

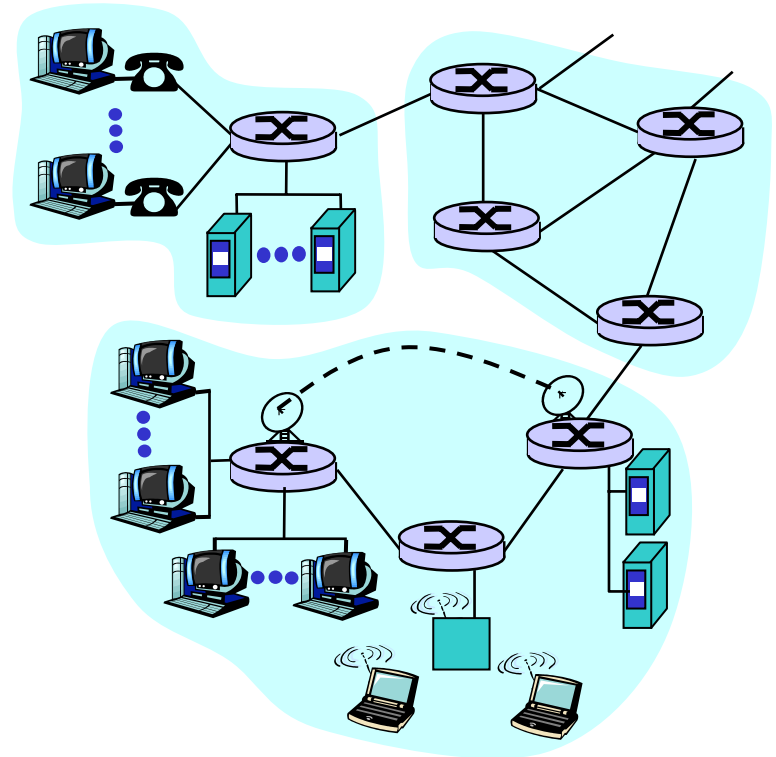
Applicazioni e protocolli a livello applicazione

Applicazione: processi distribuiti comunicanti

- vengono eseguiti sugli host di rete come processi utente
- scambio di messaggi per implementare l'applicazione
- e.g., email, ftp, Web

Protocolli a livello Applicazione

- una parte di un'applicazione
- definiscono i messaggi scambiati dall'applicazione e le azioni intraprese
- Usano i servizi di comunicazione forniti da protocolli a livello sottostante (TCP, UDP)





Applicazioni di rete: terminologia

Processo: programma in esecuzione in un host.

- processi in esecuzione su host diversi (distanti) comunicano con un **protocollo a livello applicazione**

- **user agent:** processo software, che si interfaccia con l'utente "verso l'alto" e con la rete "verso il basso".

- implementa il protocollo a livello applicazione
- **Web:** browser
- **E-mail:** mail reader
- **streaming audio/video:** media player

Paradigma Client-server

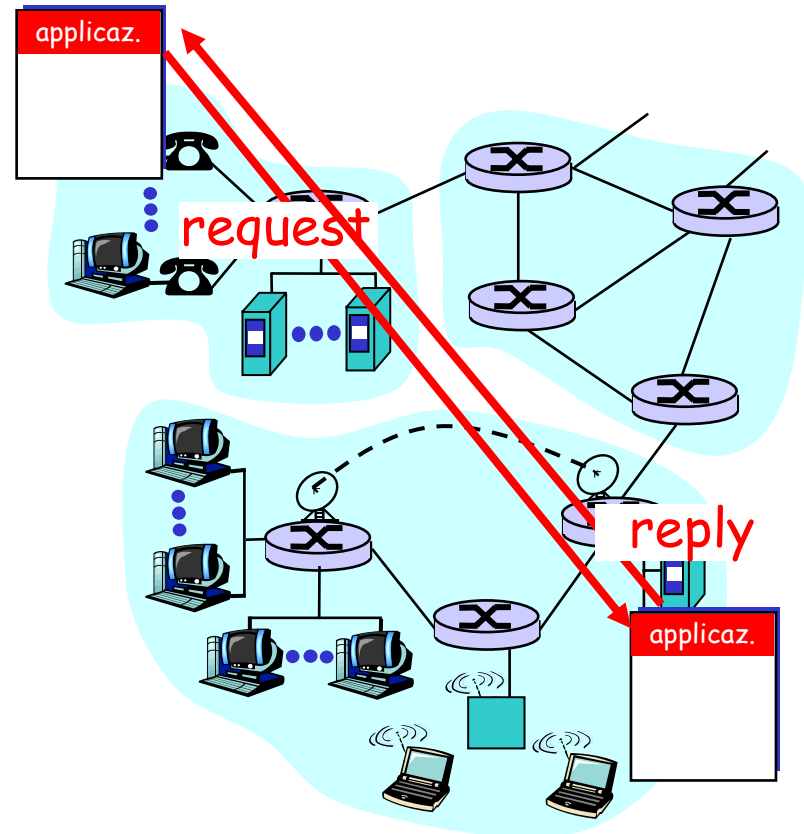
Una tipica applicazione di rete si compone di due parti: *client* e *server*

