

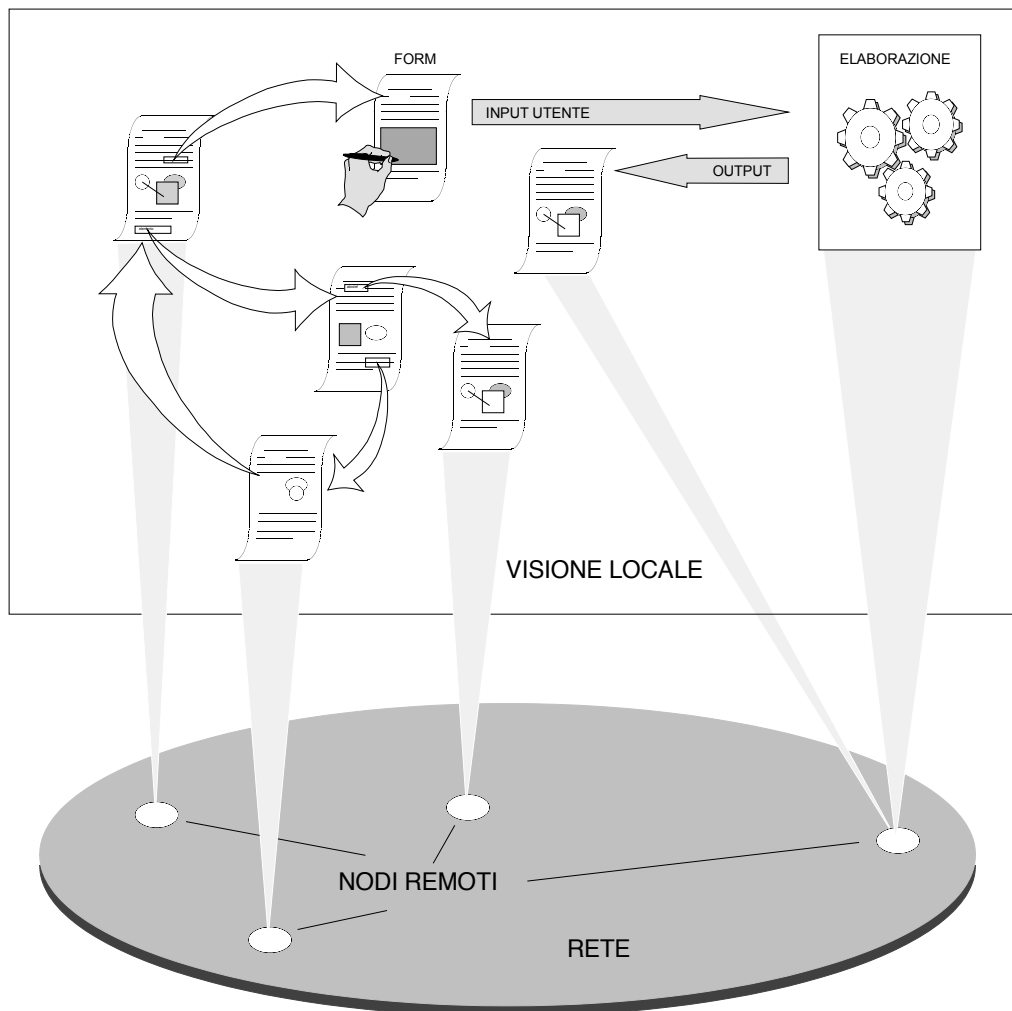
Strumenti trasparenti

unico insieme di informazioni accessibile a tutti i possibili utenti

Gopher

WWW (Explorer, Netscape)

strutturazione ipertestuale delle informazioni (trasparenza della allocazione delle informazioni) e uso di interfacce grafiche (semplicità di utilizzo)



Primo passo (gopher)

strumenti di visualizzazione a **caratteri** di informazioni

Creazione di un unico direttorio che nasconde le informazioni di allocazione

UNICO indice di informazioni per argomento

Allargamento della fascia di utenza

Secondo passo (WWW)

trasparenza allocazione (strutturazione ipertestuale)
gestione informazioni di tipo diverso (multimediali)

strumenti e protocolli con *informazioni multimediali*

- evoluzione della mail e altri tool tradizionali
- informazioni trasparenti e multimediali
- protocolli eterogenei

Il secondo approccio *amplia ulteriormente* la fascia di utenza (in modo esponenziale)

ma introduce problemi

*di **banda di occupazione** di risorse*

*di **sicurezza** delle informazioni*

*di **standardizzazione** delle informazioni*

*di **visualizzazione rapida** delle informazioni*

World Wide Web (WWW)

CERN (1989)

Progetto di integrazione in forma ipertestuale delle risorse esistenti in INTERNET

