UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

CDS IN INFORMATICA E
COMUNICAZIONE DIGITALE

Anno Accademico 2015-2016

Corso di

"Reti di Calcolatori e Comunicazione Digitale"

Modulo 6 TCP/IP : i protocolli a livello applicativo

Prof. Sebastiano Pizzutilo
Dipartimento di Informatica

Il livello di Applicazione

- **Il livello di Applicazione fornisce servizi alle applicazioni degli utenti della rete.**

- La comunicazione è realizzata per mezzo di una **connessione logica tra due identità logiche** mittente e ricevente

- Un programma che vuole comunicare con un altro programma deve gestire i primi quattro livelli dello stack **TCP/IP (aprire la connessione, inviare/ricevere dati e chiudere la connessione)**. *Un insieme di istruzioni di questo tipo viene chiamato **API (Application Programming Interface)** o **socket** .*

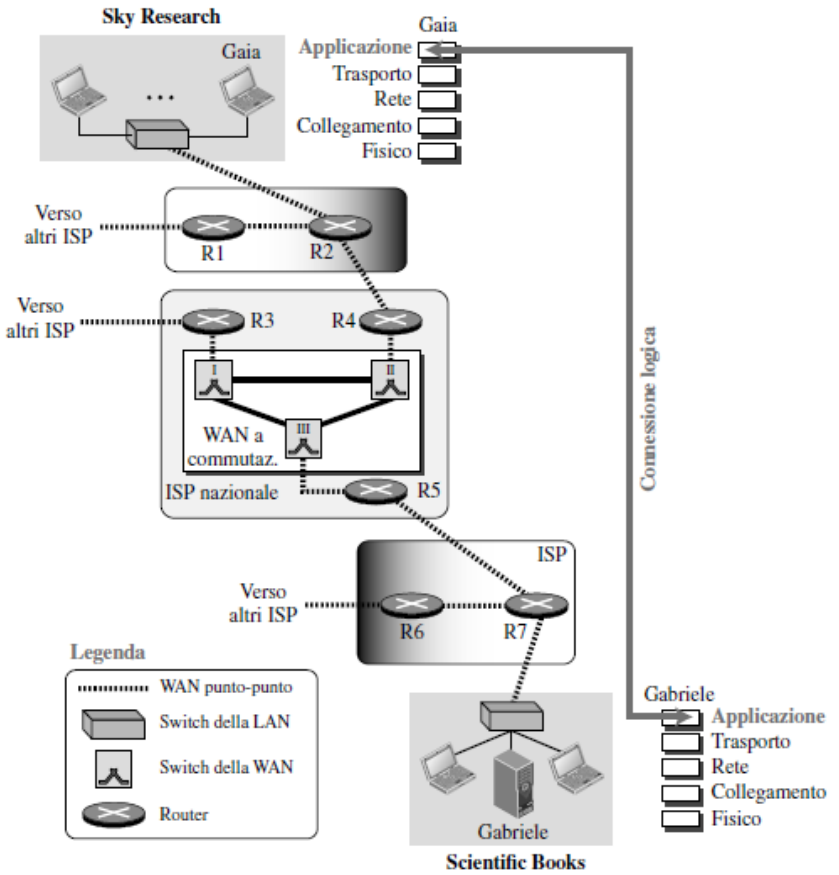
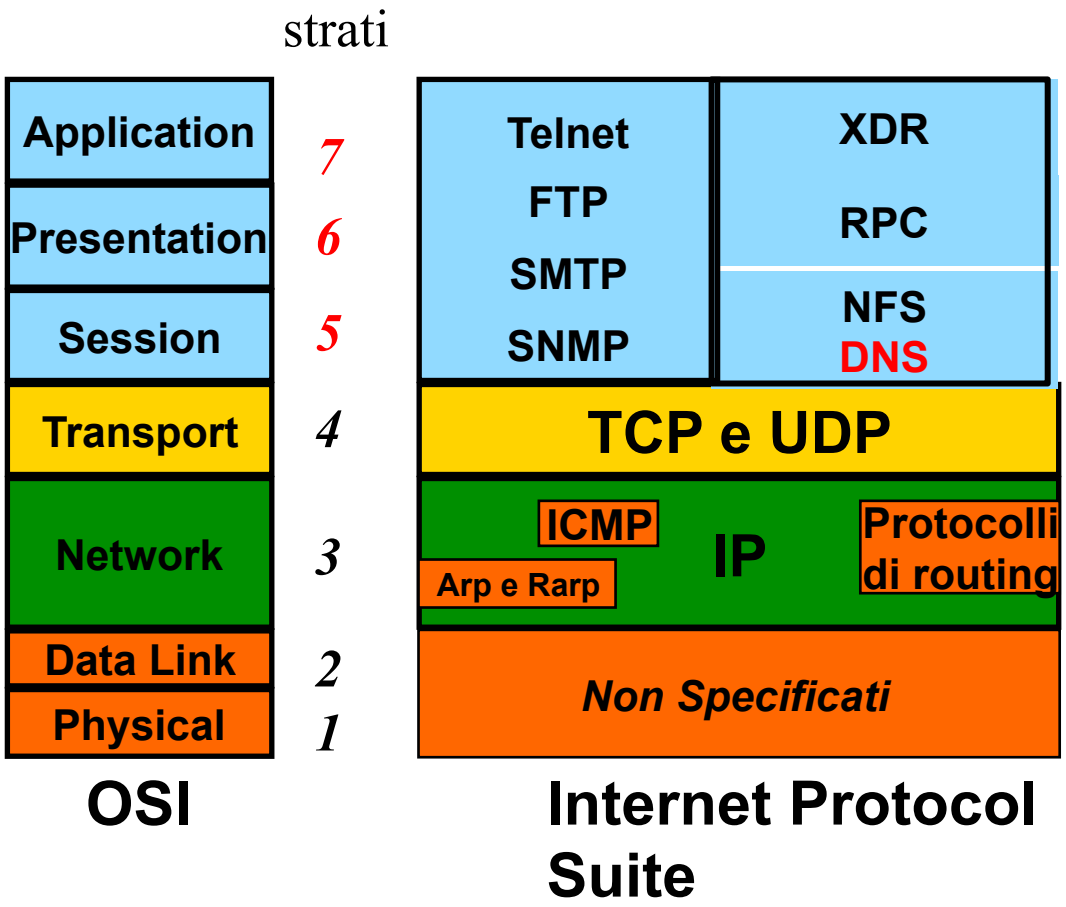


figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

TCP/IP : Livello (5-6-7) di Applicazione

ISO-OSI e TCP/IP



Lo strato dell'applicazione TCP/IP, comprende gli strati di ***applicazione***, ***presentazione*** e ***sessione*** del modello OSI.

Questo strato ***utilizza sistemi di naming legati alle funzioni svolte dalle diverse applicazioni*** e ***basati nomi mnemonici*** *per identificare* ***l'applicazione*** **(o l'utente)** *insieme* ***al "dominio"***, (ovvero alla rete degli utenti che comunicano).

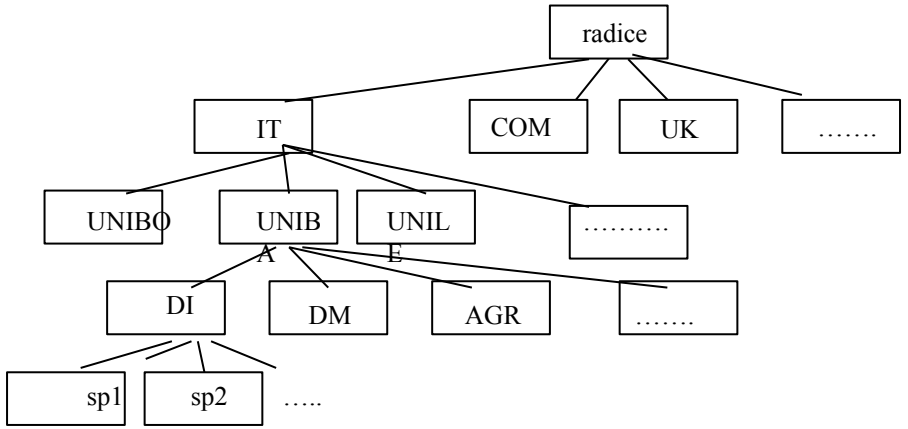
L'applicativo che ***associa gli indirizzi IP ai nomi mnemonici*** delle reti è il **DNS**.

Nomi e indirizzi – Domain Name System

Il DNS (“sistema dei nomi di dominio”) è una applicazione *client-server* che viene utilizzata dalle applicazioni per tradurre i nomi logici mnemonici utilizzati dalle applicazioni (*e-mail, web,…*) in indirizzi IP e viceversa, attraverso un *processo di risoluzione*.

Organizzazione del DNS – I Domìni

- Internet è partizionata in **aree logiche dette “domini”.** I singoli **domìni** possono a loro volta essere suddivisi in **sottodomìni** (non esiste limite al numero di ripartizioni di un domìnio o sottodomìnio).
- La struttura dei nomi segue questa ***organizzazione gerarchica*** a partire da destra, dove si trova la stringa di maggior valore (domìnio primario)



I nomi di dominio sono definiti in una struttura ad albero invertito con la radice (root) in cima. L’abero può avere fino a 128 livelli (da 0 a 127).

Domìni primari

- **edu** istituzioni scolastiche o di ricerca USA
- **gov** istituzioni governative USA
- **com** organizzazioni commerciali
- **mil** gruppi militari USA
- **org** altre organizzazioni
- **net** centri di supporto alla rete
- **country code** sigle standard per identificare le nazioni (ISO 3166)
  it, fr, uk, de, au, jp, ie, dk, br, …

Il database con tutti i nomi degli host di Internet viene gestito in modo che:

– Lo **spazio dei nomi è suddiviso in zone non sovrapposte** contenenti uno o più sottodomìni.

– Ciascuna **zona** prevede un **DNS server** (*sulla porta 53*) principale ed uno o più server secondari.

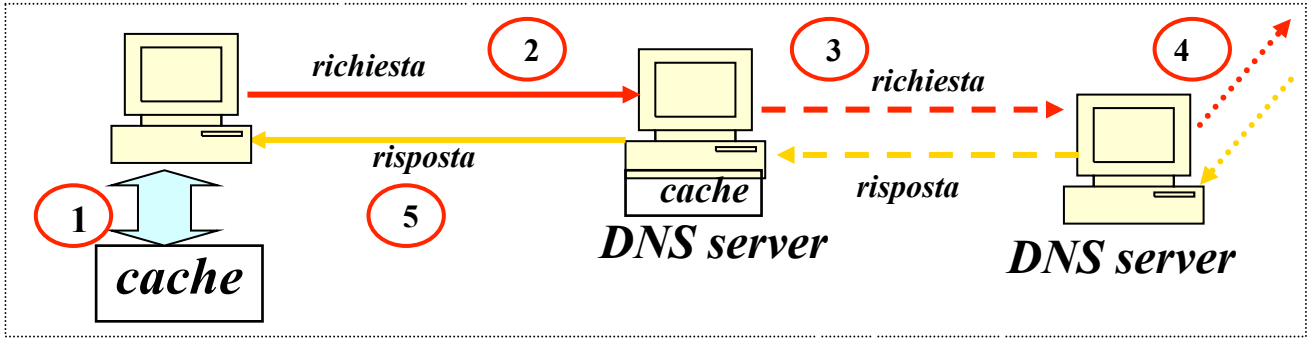
– Ogni **name server** gestisce un data base degli indirizzi IP corrispondenti ai nomi degli host contenuti nella zona di cui è responsabile.