Client:

- Avvia il contatto con il server ("parla per primo")
- Solitamente, richiede un servizio al server
- **Web:** il client è implementato nel browser; **e-mail:** in mail reader

Server:

- Fornisce il servizio richiesto al client
- e.g., il Web server spedisce la pagina Web richiesta, il mail server recapita l'e-mail



Di quale servizio di trasporto necessita un'applicazione?

Perdita di dati

- Alcune applicazioni (e.g., audio) possono tollerare perdite
- Altre applicazioni (e.g., trasferimento file, telnet) richiedono un trasferimento dati affidabile al 100%

Time-sensitive

- Alcune applicazioni (e.g., telefonia su Internet, giochi interattivi) richiedono piccoli ritardi

Larghezza di banda

- Alcune applicazioni (e.g., multimediali) richiedono un ammontare minimo di larghezza di banda per essere "efficaci"
- Altre applicazioni ("applicazioni elastiche") fanno uso di qualunque larghezza di banda riescono ad ottenere

Requisiti del servizio di Trasporto di applicazioni comuni

Applicazione	Perdite dati	Larghezza di banda	Time Sensitive
trasferimento file	senza	elastica	no
e-mail	senza	elastica	no
documenti Web	tollerante	elastica	no
real-time audio/video	tollerante	audio: 5Kb-1Mb video: 10Kb-5Mb	si, alcuni 100 msec
stored audio/video	tollerante	come sopra	si, pochi secs
giochi interattivi	tollerante	alcuni Kbps	si, alcuni 100 msec
applicazioni finanziarie	senza	elastica	si e no

Servizi di trasporto in Internet

Servizio TCP:

- *connection-oriented*: fase iniziale di "setup" necessaria tra client e server
- *trasporto affidabile* tra processo mittente e destinatario
- *controllo di flusso*: il mittente non sovraccaricherà il ricevitore
- *controllo di congestione*: regolazione della velocità del mittente quando la rete è sovraccarica
- *non fornisce*: tempi, garanzie su larghezza di banda minima

Servizio UDP:

- trasferimento dati non affidabile tra processo mittente e processo destinatario
- non fornisce: setup della connessione, affidabilità, controllo di flusso, controllo di congestione, tempi o larghezza di banda garantiti

Domanda: perché mai esiste UDP?