Scopi

- Trasparenza accesso e allocazione
- Presentazione multimediale
- Interfaccia unica per protocolli diversi
(integrazione con gli altri protocolli)
- Modificabilità e condivisione delle informazioni

Ampia scelta di interfacce testuali e grafiche
Possibilità di estensioni sperimentali del sistema

Componenti

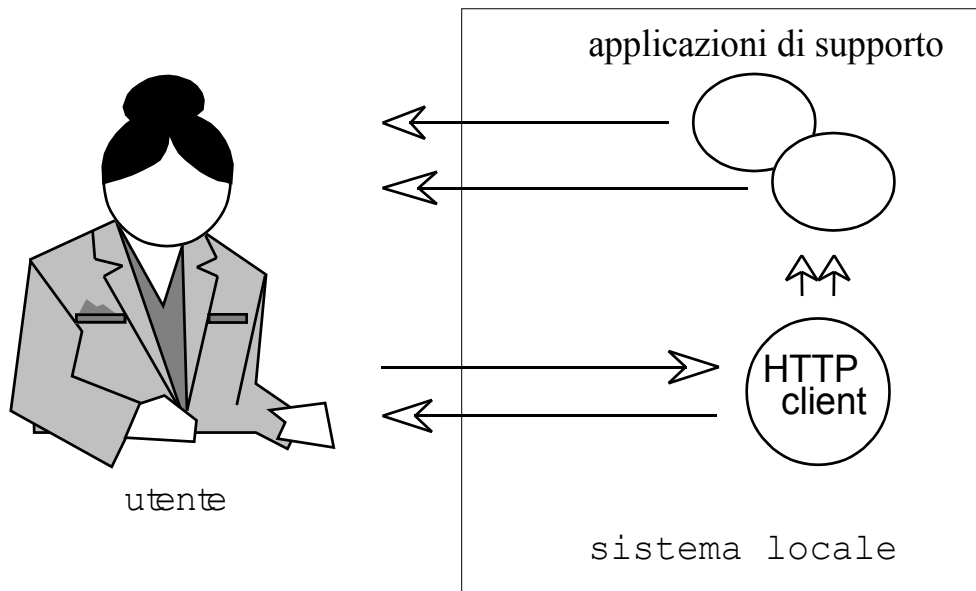
- Browser (presentazione e gestione richieste)
- Server (accesso e invio informazioni)
- Helper applications (particolari presentazioni)
- Applicazioni CGI (esecuzione remota)

Specifiche standard

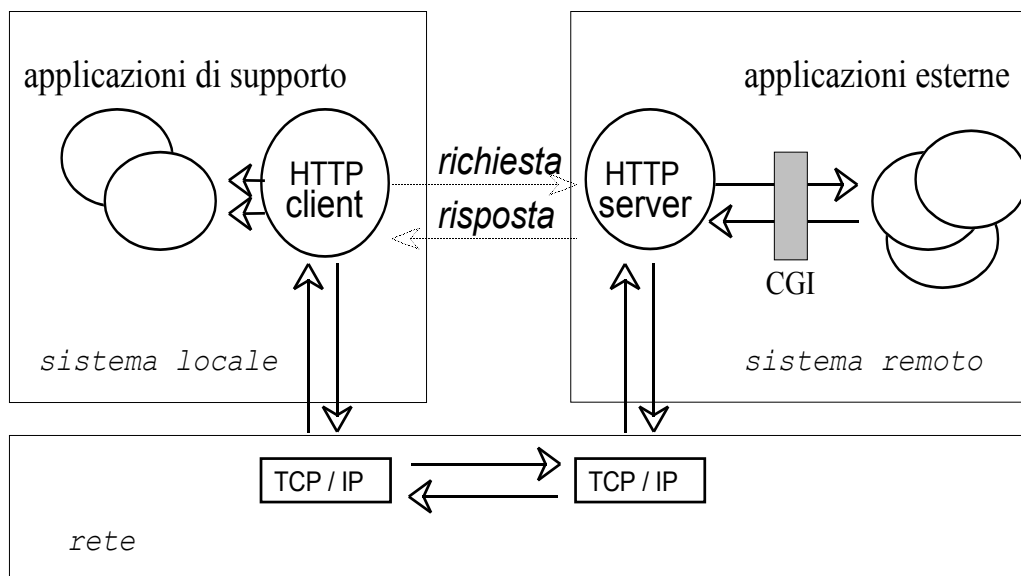
- Sistema di indirizzamento universale URI e URL
(Uniform Resource Identifier/Location)
- Protocollo HTTP (HyperText Transfer Protocol)
- Linguaggio HTML (HyperText Markup Language)
- Interfaccia CGI (Common Gateway Interface)