Il DNS resolver

Il DNS resolver realizza il mapping tra nomi logici di host (e dominio) e indirizzi IP mediante un processo di risoluzione (iterativa o ricorsiva) .

*Realizza le funzioni di conversione da…**hostname** …a…**IP** e da…**IP**…a…**hostname***

Processo iterativo di resolving



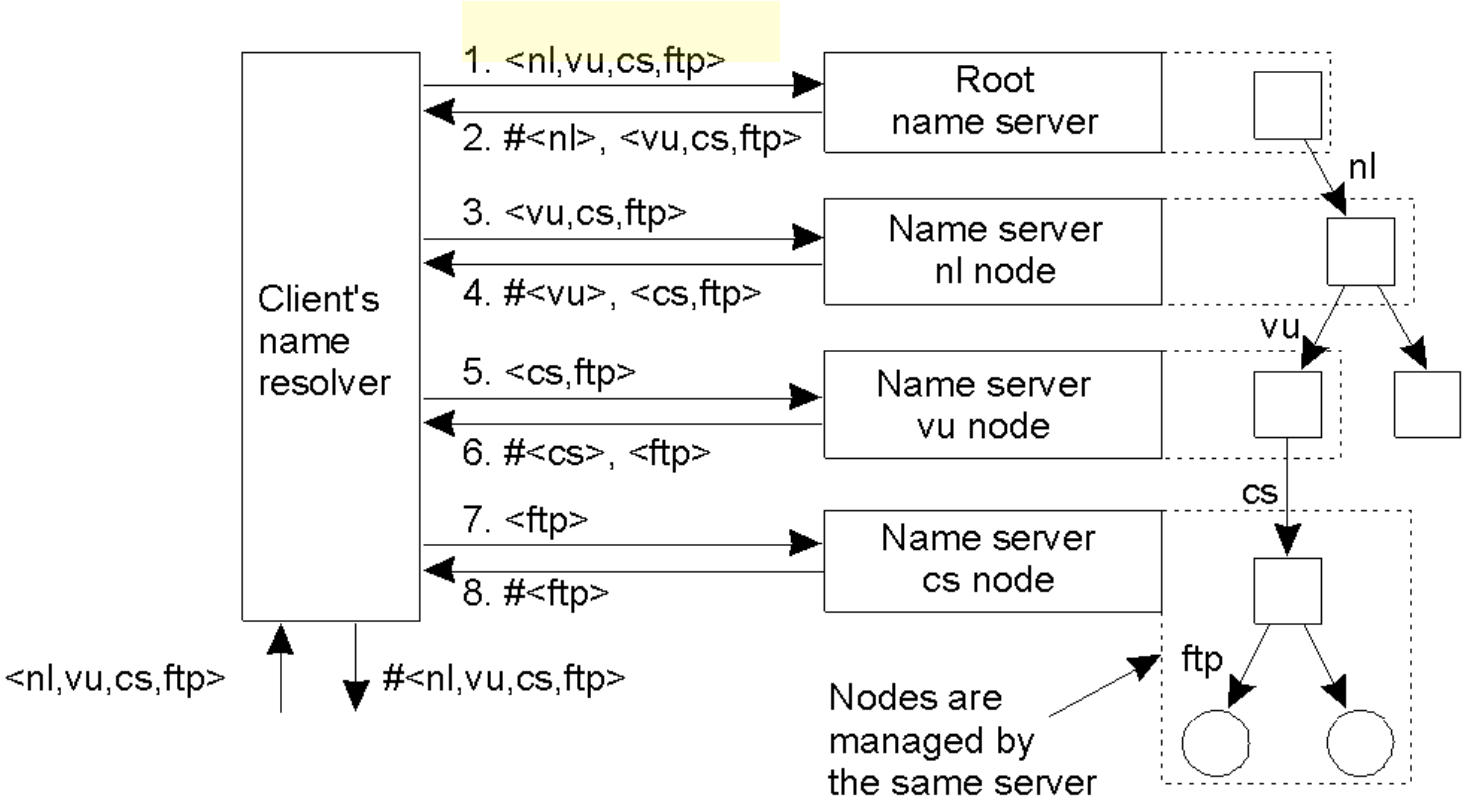
1. **Il dato** richiesto *(il nome del server)* viene cercato nella cache.
2. **Se il dato** non è nella cache, si cerca un DNS server cui sottoporre la richiesta, partendo da un DNS server locale.
3. **Il DNS server locale** dispone di una lista di ***DNS server esterni*** cui chiedere nel caso non sia in grado di rispondere (entro un time out stabilito).
4. **Se la richiesta fallisce** anche sui ***DNS server esterni***, la richiesta viene girata al ***server di top level*** domain.
5. **Se viene ricevuto un messaggio** positivo o la segnalazione di un errore nel nome, i dati vengono inseriti nella cache del DNS server locale prima di essere spediti al client.

Implementazione della Name Resolution Iterativa

E' possibile utilizzare **due** diverse modalità di **risoluzione**: **Iterativa** e **Ricorsiva**

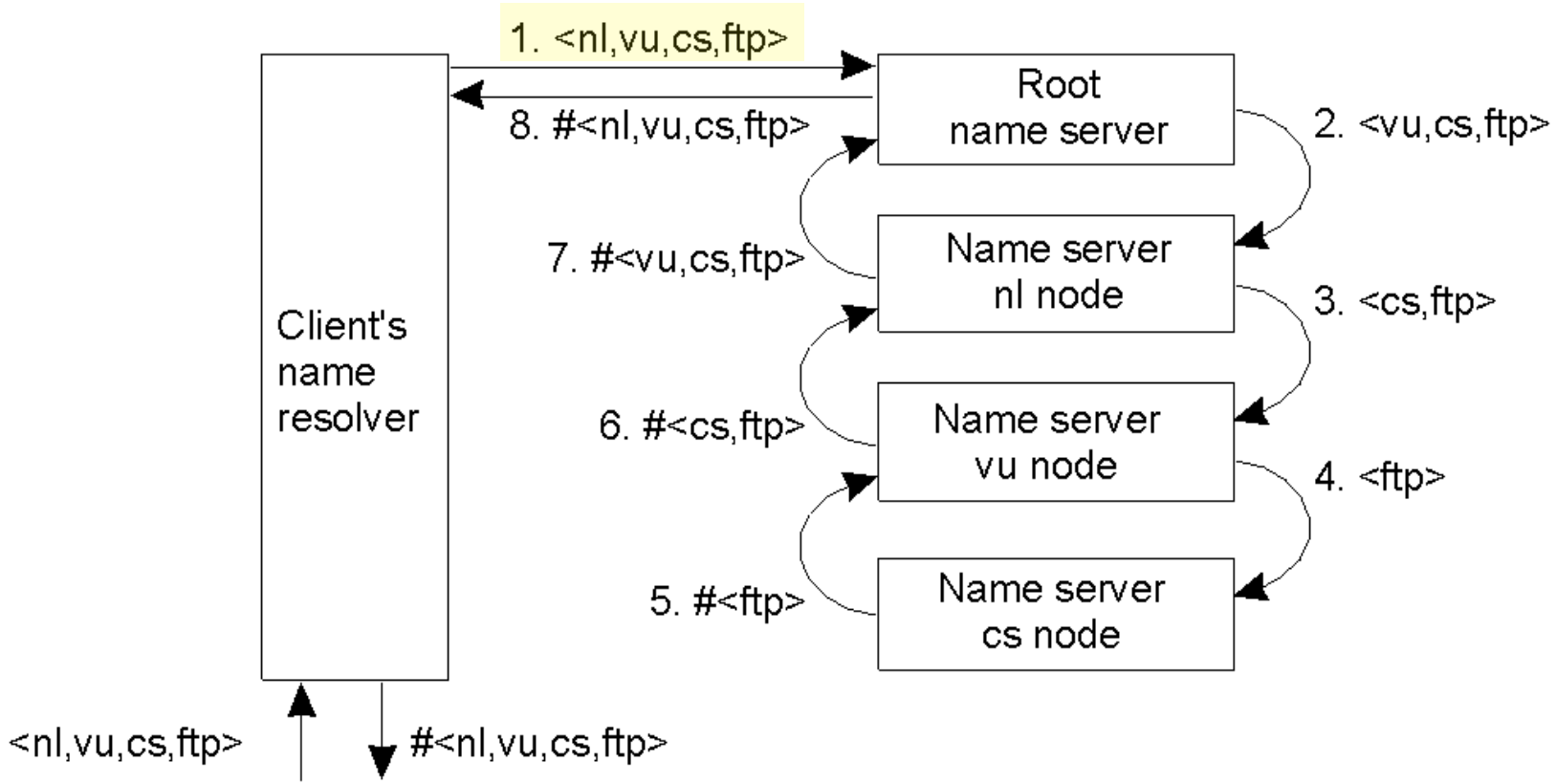
In genere si effettua una risoluzione ricorsiva tra resolver e DNS locale, iterativa tra DNS server.

Iterativa: ***il name server dell'host client provvede a interrogare iterativamente i name server di ogni livello di dominio per ottenere l'informazione.***



Implementazione della Name Resolution Ricorsiva

***Ricorsiva**: il name server del dominio di più alto livello reperisce l'informazione interrogando ricorsivamente i name server dei livelli inferiori e poi la trasmette al name server dell'host client.*



Messaggi DNS

I messaggi **DNS** sono solo di due tipi: **QUERY** e **RESPONSE.**

- **DNS** usa **UDP** a livello di trasporto quando il messaggio di risposta è < 512 byte, e usa **TCP** quando il messaggio di risposta è >512 byte.
- ***"Named"*** è il *demone* (processo di background) che risolve i nomi di dominio e usa la ***porta 53*** per la risoluzione.
- ***"Nslookup"*** è il comando (Unix e Windows) che permette di interrogare il DNS.

Esempio di interrogazione del DNS:

- nslookup 192.168.1.2
  - *restituisce il nome corrispondente all'IP indicato. Utilizza la risoluzione inversa*
- nslookup roggen.brot.dg dinkel.brot.dg
  - *interpella il servizio di risoluzione dei nomi offerto dall'elaboratore dinkel.brot.dg per ottenere le informazioni sul nodo roggen.brot.dg*

TCP/IP : Livello (5-6-7) di Applicazione

Telnet

***Telnet* è un protocollo applicativo *client-server* basato su TCP.**

Il protocollo *Telnet* permette di aprire una sessione di comunicazione in chiaro (*non criptata*) bidirezionale tra due host.
Una volta realizzata la connessione, il client può lavorare sulla macchina remota come se fosse direttamente collegata al proprio computer.

Il client *(che lancia il comando "telnet")* utilizza di solito la *porta 23* per la connessione al server *(sul quale deve girare il demone "telnetd" in ascolto sulla porta 23)*.

La sintassi di un comando *telnet* sul client è la seguente :
telnet [<opzioni>] [<host-remoto> [<porta>]]

La sessione aperta dal ***telnet*** consente di utilizzare **solo comandi richiamabili da linea di comando**, o che comunque **non necessitano di interfaccia grafica**.