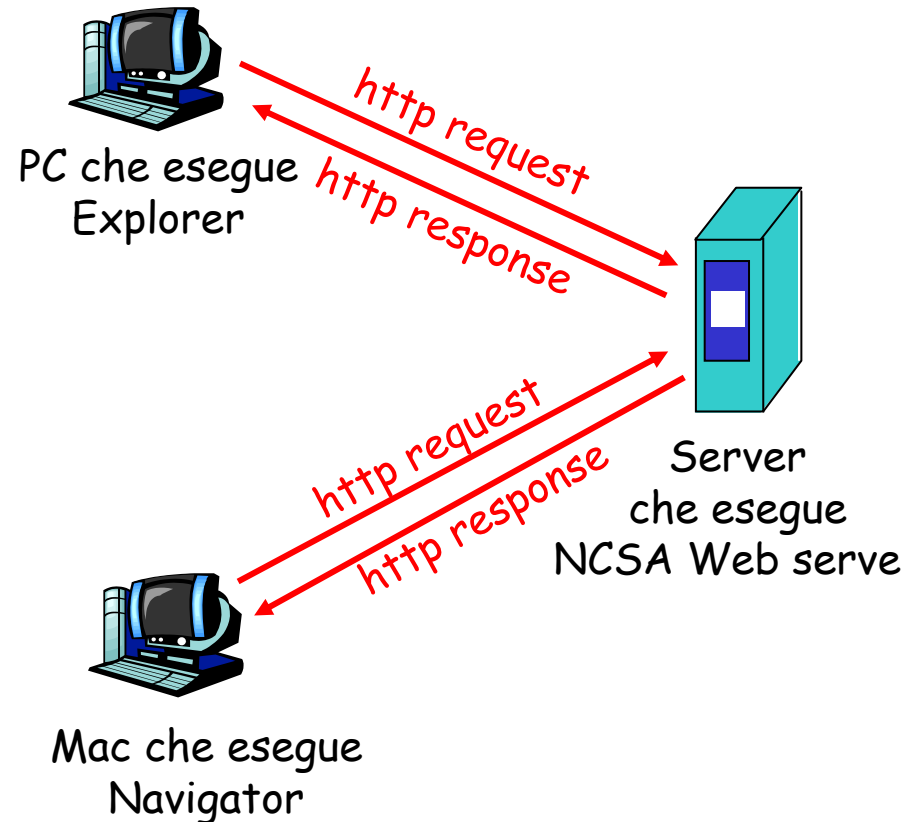
Applicazioni Internet: protocolli a livello applicazione e trasporto

	Applicazione	Protocollo a livello applicazione	Protocollo a livello trasporto sottostante
	e-mail	smtp [RFC 821]	TCP
accesso a terminale remoto		telnet [RFC 854]	TCP
	Web	http [RFC 2068]	TCP
	trasferimento file	ftp [RFC 959]	TCP
streaming multimedia		proprietario (e.g. RealNetworks)	TCP or UDP
	file server remoto	NSF	TCP or UDP
	telefonia su Internet	proprietario (e.g., Vocaltec)	solitamente UDP

Il Web: il protocollo http

http: hypertext transfer protocol

- Protocollo a livello applicazione per il Web
- Modello client/server
 - *client*: il browser che richiede, riceve e mostra oggetti Web
 - *server*: Web server che spedisce oggetti in risposta ad una richiesta
- http1.0: RFC 1945
- http1.1: RFC 2068-2616



Il WEB: terminologia

- **pagina WEB (documento):** collezione di oggetti
- **oggetto:** un file (HTML, JPEG, ...)
- **file HTML base:** con direttive e riferimenti ad altri oggetti
- **URL (Uniform Resource Locator):** meccanismo di identificazione risorse. Si compone del nome del host sul quale risiede l'oggetto e il path-name dell'oggetto

- www.di.unito.it/various/presentation_en.html

nome host

path-name

Il protocollo http

http è "stateless"

- il server non mantiene alcuna informazione sulle richieste passate dei client

http: usa servizio TCP:

- il client avvia una connessione TCP con il server
- il server accetta la connessione TCP dal client
- vengono scambiati messaggi http (messaggi del protocollo di livello applicazione) tra il browser (client http) ed il Web server (server http)
- la connessione TCP viene chiusa

nota

I protocolli che mantengono lo stato sono complessi!

- Tutta la storia passata della connessione (stato) deve essere mantenuta, memorizzata
- se server o client subiscono un crash, la loro conoscenza dello stato può essere inconsistente e deve essere ricostruita

http: esempio

Supponiamo

www.someSchool.edu/someDepartment/home.index

l'utente

digiti

l'URL

(contiene testo e
10 riferimenti ad
immagini jpeg)

1a. il client http inizia una
connessione TCP al server http
(che è un processo)
all'indirizzo
www.someSchool.edu.