SISTEMA WWW

Cliente e sua interazione

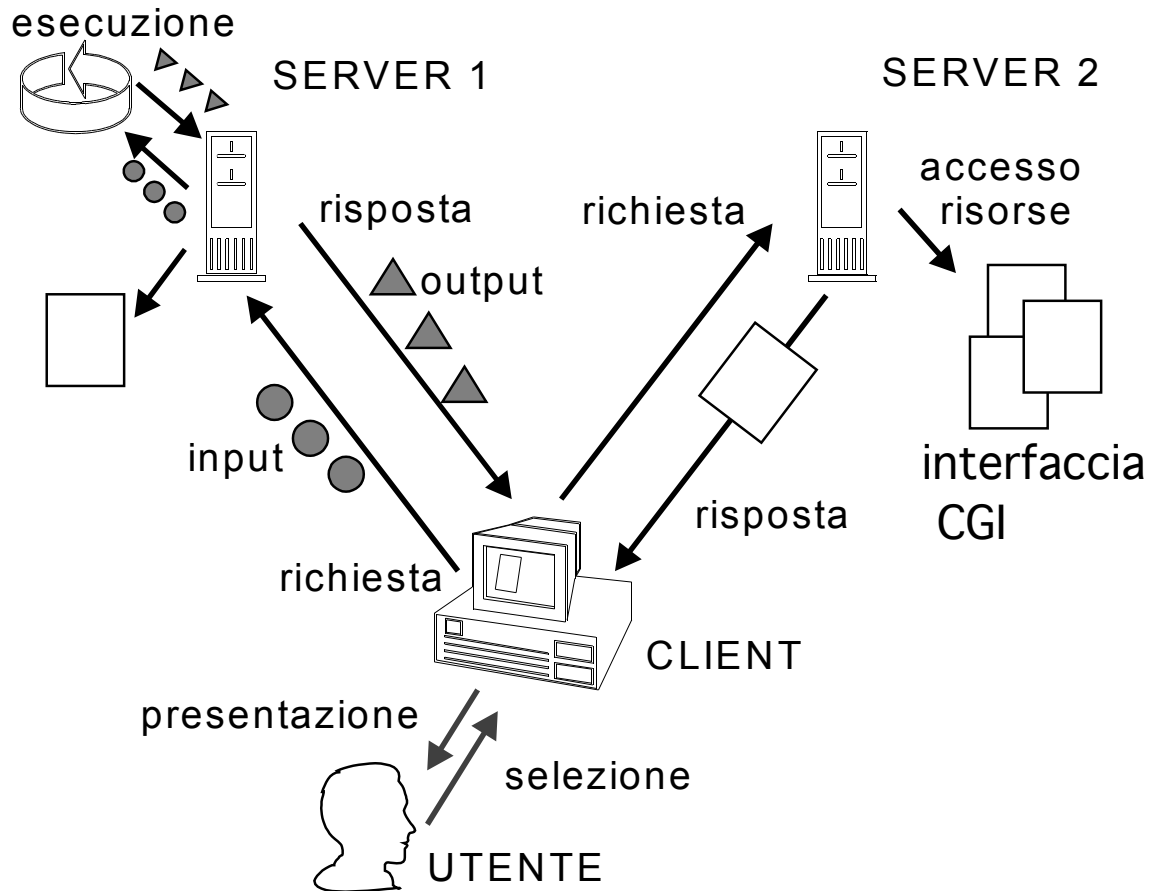


Il Cliente HTTP usa un modo cliente/servitore nei confronti di un server per volta e può anche interagire con risorse locali



Le interazioni cliente/servitore usano il protocollo TCP creando una connessione per ogni informazione da ritrovare (tipica porta fissa **80**)

Implementazione WWW



Modello di comunicazione

client / server

- parallelo
- stateless (ma CGI)

Funzionalità offerte

- Accesso ipertestuale a risorse informative
- Esecuzione applicazioni remote
 - ◊ invio di input da utente
 - ◊ presentazione output

URL Uniform Resource Locators

nomi unici per le risorse del sistema

specificati dal cliente per determinare il servitore

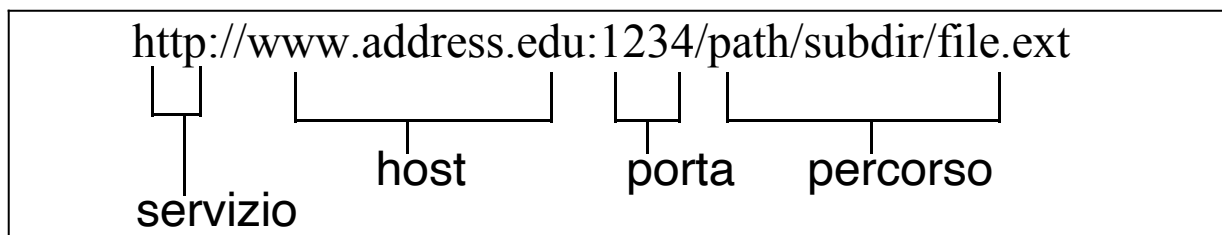
o **URN** (*Uniform Resource Names*) e servizio di name

o **Uniform Resource Locators (URL)**:

- nodo contenente la risorsa (*documento o dati*)
- protocollo di accesso alla risorsa (*e.g. http, gopher*)
- numero di porta TCP (*porta di default del servizio*)
- localizzazione della risorsa nel server.

