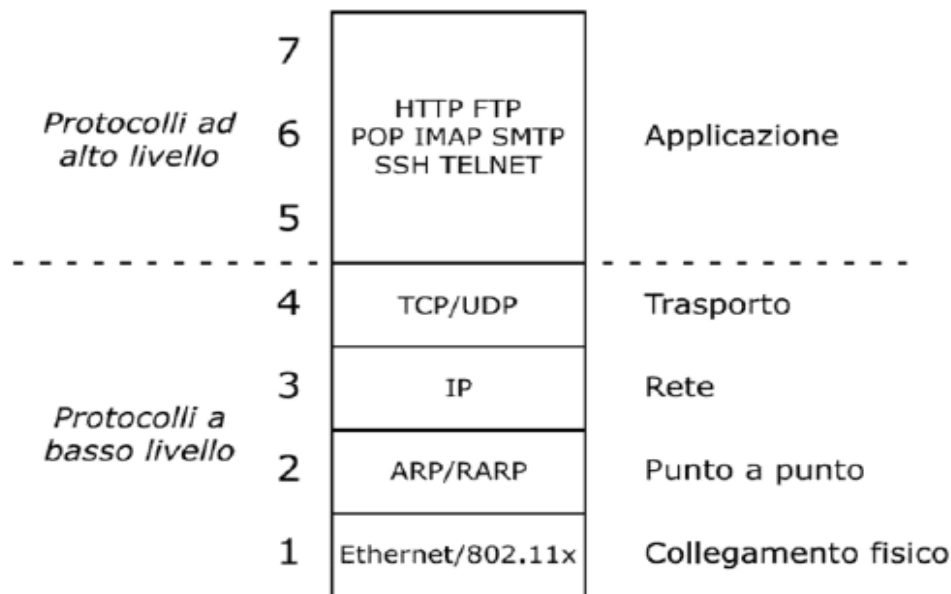
Modello ISO-OSI

- Ogni livello comunica con i due adiacenti (sopra e sotto)
- Gli stessi livelli su nodi di rete distinti ‘parlano’ tra loro
=> protocollo e indirizzamento per identificazione



Internet e i suoi protocolli

- ‘internet’: rete che impiega specifici protocolli per i livelli di rete e trasporto: Internet Protocol IP e Transmission Control Protocol TCP e User Datagram Protocol UDP
- TCP/IP NON è un protocollo



Livello di rete

- Dati spediti in maniera indipendente e raggiungono destinazione senza coordinazione seguendo percorsi probabilmente diversi
- Consegna non garantita o non ordinata
- IP è connectionless: no connessione permanente tra entità che dialogano
- Politica best effort: no meccanismi per ritrasmissione pacchetti persi
- Fornisce sistema di indirizzamento globale = indirizzi IP

Indirizzamento IP

- Indirizzi IP = numeri di 32 bit espressi in notazione decimale puntata
- Indirizzi gestiti ed assegnati ai vari soggetti da organizzazione internazionale: Internet Assigned Numbers Authority IANA
- Si differenziano in pubblici (assegnati da IANA) e privati (raggiunto solo dall'interno della stessa rete)

Indirizzo iniziale	Indirizzo finale
10.0.0.0	10.255.255.255
127.0.0.0	127.255.255.255
169.254.0.0	169.254.255.255
172.16.0.0	172.31.255.255
192.168.0.0	192.168.255.255

- Indirizzo di loopback: indica se stesso

Livello di trasporto

- Livello di trasporto estende i servizi di IP: TCP e UDP
- UDP: no miglioramenti rispetto ad IP ma raggiungibile da applicazioni
- TCP: simula sistema di trasporto orientato alla connessione, affidabile e consegna ordinata, controllo di flusso e di congestione
- Rendono possibile connessione virtuale tra due applicazioni. Estensione indirizzamento di IP portandolo a granularità più fine
- Indirizzo a 48 bit = 32 bit per IP e 16 per numero di porta (riferimento al canale di comunicazione dell'applicazione)