1b. il server http sull' host
www.someSchool.edu in attesa di
connessioni TCP. "Accetta" la
connessione, notificandola al client

2. il client http spedisce il messaggio
http *request message* (contenente
l'URL) usando la connessione TCP

3. il server http riceve il messaggio di
richiesta, forma un messaggio http
response message contenente
l'oggetto richiesto
(someDepartment/home.index),
e spedisce un messaggio usando la
connessione TCP

tempo

http: esempio (continuazione)

4. il server http chiude la connessione TCP.

5. il client http riceve il messaggio di risposta contenente il file html e lo mostra. Parsifica (analizza) il file html, trova i riferimenti a 10 oggetti jpeg.

6. passi 1-5 si ripetono per ognuno dei 10 oggetti jpeg.

emppo

Formato dei messaggi http: request

- due tipi di messaggi http: *request, response*
- *http request message:*
 - ASCII (formato human-readable)

request line
comandi GET, POST,
HEAD)

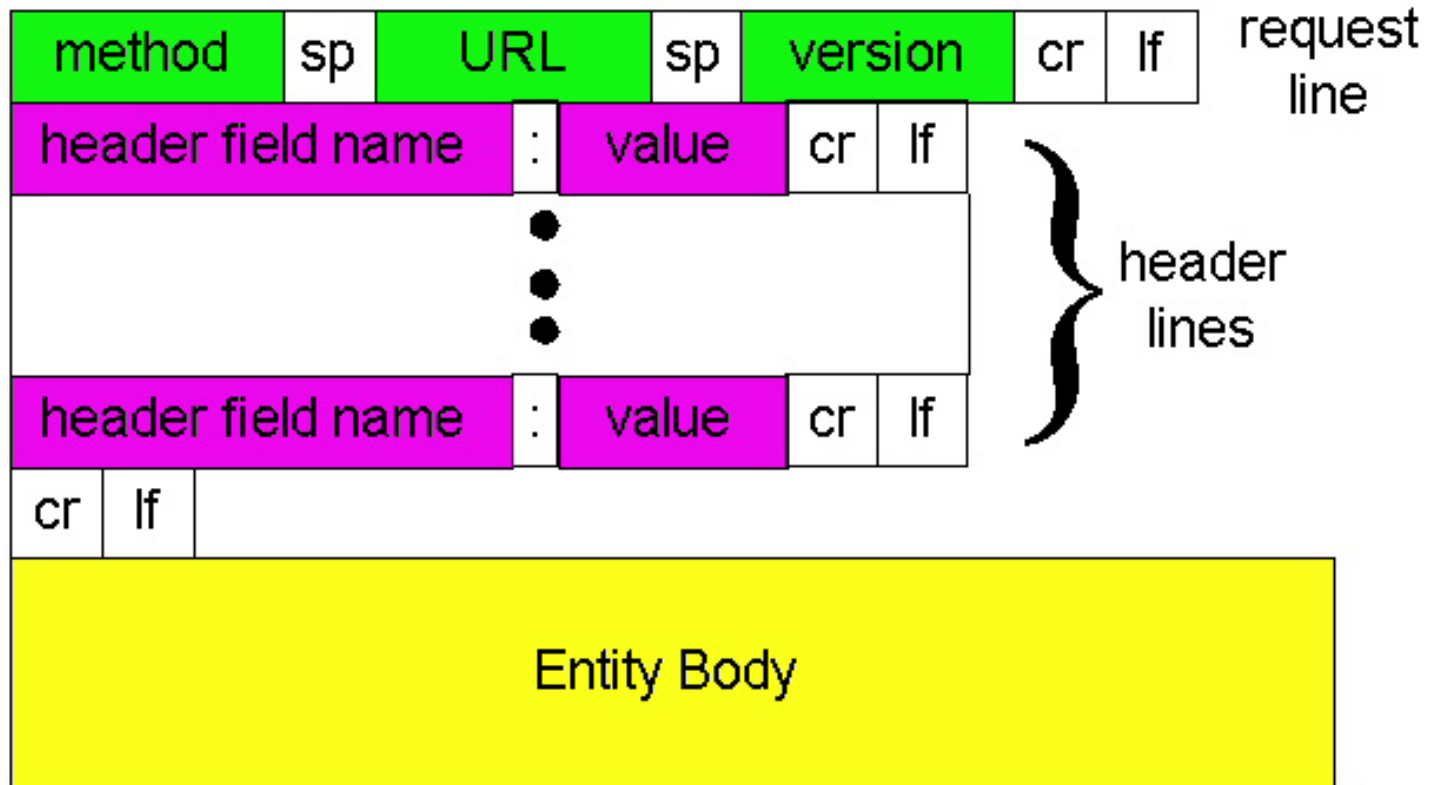
linee
header

```
GET /somedir/page.html HTTP/1.0
Host: www.someschool.edu
User-agent: Mozilla/4.0
Accept: text/html, image/gif, image/jpeg
Accept-language: fr
```