`<protocollo>[://<host>][:<porta>][<percorso>]`

Sono riconosciuti i servizi internet e relativi protocolli =>
http, gopher, ftp, wais, telnet, news, nntp, e mail



Uso di default per localizzare risorse

Un **URL** può anche determinare un insieme di risorse:
ad esempio versioni multilingue tra cui scegliere

HTTP HyperText Transfer Protocol

protocollo di interfaccia tra cliente e servitore

Uso di TCP e di connessione (porta **80** default)

Caratteristiche HTTP:

- request/response
- one-shot connection
- stateless

Request/response: richiesta e ricezione di dati.

One-shot connection: la connessione TCP è mantenuta solo per il tempo necessario a trasmettere i dati

Stateless: non mantiene nessuna informazione tra una richiesta e la successiva

in genere:

- **richiesta** del cliente con **informazioni** per il **server**
- **risposta** con informazioni dal server

il cliente può determinare una forma di scelta (**negoziazione**) sulle informazioni ed i servizi

HTTP-message = Simple-Request	;HTTP/0.9
/ Simple-Response	
/ Full-Request	;HTTP/1.0
/ Full-Response	

NON c'e' stato del server

HTTP request/response

Formato del messaggio di **richiesta**:

indirizzo del server	metodo di richiesta	campi opzionali	path+ nome file (in generale, i dati)
----------------------	---------------------	-----------------	--

Metodo di richiesta:

- GET richiesta di leggere una pagina web
- HEAD richiesta di leggere l'header di una pagina web (es. contiene data ultima revisione del documento) (metodo usato nelle tecniche di caching nei client)
- PUT richiesta di pubblicare una pagina web
- POST append di dati (di solito HTML form)
- DELETE rimuove una pagina web

Campi opzionali:

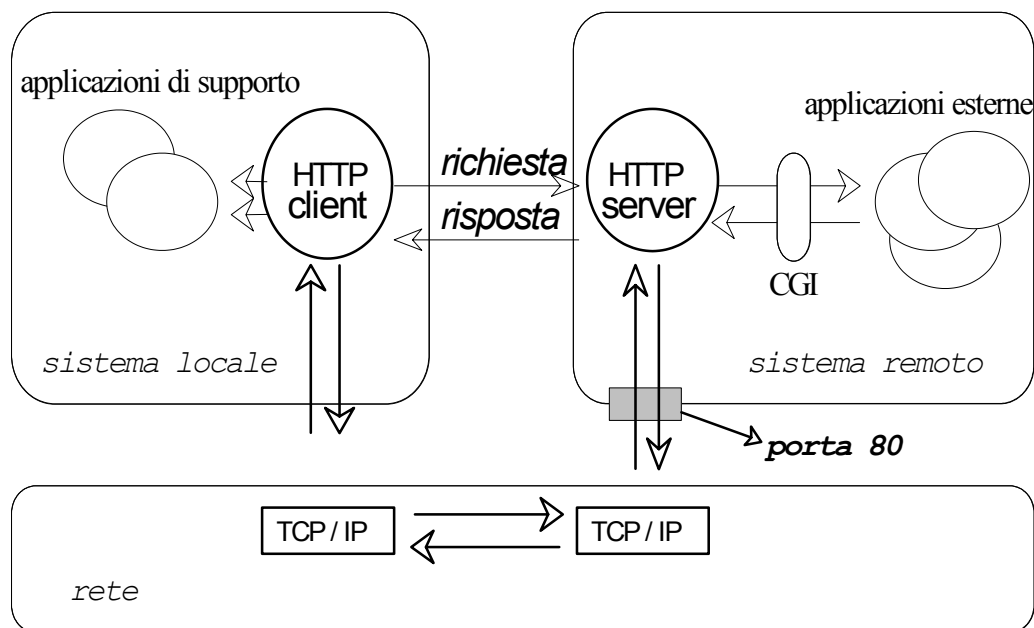
- from: identità dell'utente richiedente (debole forma di autenticazione degli accessi)
- referer: il documento contenente il link che ha portato al documento corrente

Formato del messaggio di **risposta**:

status code	informazioni sull'oggetto	dati
-------------	---------------------------	------

- **status code**: successo o fallimento (es. file not found)
- **informazioni sull'oggetto**: data di modifica, tipo di oggetto. Ogni tipo di oggetto (immagine, HTML, audio) attiva una gestione specifica da parte del client.