Livello di applicazione

- Protocolli tra applicazioni per la fruizione di servizi
 - Posta elettronica (IMAP, SMTP, POP)
 - Trasferimento file (FTP, SFTP)
 - Condivisione di file system (SMB, NFS)
 - Gestione sessioni di lavoro (telnet,SSH)
 - Trasporto (HTTP)
 - Ricerca di informazioni (DNS, whois)
- Specifica servizi standardizzati su sito di *Internet Engineering Task Force* in documento chiamati *Request for Comments* RFC

- Uniform Resource Identifier URI: sistema per l'identificazione di una singola risorsa all'interno della rete

`<scheme>:<scheme-specific-part>`

- URI per accedere a informazioni di maniera gerarchica

`<scheme>://<authority><path>?<query>`

- Scheme: protocollo di accesso ad informazione
- Authority: entità per gestione e fornitura
- Path: percorso nello schema gerarchico
- Query: informazione elaborata da una risorsa

`http://www.unimi.it/chiedove/schedaPersonaleXML.jsp?matricola=16088`

- Uniform Resource Locator: riferimento ad una risorsa in internet, metodo di accesso per raggiungere informazione
 - Categoria di URI
- Applicazioni Java usano classe URL nel package `java.net` per rappresentare indirizzi URL

Creazione URL

- Il modo più facile è creare un URL da una stringa => crea un riferimento assoluto
- Si possono creare riferimenti relativi (HTML) usando il costruttore `URL(URL baseURL, String relativeURL)`
- Per codificare i caratteri speciali nell'URL usa la classe `java.net.URI` che in modo automatico si occupa della codifica. Per passare da URI a oggetto URL uso il metodo `toURL()`
- `MalformedURLException`

Parsing URL

Posso ottenere tutte le componenti di un URL

- `getProtocol`
- `getAuthority`
- `getHost`
- `getPort`
- `getPath`
- `getQuery`
- `getFile`

Leggere e connettersi

- Metodo `openStream()` restituisce un `InputStream`
- `openConnection()` restituisce una `URLConnection` con cui posso settare alcuni parametri per la connessione
- La connessione inizia invocando il metodo `connect()` di `URLConnection`

- Rappresenta un indirizzo IP e permette al programmatore di usare nomi in formato alfanumerico
- Non è dotata di un costruttore, si crea un'istanza solo usando uno dei suoi metodi statici usati per la risoluzione dei nomi
- `getByName(String name)`, `getAllByName(String name)`, `getLocalHost()`;
- `getAddress()`

Esercizi

- Si costruisca l'URL del pdf che state leggendo, prima si usi l'URL assoluto, poi si faccia uso del costruttore che permette di specificare l'URL relativo (gestendo le eccezioni)
 - Confrontate se i due URL ottenuti sono gli stessi (cercare nella JavaDoc quale metodo fa per voi)
 - Cosa succede se confronto l'url
`http://download.oracle.com/javase/6/docs/api/java/net/URL.html#sameFile(java.net.URL)` con `http://download.oracle.com/javase/6/docs/api/java/net/URL.html`
 - Si esegua una ricerca in Google, si parsi l'URL della ricerca e si stampino a console le varie parti (<http://www.google.it/search?gcx=c&sourceid=chrome&ie=UTF-8&q=www.unimi>). Utilizzando le varie parti si costruisca un oggetto URL
 - Se vi chiedessi di stampare il codice HTML di www.google.it ? usando solo quello che sapete
 - Esageriamo: scaricate il pdf di una lezione, chiamando il file 'qualcosa.pdf'.
 - Prendiamo un po' di nomi di host a caso e confrontiamo gli ip dati con quelli del comando ping
 - Qual è il vostro IP
- (sia b[] un array di byte, provate a vedere cosa succede se faccio `println((b[0] & 0xff))`)

- `System.setProperty("http.proxyHost",
«159.149.107.42»);
System.setPropery("http.proxyPort", "8080");`