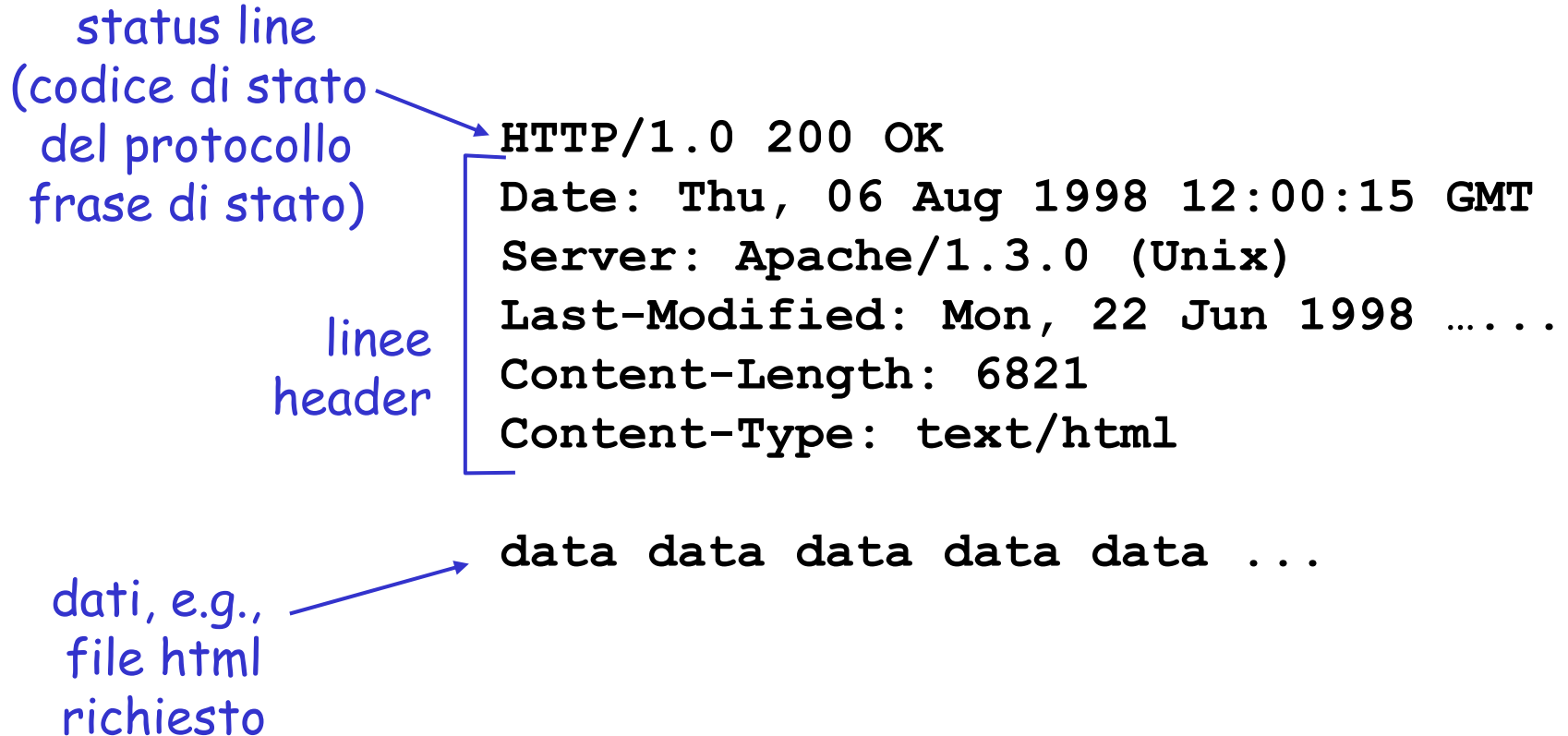
Carriage return,
line feed
indicano la fine
del messaggio