HTTP One-shot connection



1. Il browser (http client) riceve un URL dall'utente
2. Il browser richiede al DNS di risolvere il nome della macchina specificata nell'URL
3. Il DNS risponde con l'indirizzo IP
4. Il browser stabilisce una connessione TCP sulla porta 80 dell'indirizzo IP
5. Il browser manda una richiesta con il metodo:
GET nome_pagina_web
6. Il server risponde mandando la pagina web richiesta

Attenzione: la presenza in una pagina web di altri oggetti (immagini, applets, ecc.) costringe il client ad aprire una **differente connessione** per ognuno degli oggetti necessari.

Il cliente http può eseguire richieste seguendo protocolli differenti (es. ftp) $\Rightarrow$ porte differenti.

HTML HyperText Markup Language

HTML è un linguaggio di specifica delle informazioni che deriva da SGML (Standard Generalized Markup Language). E' un **markup language** (TeX, RTF).

I linguaggi markup usano dei **tag** definiti **funzionalmente** per caratterizzare il testo incluso.

tag HTML

testo di tipo header 1: `<H1>testo</H1>`

testo in grassetto: `<STRONG>testo</STRONG>` oppure
`<B>testo</B>`

Visualizzazione dipendente dal browser

link: `<A HREF = "destinazione"> descrizione </A>`

immagini: `<IMG SRC = "myimage.gif">`

applet Java:

`<APPLET CODE="Hello.class" WIDTH=100 HEIGHT=80>`

HTML molto semplice per non complicare il cliente

versione	browser	proprietà
1.0	storico	header, liste, enfasi
2.0	Mosaic	Inline Image, form
2.1	Navigator/Explorer	tabelle, allineamento
3.2	Navigator/Explorer	frame, ...
4.0	nessuno...	stili, ...

Esempio pagina HTML (codice)

```
<head> <title>Getting Started</title> </head>

<body>
<h1> Getting Started <img src=../images/Start.gif
height=40 width=40 align=top>
</h1>
<p>
<h3><em>by      Kathy      Walrath      and      Mary
Campione</em></h3>
<p>
The lessons in this trail show you the simplest
possible Java programs and tell you how to compile
and run them. They then go on to explain the
programs, giving you the background knowledge you
need to understand how they work.
<p>
.....

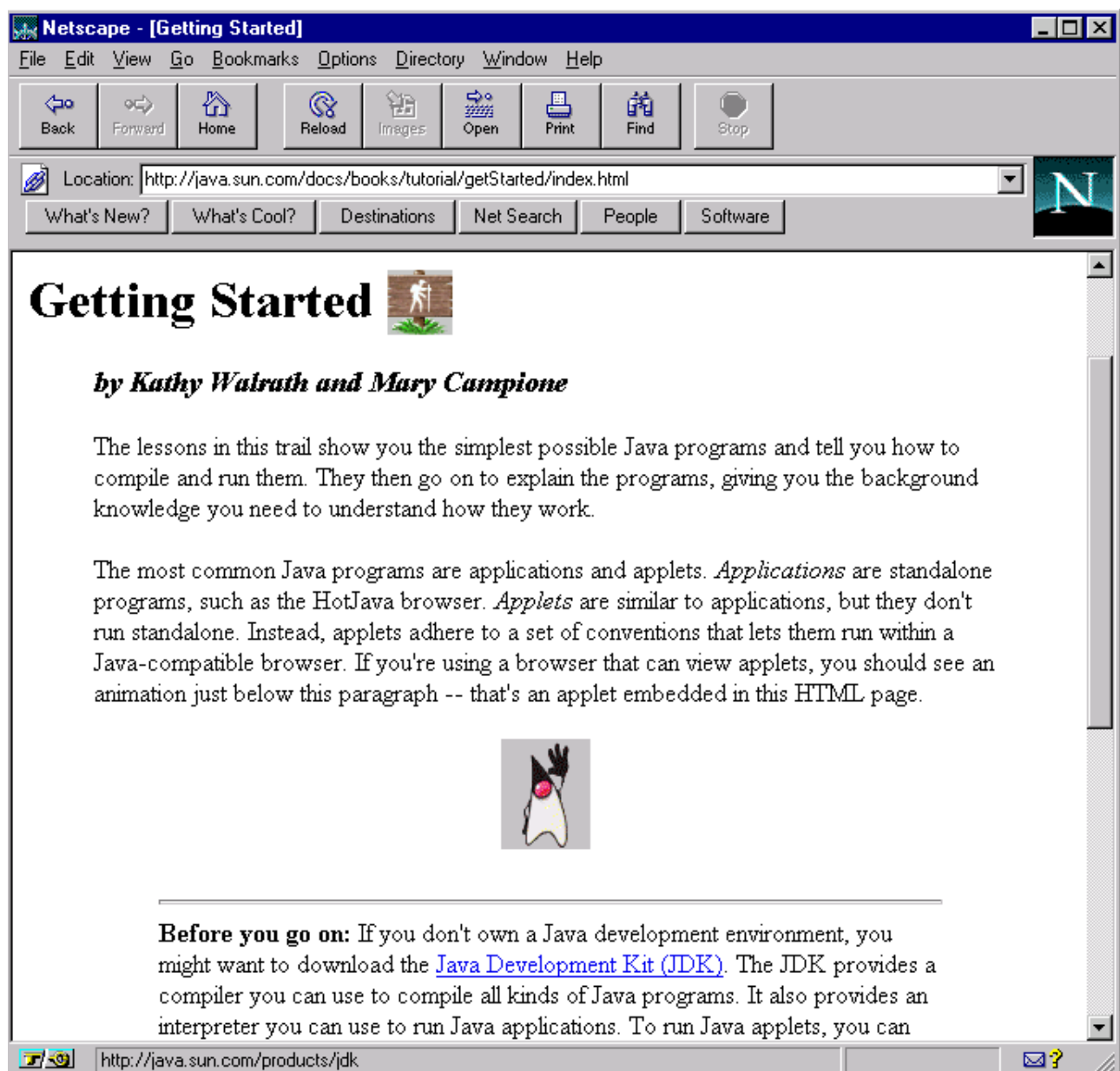
<p align=center>
<center>
<applet code=Animator.class codebase="../example"
width=55 height=68>
    <param name=endimage value=10>
    <param name=pauses value="2500|100">
</applet>
</center>
</p>

<hr>
<strong>Before you go on:</strong> If you don't
own a Java development environment, you might want
to      download      the      <a
href="http://java.sun.com/products/jdk">      Java
Development Kit (JDK)</a>. The JDK provides a
compiler you can use to compile all kinds of Java
programs. It also provides an interpreter you can
use to run Java applications. To run Java applets,
you can .....
```