Come funziona Telnet

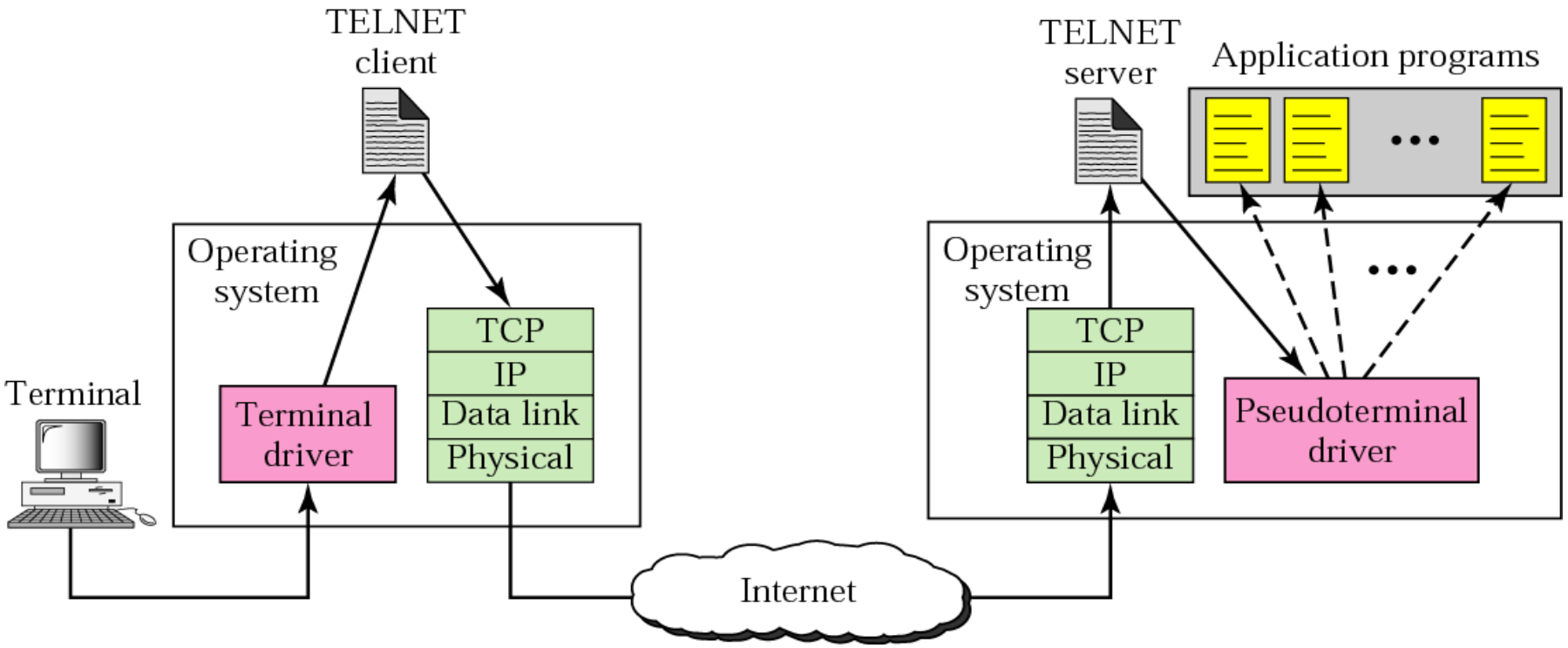


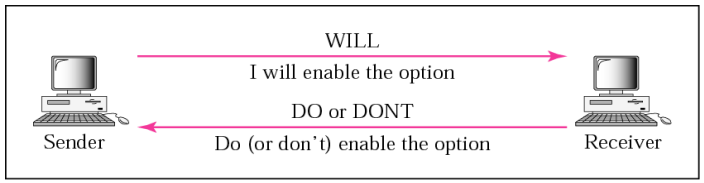
figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Funzioni di Telnet

- **Network Virtual Terminal**

Ambiente virtuale (una interfaccia) creato ai due estremi della connessione (il client ed il server Telnet). Le ***funzioni disponibili per la connessione sono negoziate*** **tra i due host partendo da un insieme minimo**.

- **Negoziazione delle opzioni**



E' implementato un meccanismo di **negoziazione delle opzioni** grazie al quale una sessione può avvenire utilizzando le caratteristiche comuni ai due host. Le opzioni di base disponibili possono poi essere successivamente ampliate. Ad es.: bin, line, echo, status, terminal type,….

- **Viste simmetriche di terminali e processi**

La negoziazione delle opzioni, essendo simmetrica, può dare origine ad un ciclo infinto di richieste.
Telnet prevede delle funzioni per evitare questo tipo di problema*: ad es. la richiesta di definizione di un'opzione può essere inviata ad un NVT solo se viene richiesto un cambiamento dello stato delle opzioni; se un NVT riceve una richiesta di definizione di un'opzione già definita non deve inviare nessun messaggio di accettazione al fine di evitare la possibile nascita di un loop senza fine; ....*

SSH, Secure SHell

***Telnet ha il grosso limite di prevedere la trasmissione tra client e server solo in chiaro**. E' quindi stato sostituito da **SSH** che permette di stabilire una **sessione remota cifrata** tramite interfaccia a riga di comando con un altro host in rete.*

Sintassi: ***ssh [opzioni] nomeutente@host [comando]***

SSH realizza un ***canale criptato*** nel quale la comunicazione avviene in maniera **cifrata:** **SSH** realizza sia la **mutua autenticazione (**del client e del server) che la **protezione** della comunicazione durante l'intera **sessione di lavoro** .

SSH realizza anche la funzionalità di ***port forwarding***, che consente di aprire una socket TCP sul client SSH o sul server che re-dirige le connessioni ricevute su una porta verso un host con una porta specificata (nell'SSH) .

Il port forwarding è utile anche per trasportare applicazioni X Window (interfaccia grafica) attraverso una connessione SSH (X forwarding)

Numerosi tool permettono di accedere in sicurezza ad un device remoto in modalità grafica : ad es. TeamViewer

TCP/IP : Livello (5-6-7) di Applicazione

File Transfert Protocol

Il *File Transfer Protocol* è un protocollo applicativo di tipo *client-server* che consente di trasferire file di dati tra host remoti in una rete TCP/IP. Quindi consente ad un client l'accesso al file system di una macchina remota (server) per scrivere (put) o leggere (get) files.

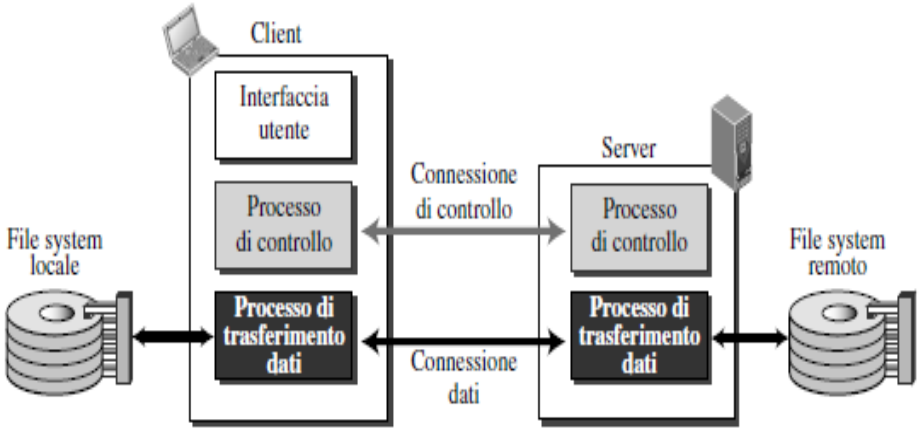
Il protocollo **FTP** utilizza due processi distinti per il trasferimento dei dati:

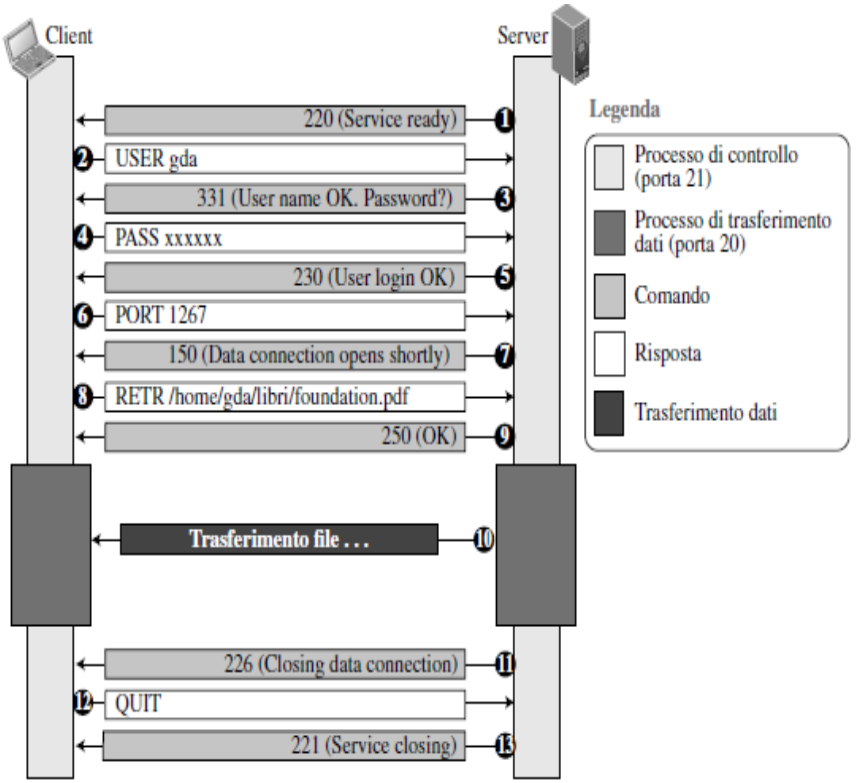
- il **Protocol Interpreter** (***PI***) che è usato per trasmettere i comandi tra client e server (per creare la connessione sulla ***porta 21*** trasmettere login e password,
- il **Data Transfer Process** (***DTP***) utilizzato per la vera e propria trasmissione dei dati (sulla ***porta 20***).

Una sessione FTP è in effetti costituita da **due sessioni separate**:

1. ***la prima sessione si effettua tra i servizi PI del client e del server*** : serve a stabilire i meccanismi della connessione (*quali il nome dell'utente, la verifica della password*). In questa fase viene anche concordata le modalità per effettuare la connessione dati tra il DTP del server e quello del client.
2. ***nella seconda avviene l'effettivo scambio dei dati tra i due processi DTP.***
   Il trasferimento dei dati può avvenire utilizzando diversi formati: ASCII, EBCDIC e in BINARIO.

File Transfert Protocol





Per potersi collegare ad un client FTP bisogna essere autorizzati (possedere una ***username*** ed una ***password*** per poter accedere al servizio FTP fornito dal server).

Questo tipo di connessione presenta l'inconveniente di far viaggiare sulla rete la password di accesso al server in chiaro.

figure da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

FTP e FTP anonimo

Un modo per evitare *che un eventuale* ***sniffer*** *possa accedere al sistema con tutti i privilegi di accesso (username e password) di un utente che abbia già usato in precedenza l'FTP,*
è quello di utilizzare, **laddove il server FTP lo metta a disposizione**, l'**FTP anonimo**.

Con **l'FTP anonimo** si utilizza uno ***username*** standard (***anonymous ftp)*** e la password da fornire è il proprio ***indirizzo di posta elettronica***.
Configurare un server per consentire l'**ftp anonimo** vuol dire definire un'area di file system per consentire a chiunque di accedervi per effettuare operazioni l'upload/download di files.

Questo metodo di accesso evita di far viaggiare in rete delle password in chiaro, ed ha l'ulteriore vantaggio che, nel caso di macchine UNIX, il server FTP prima di consentire l'accesso effettua una* chroot*, impedendo di fatto all'utente collegato di vedere la reale strutturazione del file system della macchina.