(extra carriage return, line feed)

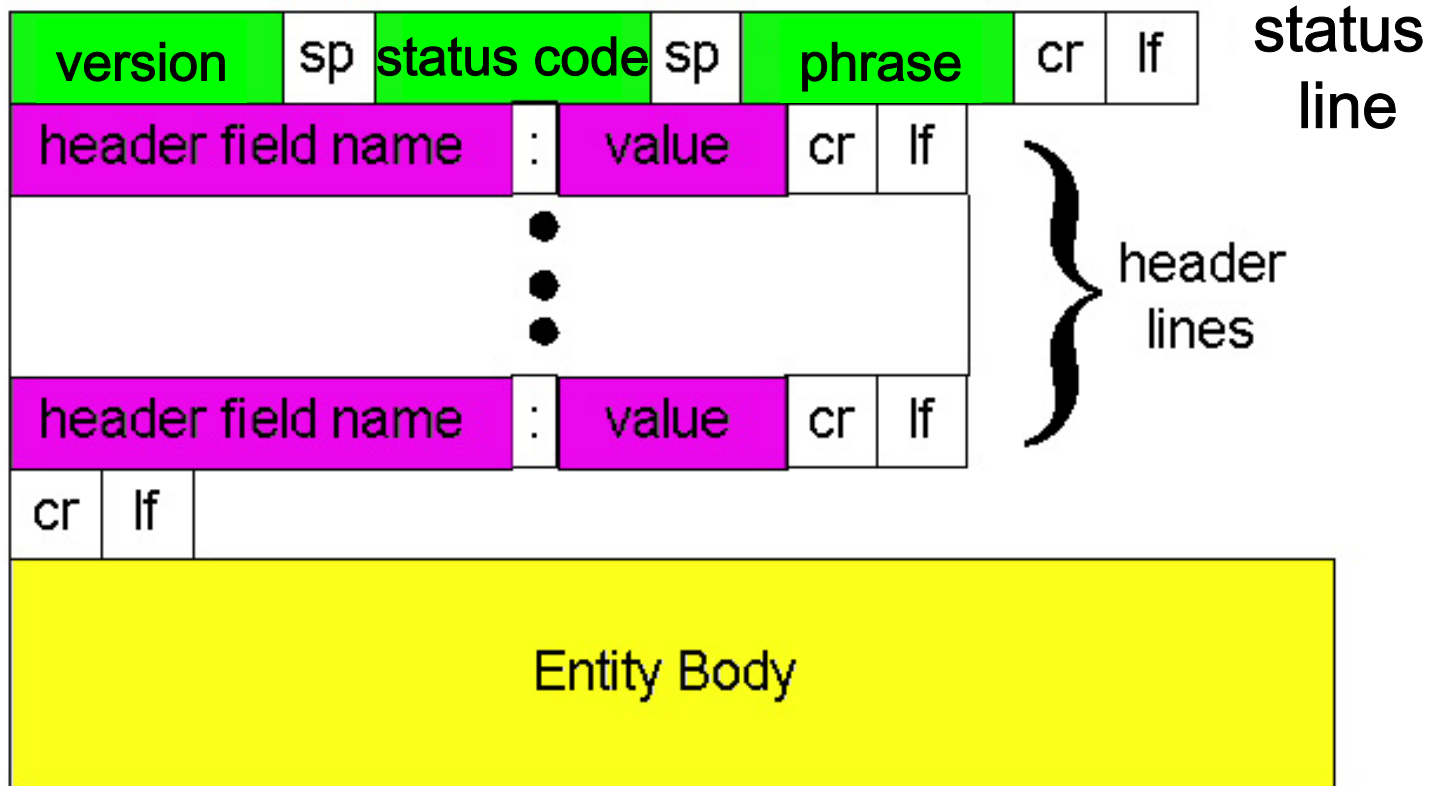
http request message: formato generale



Formato dei messaggi http: response



http response message: formato generale





Codici di stato per http response

Nella prima linea del response message server->client.

Alcuni codici d'esempio:

200 OK

- richiesta con successo, l'oggetto richiesto segue in questo messaggio

301 Moved Permanently

- L'oggetto richiesto è stato spostato, la nuova locazione è specificata dopo in questo messaggio (Location:)

400 Bad Request

- request message non compreso dal server

404 Not Found

- Documento richiesto non trovato su questo server