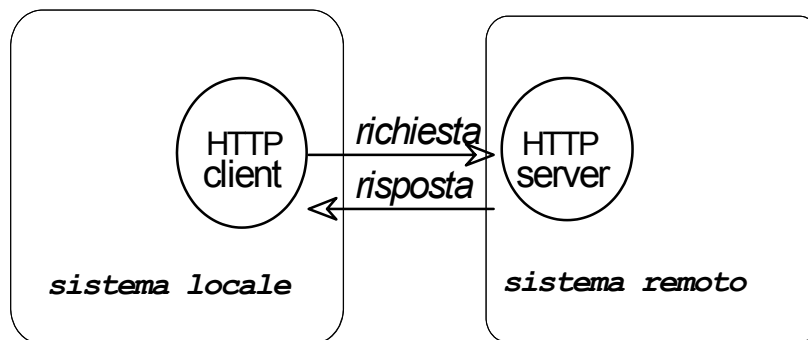
Esempio (visualizzazione)

pagina

HTML



Programmazione client/server in WWW

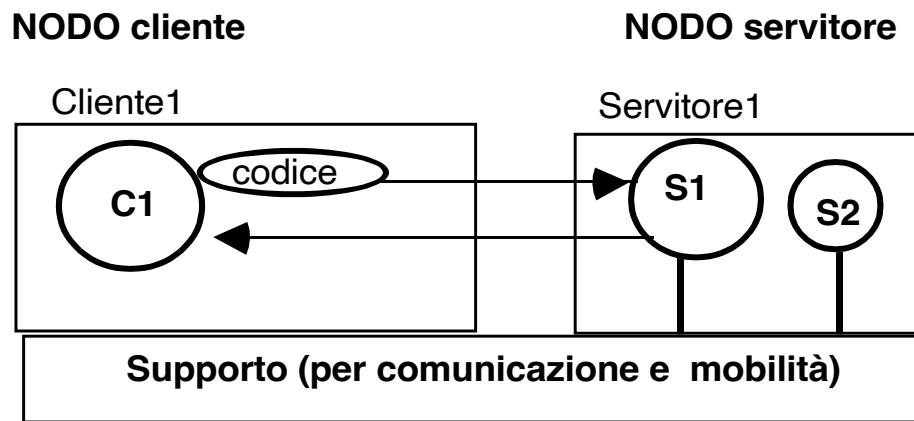


Possibilità di avere **risposta** con informazioni dinamiche

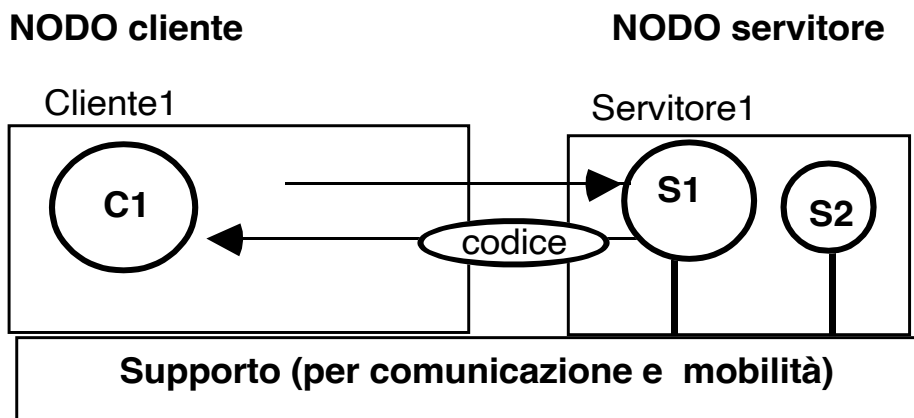
Che tipo di elaborazione delle informazioni e
dove viene eseguita

richiesta	risposta	tipo di elaborazione
Documento HTML	Statica (la pagina è un file, non modificabile)	semplice trasferimento file dal server