TCP/IP : Livello 5 (-6-7) di Applicazione

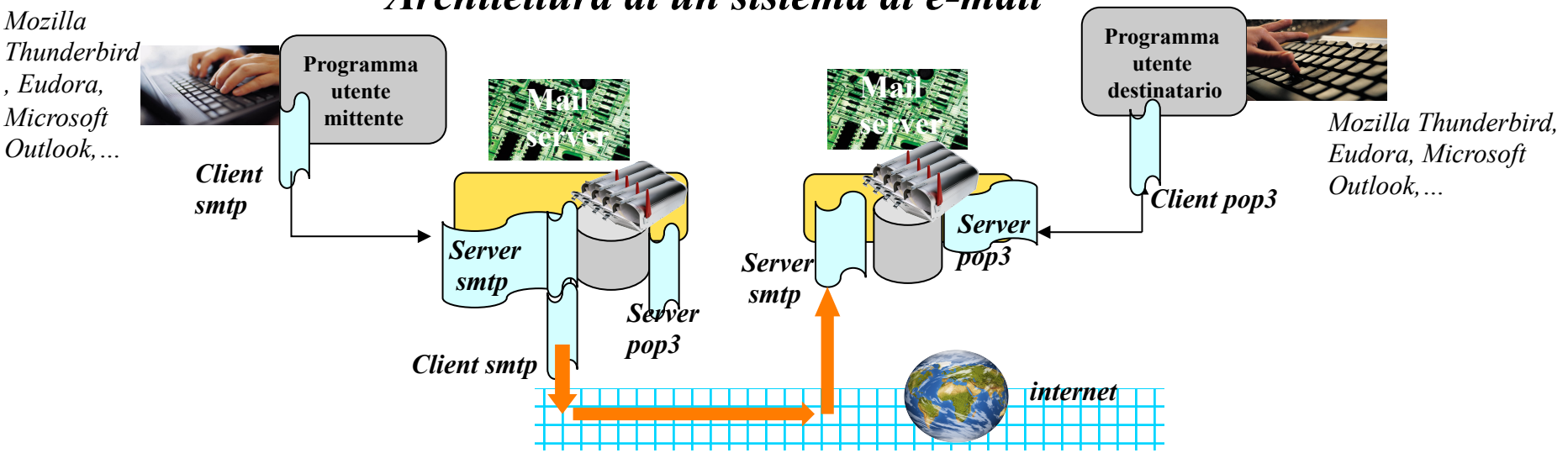
Posta elettronica (Electronic Mail — E-Mail)

La posta elettronica o e-mail è un servizio Internet grazie al quale ogni utente può inviare o ricevere dei messaggi.

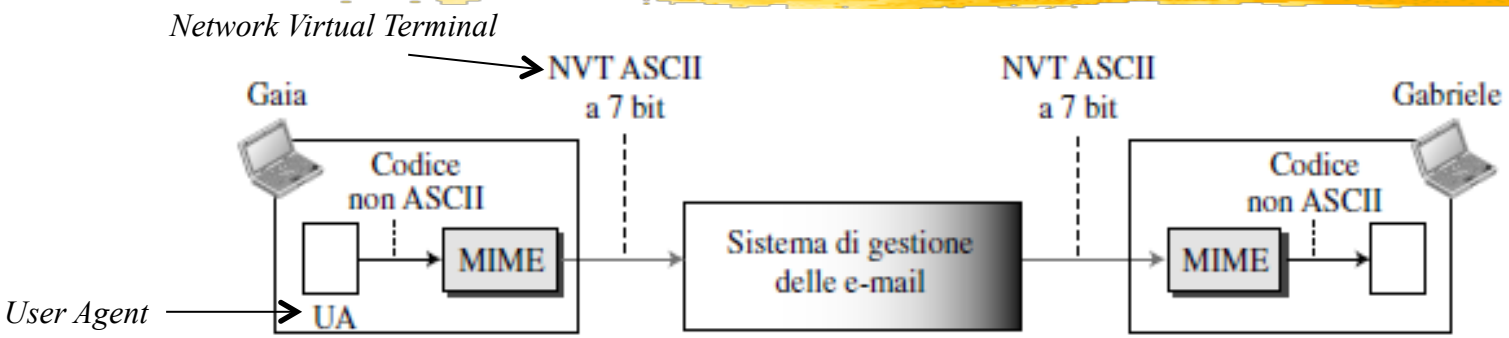
La modalità di accesso al servizio è quindi asincrona, unidirezionale e la consegna al destinatario non è garantita.

Indirizzo di e-mail = identificativo mail box utente @ dominio del server di posta

Architettura di un sistema di e-mail



Formato di un messaggio di e-mail



Multipurpose Internet Mail Extension

Tipo	*Sottotipo*	*Descrizione*
Text	Plain	Testo non formattato
	HTML	Formato HTML
Multipart	Mixed	Il corpo è composto da una lista ordinata di parti con formati diversi
	Parallel	Come il precedente, ma le diverse parti non sono ordinate
	Alternative	Le parti sono versioni differenti dello stesso messaggio
Message	RFC822	Il corpo è un messaggio incapsulato
	Partial	Il corpo è un frammento di un messaggio più grande
	External-body	Il corpo è un riferimento a un altro messaggio
Image	JPEG	Immagine in formato JPEG
	GIF	Immagine in formato GIF
Video	MPEG	Video in formato MPEG
Audio	Basic	Codifica audio mono a 8 Khz
Application	PostScript	Adobe PostScript
	Octet-stream	Codifica dati binari (con byte di 8 bit)

figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Posta elettronica (Electronic Mail — E-Mail)

SMTP

L'*SMTP* (*Simple Mail Transfer Protocol* – RFC 821) è il protocollo utilizzato solo per trasmettere messaggi di posta elettronica associato al protocollo TCP per il trasporto.

Un **server SMTP** è un programma sempre attivo in ascolto sulla ***porta 25***.

Per associare il server SMTP a un dato nome di dominio (DNS) si usa un ***Resource Record di tipo MX (Mail eXchange)***.

Poiché SMTP è un protocollo **testuale** basato sulla codifica **ASCII**, non è permesso trasmettere direttamente testo composto con un diverso set di caratteri. Lo standard **MIME (*Multipurpose Internet Mail Extensions* – RFC 2045)** permette di estendere il formato dei messaggi mantenendo la compatibilità col software esistente.

Posta elettronica (Electronic Mail — E-Mail)

POP3 – IMAP

Il **POP3** **(Post Office Protocol version 3 –** RFC 1939**) è il protocollo più comunemente usato per leggere, prelevare o cancellare i messaggi di posta elettronica.**

In una sessione POP3 si seguono i seguenti passi:

- Il client si connette alla ***porta 110*** del server.
- Il server invia un messaggio di saluto.
- Si inizia la sessione vera che consiste di una fase di AUTHORIZATION e di una successiva di TRANSACTION.
- Allo stato di TRANSACTION si passa solo dopo aver superato con successo lo stato di AUTHORIZATION, fornendo la propria identificazione.

IMAP ***(*****Internet Message Access Protocol** RFC 3501**) è l'evoluzione del POP3**

Che prevede procedure di **sincronizzazione** più complesse e complete rispetto a POP.

La porta predefinita di IMAP è la **143**.

Se si utilizza una connessione sicura tramite SSL, allora la porta è la 993.

Il protocollo Secure Socket Layer (SSL-TLS) fornisce un canale crittografato del tipo end-to-end tra il client ed il server. Prima che venisse definito questo protocollo le transazioni avvenivano in chiaro e potevano essere intercettate.

TCP/IP : Livello (5-6-7) di Applicazione

Il World Wide Web

Il ***World Wide Web*** ha iniziato ad avere diffusione all'inizio degli anni 90 sulla spinta del protocollo **HTTP**. Attualmente è noto come ***WWW, W3*** o semplicemente ***Web***.

Il WWW non è altro che una vasta rete di server HTTP in grado di comunicare tra di loro grazie alla rete Internet.

Il *Web non è Internet*: è solo uno dei servizi che è possibile trovare su Internet.

- Il **World Wide Web** nasce nel 1989 al CERN di Ginevra ***(Tim Berners-Lee)*** per scambiare informazioni tra gruppi di ricerca di fisica. ***("…WWW è l'universo delle informazioni globali accessibili tramite rete").***

- WWW richiede di utilizzare a livello applicativo :
  - ***sul client*** un **browser Web** ***(Gopher, Mosaic, Netscape Navigator, Internet Explorer, Lynx, HotJava,…)*** e
  - ***sul server*** un insieme di **risorse WEB** ***(in HTML, XML, PDF, Jpeg, MP3, codice Java, Perl, C ,…)*** ed un servizio di **server http.**

Architettura del WWW

L'architettura WWW è basata sul modello client-server in cui un qualsiasi host client (con un browser installato) può connettersi con un qualsiasi server WEB (opportunamente configurato), per scaricare sul proprio browser pagine ipertestuali/ipermediali in formato *html*.
La comunicazione è basata sulla suite di protocolli **TCP-IP** .

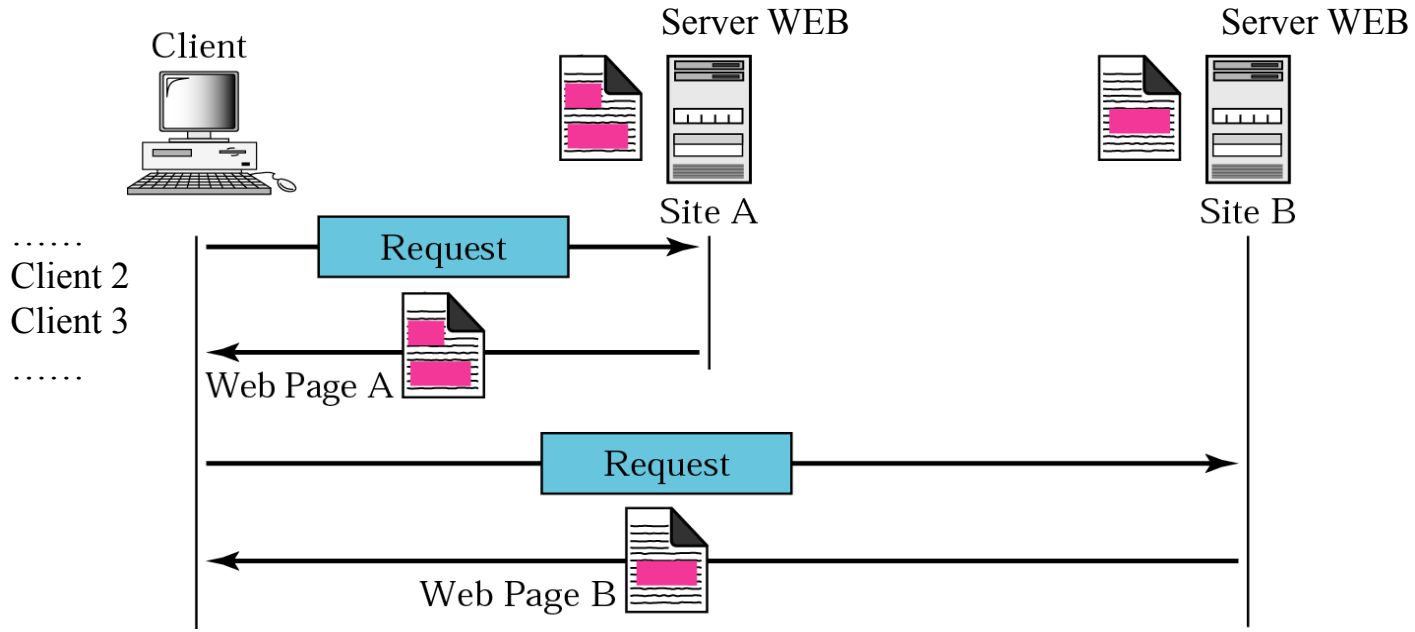


figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Client WEB

Il client WEB è una applicazione software che svolge il ruolo di interfaccia tra l'utente ed il WWW.