505 HTTP Version Not Supported



Provate http (lato client)

1. Collegatevi con telnet ad un Web server:

```
telnet www.eurecom.fr 80
```

Aprire una connessione TCP su `www.eurecom.fr`. Qualunque cosa si digiti viene spedita al web server su `www.eurecom.fr`

2. Digitate un http request GET:

```
GET /~ross/index.html HTTP/1.0
```

Digitando questo (digitate due volte carriage return), spedite un minimale (ma completo) GET request al server http

3. Guardate il response message spedito dal server http!



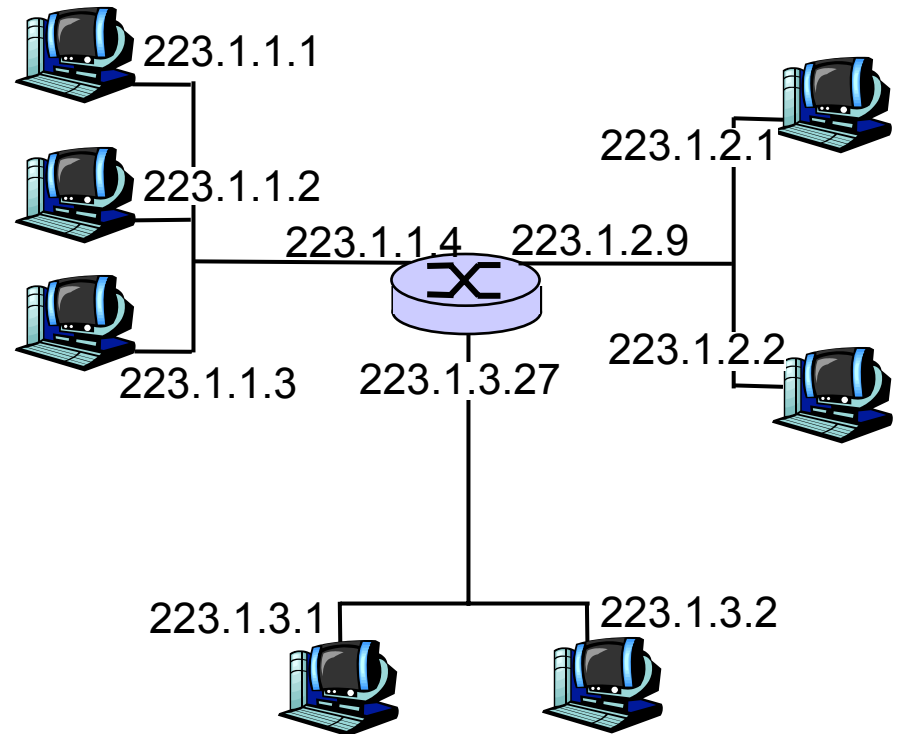
Protocolli a livello applicazione

domanda: come fa un processo ad identificare l'altro processo con il quale vuole comunicare?