CGI	dinamica	qualunque elaborazione sul nodo server
Java applet	statica	codice dal server non modificabile, esecuzione sul client
Java applicazione	dinamica	server elabora dinamicamente il codice (in base alla richiesta), esecuzione dinamica sul client

Modelli di Programmazione e mobilità



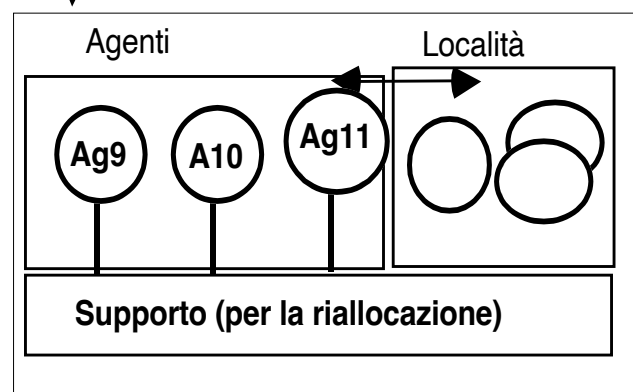
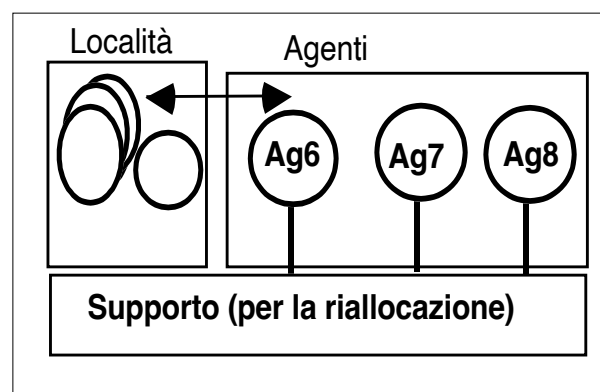
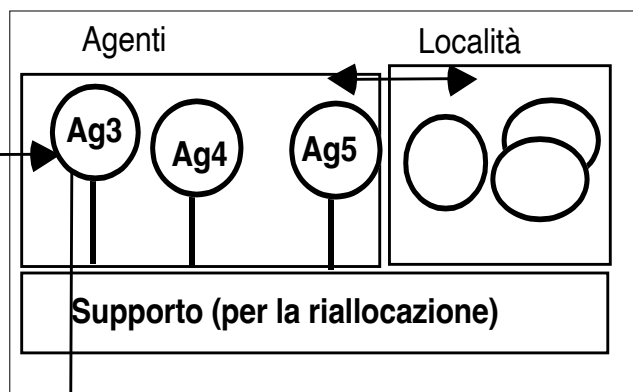
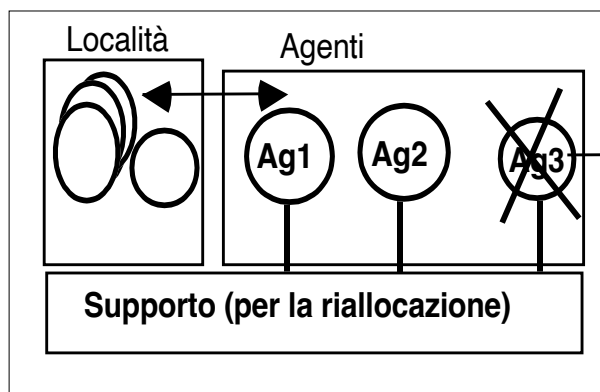
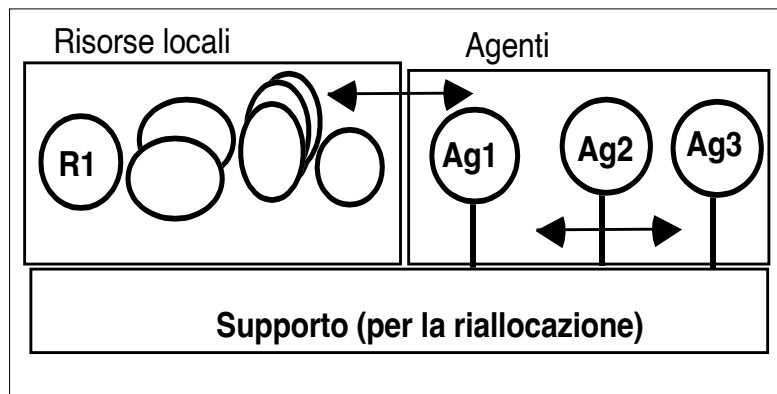
REMOTE EVALUATION