- ✓ La funzione principale è quella di **BROWSER:** ***localizzare il server*** ed inviargli opportuni messaggi per ***ottenere le risorse richieste*** (pagine html) ed ***interpretare*** (elaborare) ***il codice ricevuto*** per visualizzare in modo opportuno le informazioni.
- ✓ Il **client browser** dovrà quindi possedere capacità di ***resolving*** (fase di lookup) dei nomi, di ***caching*** delle risorse ricevute, di ***gestione dei cookie*** e di ***interpretazione*** dei dati (o dei programmi) ricevuti .

1. ***Il Browser acquisisce dall'utente l'URL da richiedere,***
2. ***Il Browser invoca il resolver per conoscere l'IP dell'URL cercato,***
3. ***Il Browser attiva una connessione TCP sulla porta 80 del server,***
4. ***Il Browser chiede al server col protocollo HTTP la risorsa ipermediale specificata,***
5. ***Il SERVER invia la risorsa richiesta,***
6. ***Se vi sono oggetti allegati, il Browser effettua una richiesta per ognuno di essi,***
7. ***Una volta inviati tutti gli oggetti collegati, il SERVER chiude la connessione,***
8. ***Il Browser attiva tutti i programmi necessari alla interpretazione degli oggetti ricevuti.***

Server WEB

Un sistema che ospita un sito WEB si compone di due elementi :

- una ***risorsa informativa*** ed
- un ***processo server*** WEB

risorsa informativa

1. **Pagine html (statiche o volatili)**
2. **Testo ASCII (statiche o volatili)**
3. **Testo PDF o PostScript (statiche o volatili)**
4. **Immagine GIF o Jpeg (statiche o volatili)**
5. Suono AIFF o MP3 (statiche o volatili)
6. **Video quicktime o MPEG (statiche o volatili)**
7. Rappresentazione VRML di scene tridimensionali, (statiche o volatili)
8. **Codice eseguibile in linguaggi tipo PERL e shell script CGI (dinamiche)**
9. Codice eseguibile in linguaggi compilati, tipo C (dinamiche)
10. **Codice Java (Java script) (attive)**
11. **Applet (attive)**

- ***Pagine statiche*** = pagine il cui contenuto è abbastanza stabile nel tempo,
- ***Pagine volatili*** = il cui contenuto viene modificato da eventi in corso,
- ***Pagine dinamiche*** = il cui contenuto è creato dinamicamente sulla base della richiesta del client.
  Il formato della pagina richiesta è di tipo ***script*** (***CGI***). La risposta può essere in diversi formati.
- ***Pagine attive*** = in cui il server WEB invia un programma eseguibile al client browser che lo esegue,

Web con pagine statiche

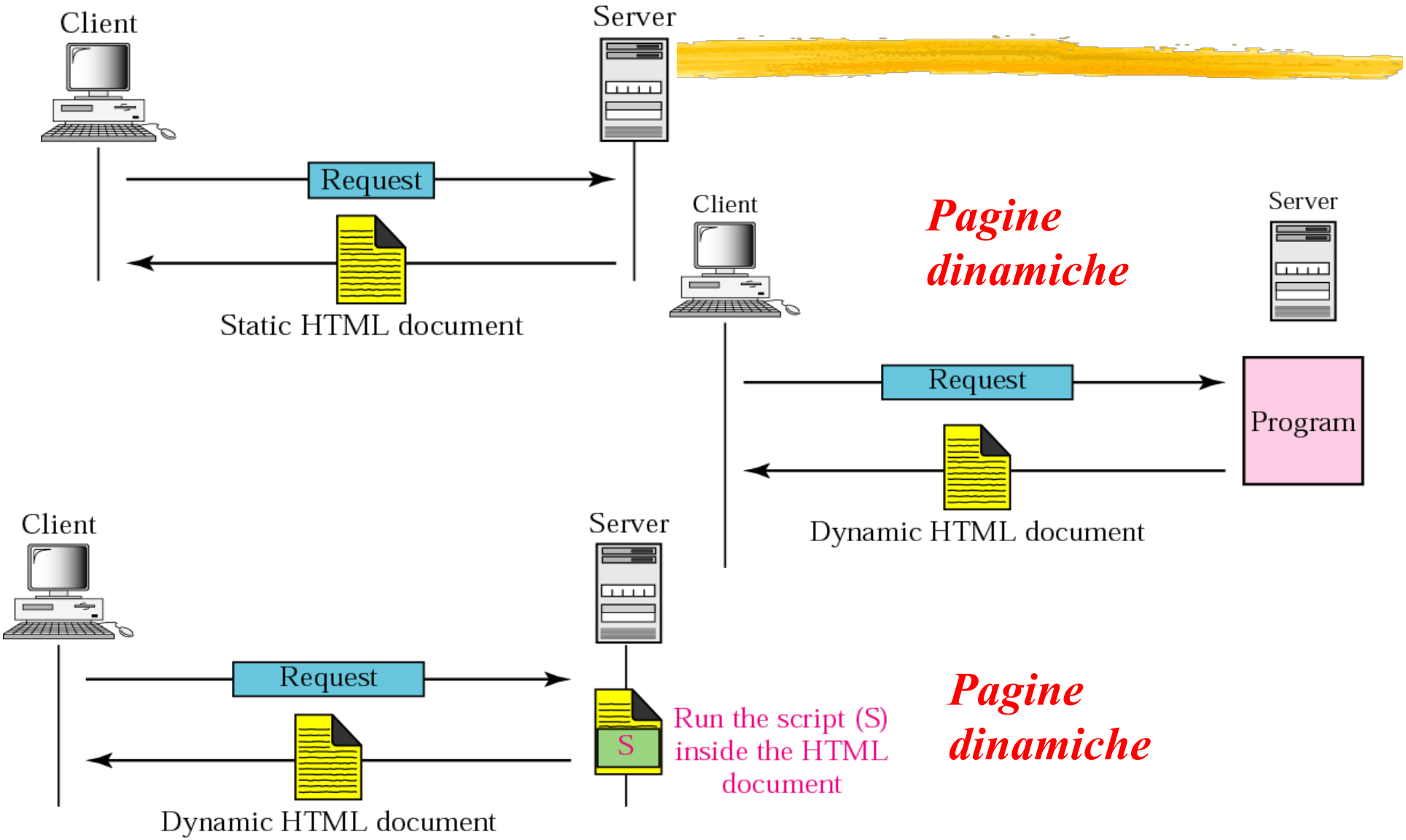


figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Server WEB

Il processo *httpd (http daemon*) è il processo sempre attivo sulla macchina server che si mette in ascolto sulla porta 80 in attesa di richieste http.

Processo server WEB

1. *Il processo server, una volta attivato, si pone in attesa di una richiesta proveniente da qualsiasi client WEB sulla porta 80,*
2. *Il server legge la richiesta del client (il metodo GET con tutte le azioni per localizzare, leggere e trasmettere il file -la risorsa-, la modalità di chiusura della connessione, il cammino della pagina ed il protocollo utilizzato – HTTP1.0 o 1.1)*
3. *Il server trasmette al client (utilizzando la stessa connessione) la risorsa (con tutti gli oggetti allegati).*
4. *Il server chiude la connessione.*

Le pagine create dinamicamente

Al fine di aumentare la tipologia e la qualità dei servizi offerti, il server WEB è stato arricchito di ulteriori funzionalità per consentire la ricerca di dati in data base, la personalizzazione del risultato dei servizi, ecc…

I meccanismi utilizzati per realizzare tali funzionalità sono PROGRAMMI (gateway script = script che svolgono il ruolo di interfaccia a servizi on line) che consentono al server WEB di connettersi ad altri servizi (accesso a data base, gestione di form interattive, ecc…)

Server WEB : gestione di pagine dinamiche

Uno **script** deve svolgere tre importanti funzioni:

a) Tradurre l'input del client nelle forme comprensibili ai servizi cui si collega,

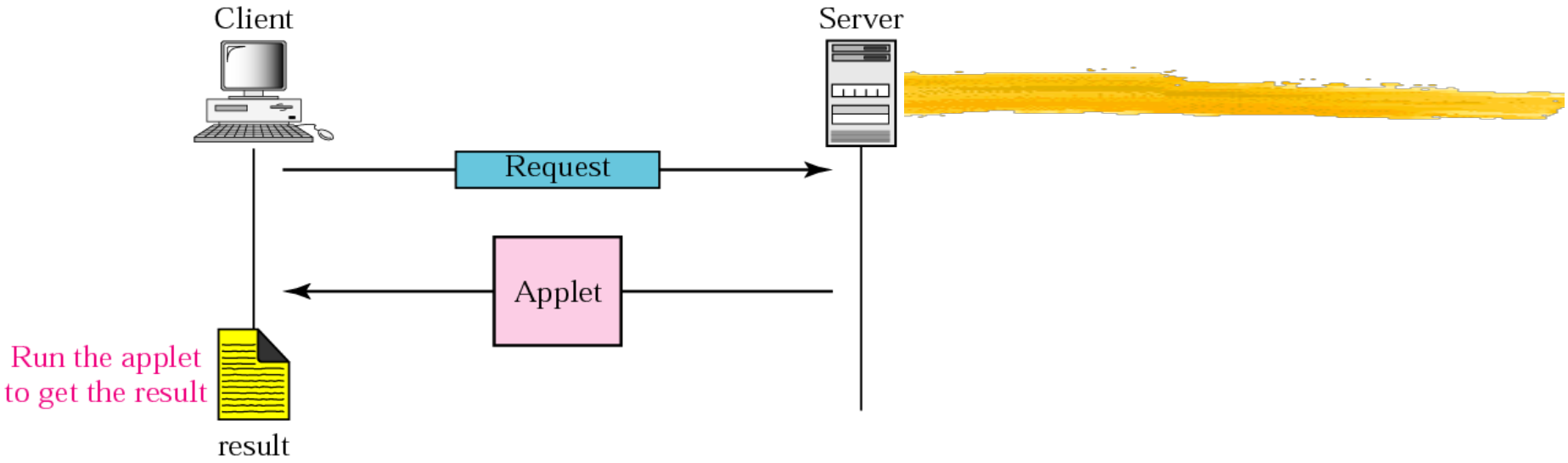
b) Invocare l'attivazione di ulteriori programmi,

c) Tradurre l'output del programma in una forma comprensibile al client (in formato Html).