- "*numero di porta*" - *permette all' host che riceve di determinare a quale dei processi che sta eseguendo (locali) debba essere recapitato il messaggio*
- Indirizzo IP dell' host sul quale è in esecuzione l'altro processo

Indirizzi IP: introduzione

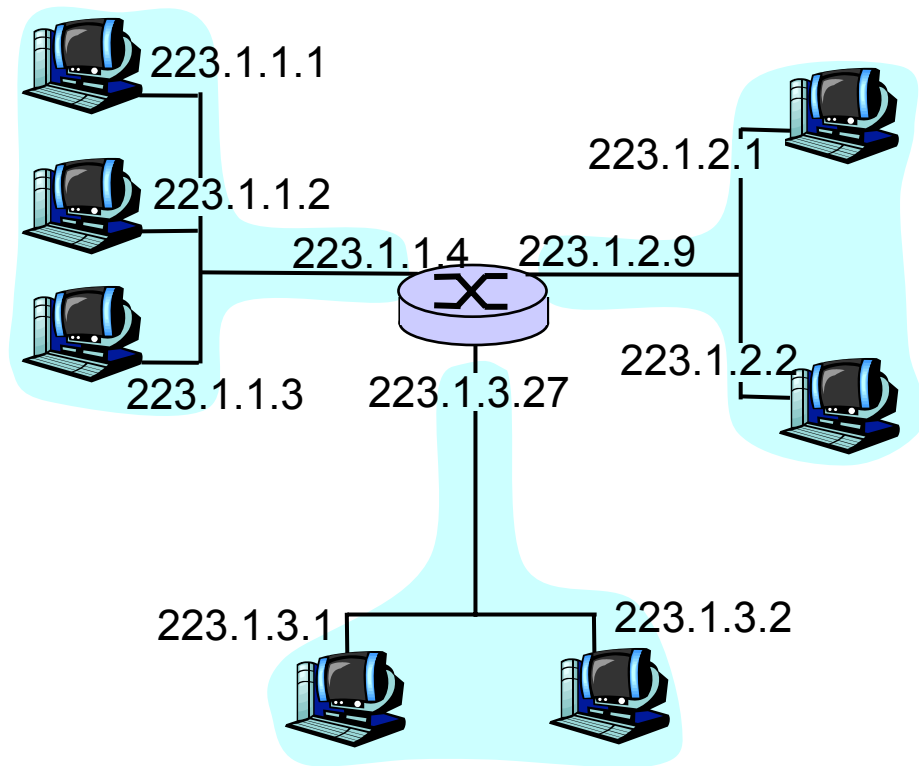
- **indirizzo IP:** identificatore di 32-bit per l'interfaccia di rete di host e router
- **interfaccia di rete:** connessione tra host, router ed il canale fisico
 - i router, tipicamente hanno interfacce multiple
 - gli host possono avere interfacce multiple
 - gli indirizzi IP sono associati alle interfacce e NON agli host o ai router



$$223.1.1.1 = \underbrace{11011111}_{223} \underbrace{00000001}_1 \underbrace{00000001}_1 \underbrace{00000001}_1$$

Indirizzi IP

- indirizzo IP:
 - parte rete (bit più a sinistra)
 - parte host (bit più a destra)
- *Cos'è una rete ?* (dal punto di vista dell'indirizzo IP)
 - dispositivi d'interfaccia con la stessa parte rete dell'indirizzo IP
 - possono fisicamente raggiungere l'un l'altra senza l'intervento di router



rete di 3 reti IP

(per gli indirizzi IP che iniziano con 223
i primi 24 bit sono l'indirizzo della rete)



DNS: Domain Name System

Persone: molti identificativi:

- # CF, nome, # passaporto

Host e router in Internet:

- indirizzo IP (32 bit) - usato per indirizzare i pacchetti
- "nome", e.g., pianeta.di.unito.it - usato dagli esseri umani

Domanda: corrispondenza tra indirizzo IP e nome ?

Domain Name System:

- *database distribuito* implementato con una gerarchia di *name server*
- *protocollo di livello applicazione* host, router, e name servers comunicano per *risolvere* nomi (traduzione indirizzo/nome)
 - nota: funzione chiave in Internet, implementata come protocollo a livello applicazione
 - complessità nella edge network