*Gli **script** possono acquisire le informazioni dall'utente con **fill-in form** che vengono inviate dal server al client, il browser lo visualizza sul client host in modo che l'utente possa inserire i dati richiesti e lo rispedisce al server, che utilizza i dati inseriti per effettuare altre operazioni (accesso a data base o accesso a pagine riservate,....*

1. ***Il server WEB deve determinare se la pagina richiesta nell'URL non è una pagina WEB ma un programma,***
2. ***Il server deve localizzare il programma e controllare che possa essere eseguito,***
3. ***In caso positivo, il processo server deve attivare lo script utilizzando i dati in input provenienti dal client browser,***
4. ***Il processo server rimane in attesa del completamento della esecuzione dello script per passare l'output al client,***
5. ***Il processo server chiude la connessione.***

Web con pagine attive (Applet)



Pagine attive (JavaScript)

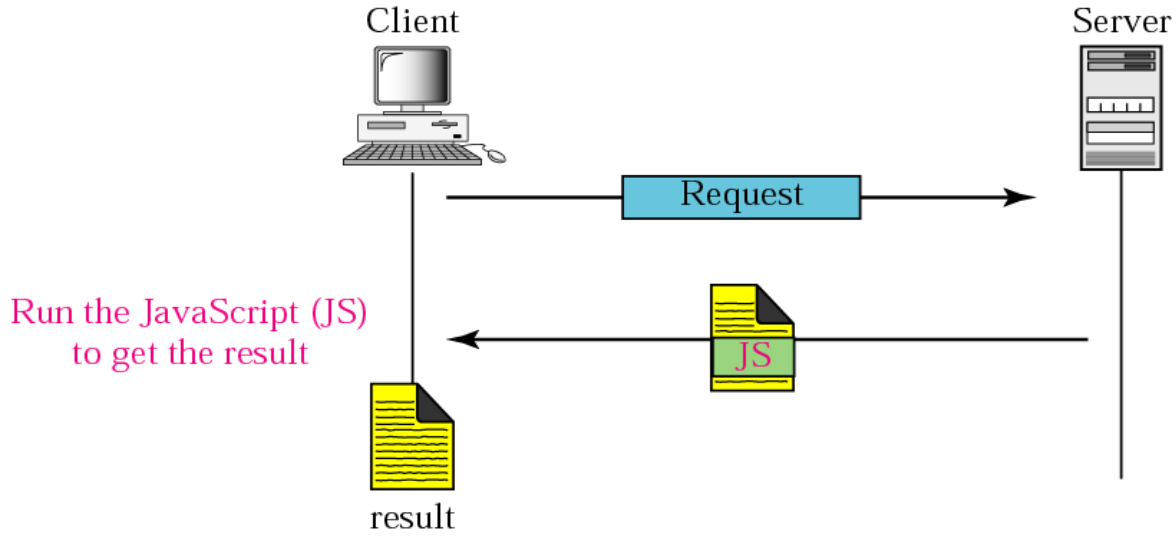


figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

I protocolli del WWW

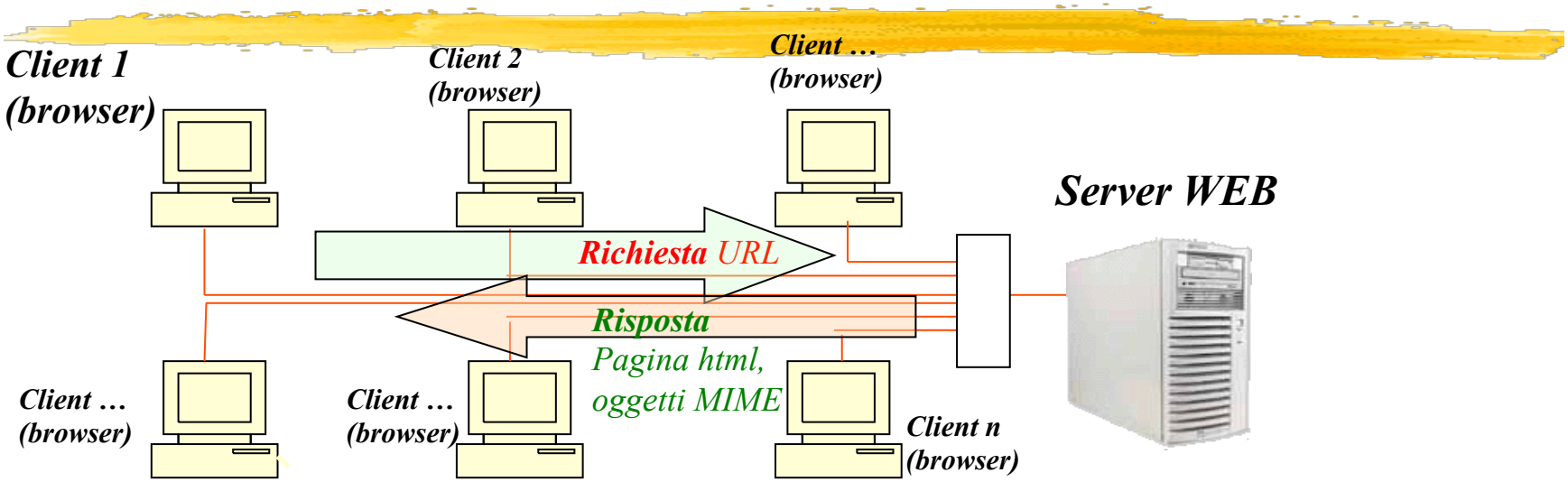
Architettura basata sui meccanismi standard di comunicazione definiti nella suite TCP/IP + i tre nuovi standard URL, HTTP, HTML,

URL = Uniform Resource Locator = ***sistema di indirizzamento*** delle risorse (insieme di meccanismi di naming) basato su un meccanismo di riferimento a tutte le risorse presenti su WEB (testo, immagini, suoni,…).

HTTP = HyperText Transfert Protocol = ***protocollo di comunicazione*** a livello applicativo (del modello ISO-OSI) basato sulla suite TCP/IP per la comunicazione tra un client (browser WEB) ed un server WEB.

HTML = HyperText Markup Language = ***linguaggio*** per visualizzare l'informazione in formato ipertestuale (ipermediale), ben formattata ed in modo indipendente dalla piattaforma.

Il protocollo di comunicazione del WWW



1. Fase di lookup, *durante la quale il client utilizza il naming system globale di Internet (**DNS**) per risalire dall'indirizzo mnemonico in formato alfanumerico del sito WEB all'indirizzo IP del server host del sito **(resolver).***

2. Se il name server locale non conosce *l'indirizzo IP della destinazione della richiesta, esso deve cercare la **risoluzione** del nome su altri DNS server autoritativi.*

3. Il client riceve l'indirizzo IP *della macchina server WEB.*

4. Il client stabilisce una connessione *TCP/IP con una **porta (80)** della macchina server WEB e trasmette la richiesta della pagina del sito WEB utilizzando il protocollo (a livello applicativo) HTTP.*

5. Il server WEB, *esaminata la richiesta, invia al client un codice numerico (nell'header) seguito dal risultato della richiesta..*

6. Il client *o il server* **chiudono la connessione**

Uniform Resource Locator (URL)

L'***URL*** è un meccanismo di indirizzamento globale che consente di specificare in maniera univoca la risorsa alla quale il client WEB è interessato.

Un URL indica il tipo di risorsa alla quale si sta accedendo (ex. http, gopher, ftp), il server che la possiede, il cammino per ritrovarla ed il suo nome.

schema:// host.domain [:port] / path [/filename [.xxxx]] [#anchor-id]

Dove:

- ***schema =*** ***indica il modo con cui accedere ad una risorsa****, cioè il protocollo da usare per interagire con il server che controlla la risorsa.*
  Può assumere i seguenti valori : **file** *(un file sul computer locale);* **http** *(valore di default, denota un file html sul World Wide Web server) ;* **gopher** *(un file su un Gopher server);* **telnet** *(per effettuare login remoti);* **ftp** *(per il trasferimento file);* **mailto** *(per spedire posta elettronica).*
- ***host.domain =*** ***indica il nome del server WEB dove risiede la pagina WEB.***
- ***port = il numero della porta*** *viene generalmente omesso, di default per il WEB è la* ***porta 80.***
- ***path =*** *localizza la risorsa WEB sul file system del server.*
- ***filename =*** *specifica il nome della risorsa ed il suo formato. In caso di assenza si assume* ***index.html*** *come* ***nome di default.***
- ***#anchor-id*** *rappresenta una parte all'interno di una pagina (risorsa).*

Ad esempio: ***"http://www.di.uniba.it/~reting"***

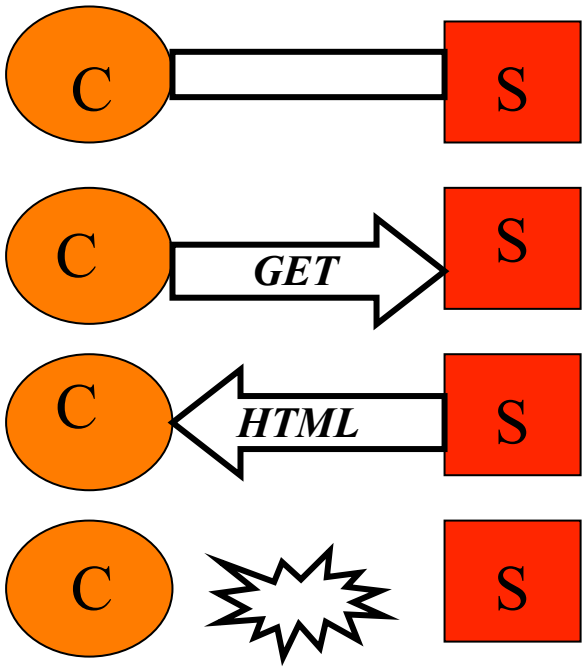
schema :// host.domain / filename

HTTP- 1.0

E' un protocollo di richiesta e risposta **client-server** basato su TCP/IP.

I dati trasportati tramite il protocollo HTTP sono in genere di tipo HTML o XML.

il *browser* può fare soltanto una domanda alla volta per evitare di sovraccaricare il *server* tenendo aperti dei canali che non si sa quando verranno utilizzati.



1) ***Apertura connessione*** : Il ***browser sul client*** vuol prelevare dati dal *server:* istituisce una **connessione** su canale *TCP* relativo alla porta 80 del *server*.

2) ***Richiesta http*** - Una volta **stabilito il collegamento,** il *browser* manda la sua **richiesta di informazione,** (**comando *GET*).**

3) **Risposta http** - Il ***server*** **riceve la richiesta** di informazioni e **fornisce la risposta** (tipicamente un *file* HTML o XML) .

4) **Chiusura connessione** - risposta arrivata, **il *server* chiude il collegamento.**

HTTP 1.0

Protocollo testuale, pseudo-anonimo e NON PERSISTENTE:

I messaggi scambiati sono caratteri ASCII (anche se i dati trasmessi dal server possono essere immagini, suoni, ecc…)

Il server conosce solo l'indirizzo IP del client, per cui due richieste dallo stesso indirizzo IP sono considerate provenienti dallo stesso utente.

Non è previsto il concetto di sessione all'interno della quale si preserva l'informazione sullo stato dell'informazione tra client e server = nessuna relazione tra due richieste, anche se si riferiscono ad oggetti contenuti nella stessa pagina html.

HTTP-1.1

HTTP-1.1 è l'evoluzione dell'http 1.0 e permette, sotto controllo del sistemista che gestisce il server, di creare :

- **connessioni persistenti** (permette al *browser* di creare un canale e di mantenerlo aperto facendo più richieste sul medesimo canale);
- **negoziare** (browser e server) **il formato dei dati**, es: se il server ha a disposizione i dati in più formati diversi, è possibile fornire al client i dati nel formato migliore.
- **gestione della *cache***; i nodi intermedi, cosiddetti ***proxy***, possono tener copia locale dei dati, per evitare ogni volta di andarli a prendere dal server. E' possibile specificare all'interno del protocollo qual è la modalità per la gestione della cache.
- **quattro nuovi metodi che il *browser client*** può fare sul *server*: ***PUT***, permette di mettere un intero documento sul *server*, ***DELETE***, permette di cancellarlo, ***TRACE***, vedere qual è la sequenza dei *proxy* tra il *browser* e il *server* e ***OPTIONS***, conoscere quali sono le opzioni del canale HTTP presente.

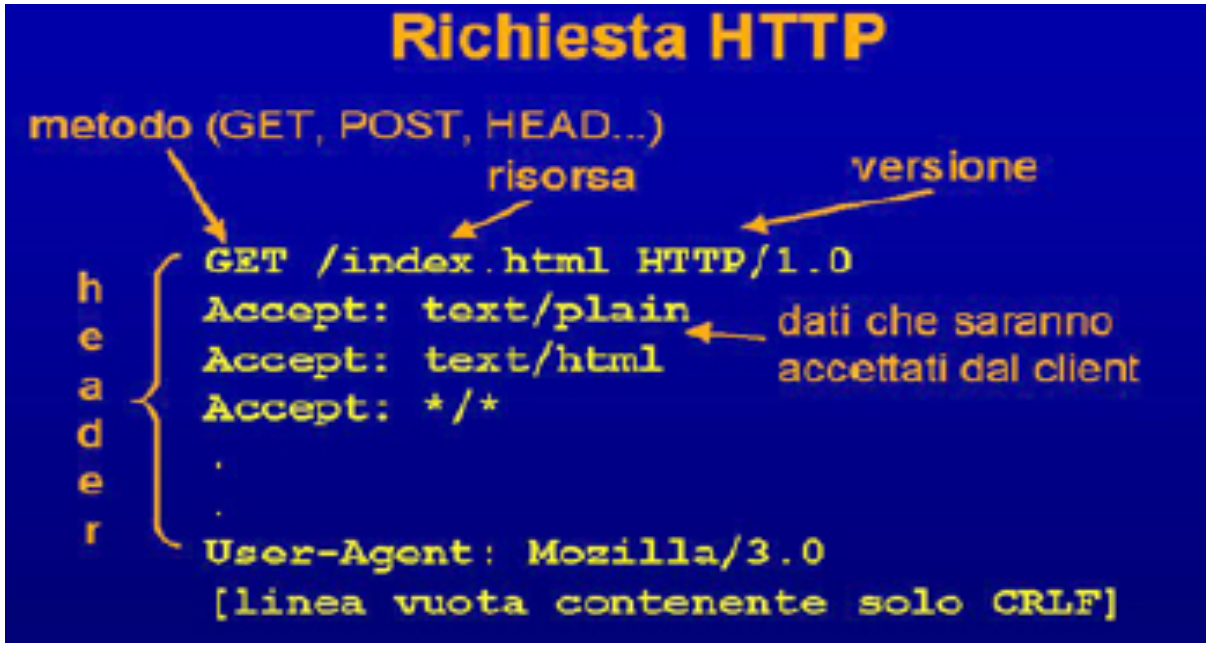
I metodi di http1.1 usabili dal client sul server

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Enquires about available options

figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Richiesta HTTP

è composta di due parti, una parte di ***header*** separata tramite una linea vuota dalla parte di ***body*** (opzionale).
primo parametro: metodo utilizzato (GET, HEAD, POST, PUT, DELETE, LINK, UNLINK, TRACE),
secondo elemento: individua la risorsa desiderata.
terzo parametro: versione del protocollo
altri parametri : per esempio, il *client* specifica che accetterà *file* di tipo *text* in formato *plain* o in formato HTML e che si tratta di un *client* di tipo Mozilla 3.0.



Risposta HTTP

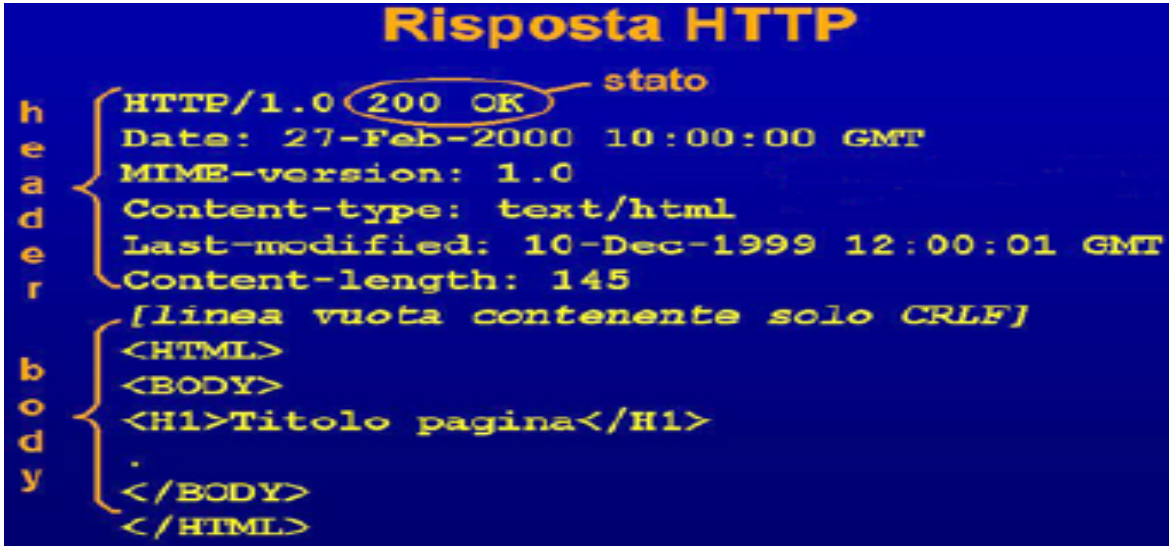
Riga di stato: informazione di stato ***three digit code***: prima cifra specifica il fatto che la richiesta sia andata a buon fine (iniziano con il numero 2)o no; le altre due cifre consentono di specificare che tipo di errore si è verificato, es: 404. (Tutti i messaggi di errore sul client cominciano con il codice 4, errori sul server con 5).

header*:* - versione del protocollo HTTP che il server utilizzerà per la risposta.
- tipo e sottotipo di documento che spedirà.

*(**MIME**: **Multipurpose Internet Mail Extension**, standard che definisce le regole per lo scambio di informazioni che potrebbero avere parti non testuali)*

body*: es.* pagina HTML che il *server* sta inviando al *client*.

altri informazioni: data in cui è avvenuta la connessione, data in cui è stato modificato per l'ultima, dimensione del documento che sta trasferendo.



Esempio: Get e Post

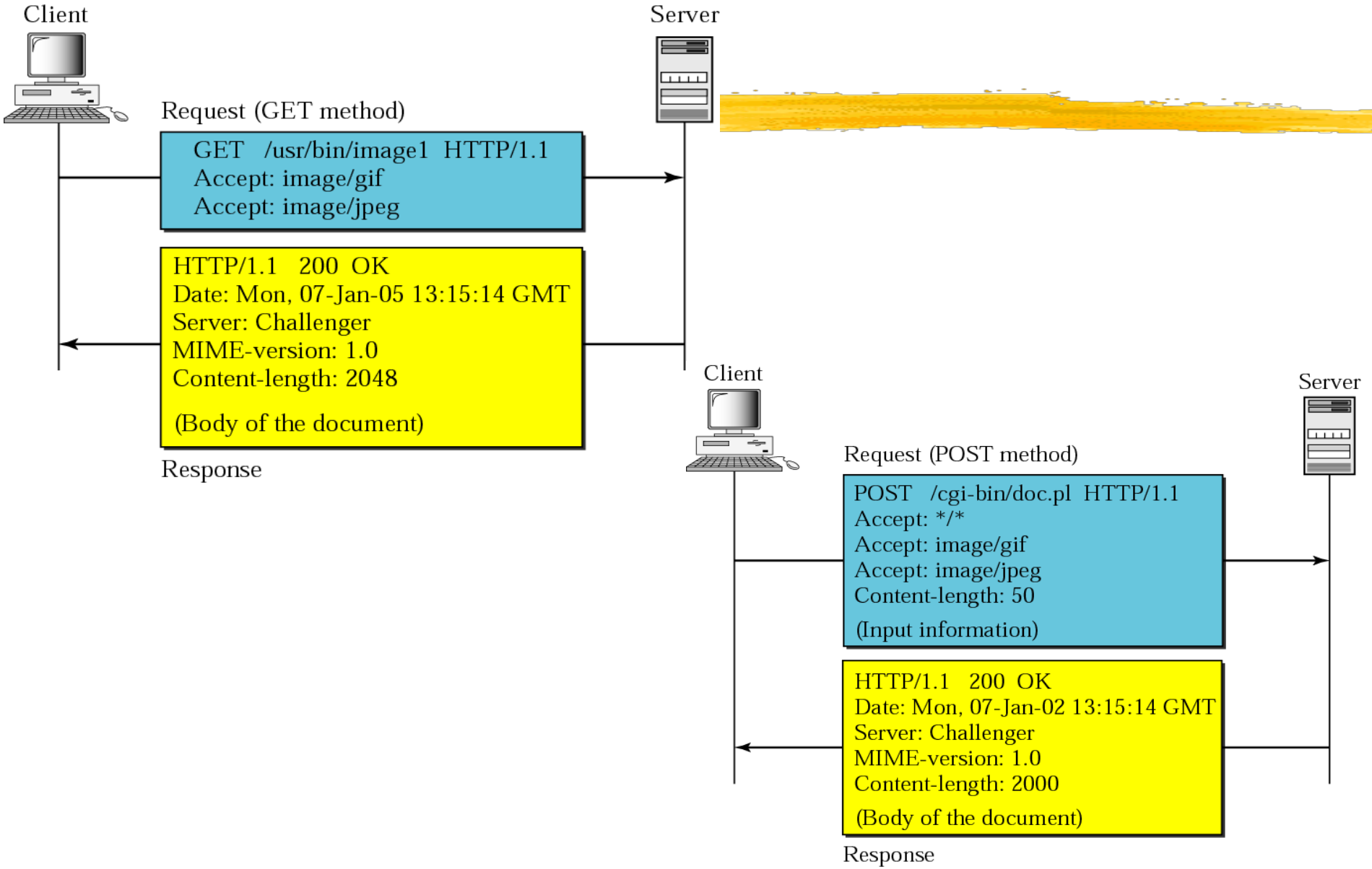
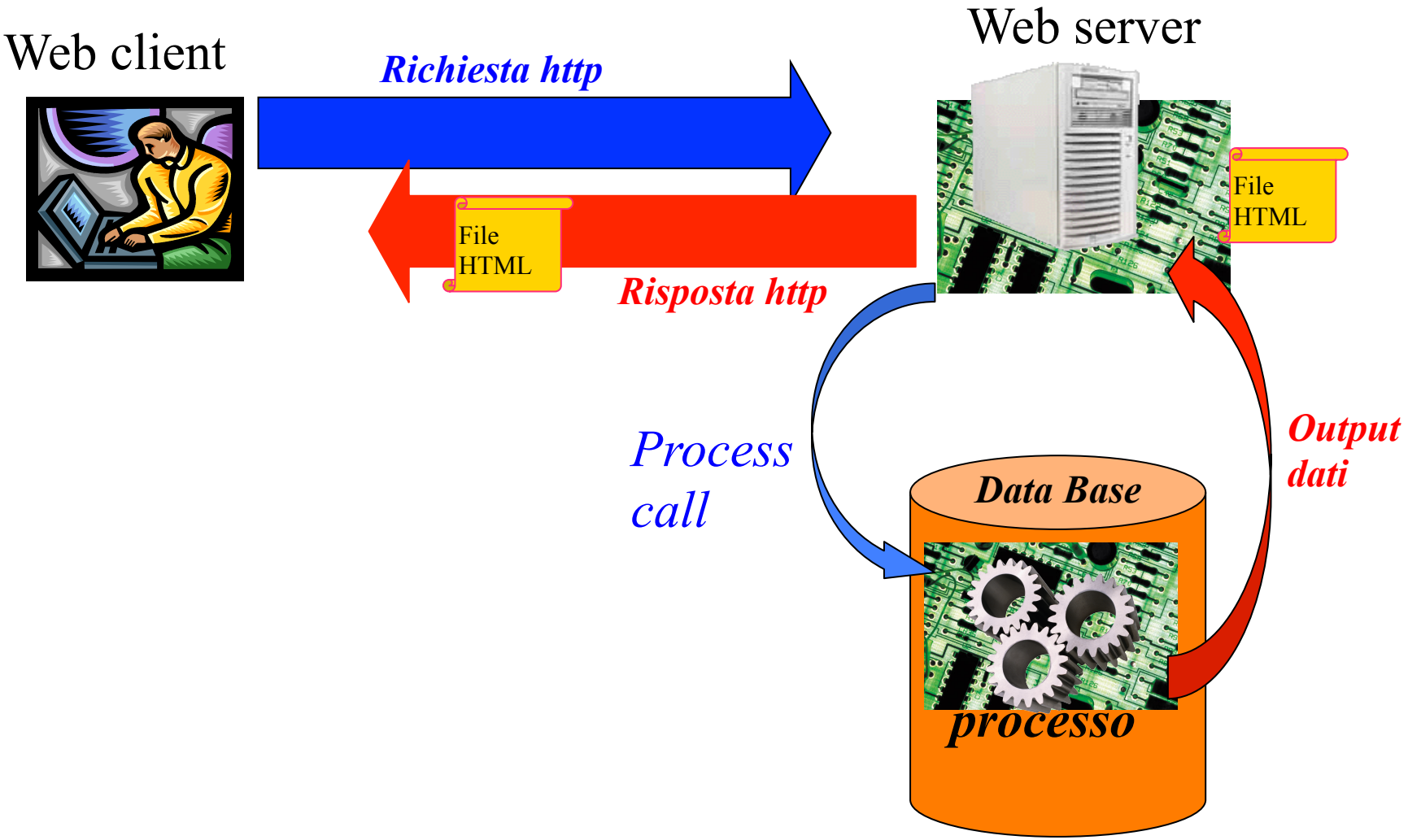


figura da : Behrouz A. Forouzan, *I protocolli TCP/IP* ed. McGraw-Hill 2005

Architettura WEB 3-tier



La tecnologia Common Gateway Interface

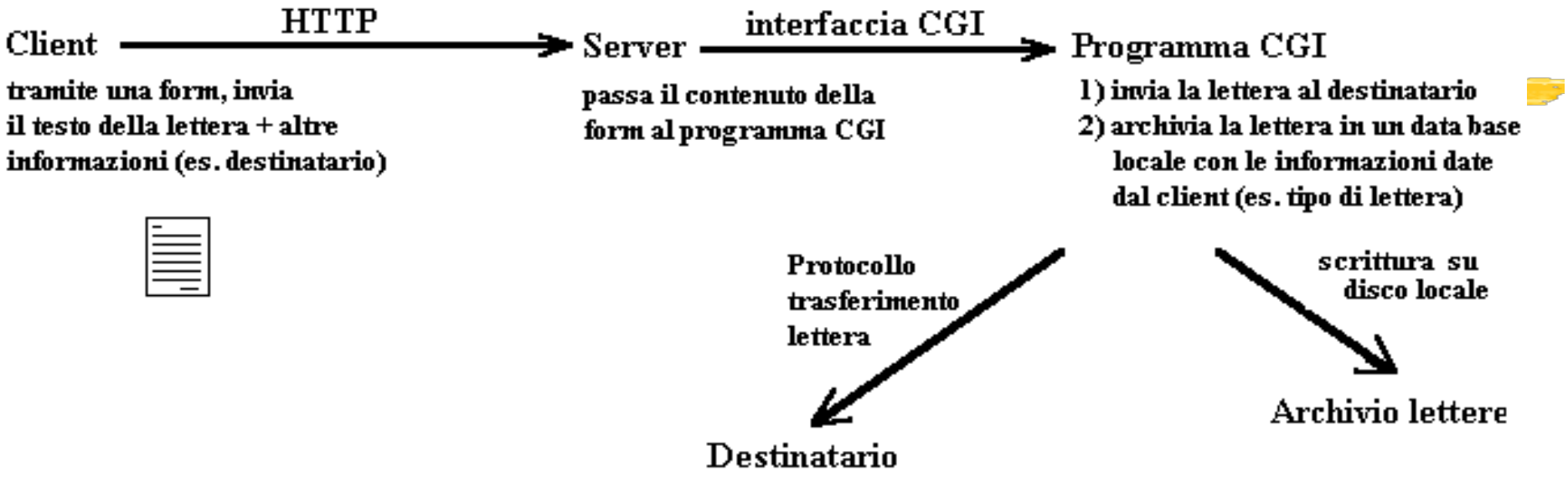
è la tecnologia standard usata dai web server per interfacciarsi con applicazioni esterne.

- ***Se l'URL invocato dal client corrisponde ad un programma CGI****, il server lo esegue in tempo reale, generando dinamicamente informazioni.*

- *Un* ***programma (o uno script) CGI*** *può essere scritto in qualsiasi linguaggio di programmazione (* ***C/C++, Perl, PHP, Visual Basic, Tcl/Tk, AppleScript,*** *ecc.), la scelta si basa sul sistema su cui girerà;*

- *il* ***Perl,*** *il* ***PHP, Python*** *e l'****ASP*** *sono i linguaggi più comunemente utilizzati.*

CGI



Il server deve "rendersi conto" che post-query non è un semplice documento HTML ma un programma CGI che deve essere eseguito da una shell di sistema operativo.

Perché ciò accada è necessario:

- *che i programmi CGI siano contenuti tutti in un'apposita directory;*
- *che nella configurazione del server sia specificato il path ove trovare i programmi CGI e l'identificatore che indica che è richiesta l'esecuzione di una applicazione. Di solito si sceglie come identificatore "/cgi-bin/" .*

HTML: HyperText Markup Language

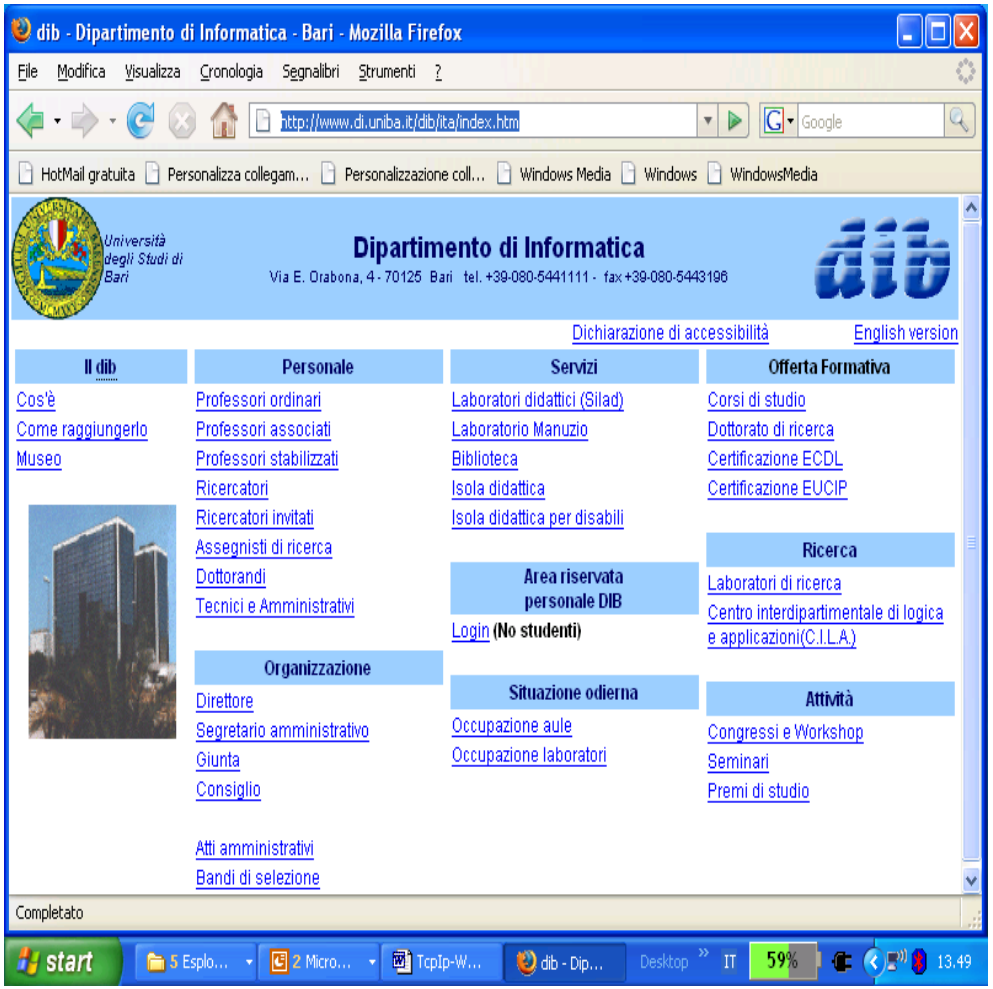
HTML è un linguaggio di markup che permette di definire una pagina ipermediale indipendentemente dal tipo di macchina da cui viene interpretato.
Markup = informazioni aggiunte ad un testo per specificare: sintassi, semantica, formattazione,….
Tramite l'inserimento di segni esterni (**tag**) al contenuto puro si specificano gli effetti da riprodurre sul contenuto stesso.
Documento HTML: file di testo con estensione ***.html*** (o .htm per i sistemi Microsoft)

Documento HTML = TESTO + TAG

TAG: delimitano la porzione di contenuto su cui applicare l'effetto. Ogni direttiva di markup ha
- *un nome: P H1 H2*
- *un tag di apertura e (non sempre) un tag di chiusura*
  p.es. apertura = <P> e chiusura = </P>.
- *eventuali attribuzioni opzionali nella forma* ***nome*** *= "****valore****"*

Esistono strumenti *WYSIWYG* per creare file .html (*Composer Mozilla,Dreamweaver, FrontPage,...)* che offrono anche funzionalità per pubblicare documenti html.

Esempio di documento HTML




```
<html lang=it><head> <meta name="description"
content="Dipartimento di Informatica -Bari-">
<meta name="Keywords" content="dipartimento
di informatica,università di
informatica,informatica,laurea triennale in
informatica,nuovi corsi in informatica,corso di
laurea in informatica, informatica bari"><title>dib
- Dipartimento di Informatica - Bari</title>
                <script language='JavaScript1.2'
src='../javascript/intestazione_ita.js' type="text/
javascript"></script> <link rel=stylesheet href="../
css/css_dipartimento.css" type="text/css"></
head><body><script language='JavaScript1.2'
type="text/
javascript">document.write(stampa_intestazione()
)</script><noscript><DIV
align="center"><FONT face="Arial, Helvetica,
sans-serif" color=#000066
size=6><B>Dipartimento di Informatica</B></
FONT><BR>  <FONT face="Arial, Helvetica,
sans-serif" color=#000066 size=4>Università degli
Studi di Bari</FONT></DIV>  <a
href="http://www.uniba.it" title="Homepage
Università degli Studi di Bari".....
```