CODE ON DEMAND

Modelli di Programmazione e mobilità (Agenti Mobili)

NODO VIRTUALE di un sistema ad agenti



Modelli di Programmazione e mobilità

Dimensioni di classificazione del modello Client/Server:

- trasferimento del **codice**
- trasferimento dei **dati**
- trasferimento della **Execution Unit** (lo stato di esecuzione di un processo, senza ripartire dall'inizio)

modello	trasferimento codice	trasferimento dati	trasferimento EU
RPC	NO	SI'	NO
COD	SI'	NO	NO
REV	SI'	SI'	NO
MA	SI'	SI'	SI'

RPC: Remote Procedure Call

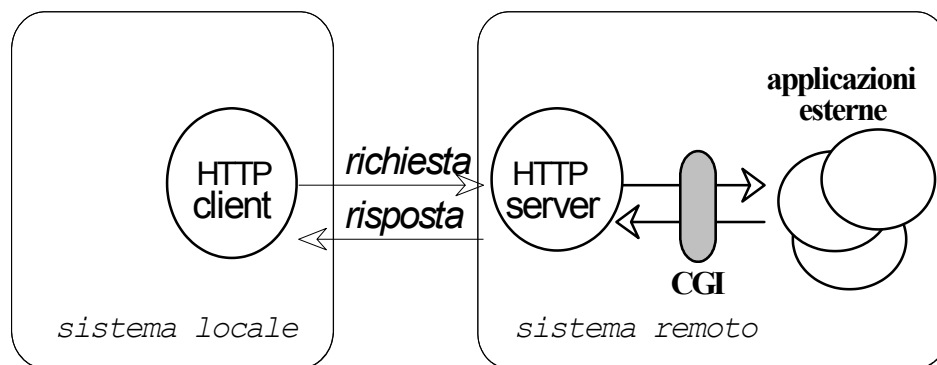
COD: Code On Demand (Applets java)

REV: Remote Evaluation (remote execution UNIX)

MA: Mobile Agents

Mobilità **strong** e mobilità **weak**

Common Gateway Interface (CGI)



CGI è uno **standard** per interfacciare un server WWW con applicazioni esterne (residenti sulla macchina server)

CGI fornisce all'utente la capacità di eseguire una applicazione sulla macchina server remota

Strumento tipico non interattivo
Collo di bottiglia

<i>richiesta</i>	<i>risposta</i>	<i>tipo di elaborazione</i>
CGI	dinamica	qualunque, sul nodo server

Programmazione CGI

Una applicazione CGI permette agli utenti di eseguire una applicazione sul nodo dove risiede il server **www**.

Applicazioni CGI possono essere scritte in:

- C/C++
- Fortran
- PERL
- TCL
- Any Unix shell
- Visual Basic
- AppleScript

normale attivazione di programma Unix, modello filtro con variabili di ambiente predefinite

Devono rispettare l'interfaccia CGI con il server

Interfaccia tra **server www** e applicazione **CGI**:

- variabili di ambiente
- linea di comando
- standard input
- standard output

Variabili d'ambiente

sono utilizzate dal server per dare informazioni di servizio all'applicazione CGI:

SERVER_SOFTWARE, nome e versione del server HTTP

SERVER_NAME, nome nodo server o suo indirizzo

GATEWAY_INTERFACE, versione interfaccia CGI cui il server aderisce

REQUEST_METHOD, metodo invocato nella richiesta

REMOTE_HOST, REMOTE_ADDR, AUTH_TYPE,

REMOTE_USER, CONTENT_TYPE,

CONTENT_LENGTH,

Linea di comando

per richieste di tipo ISINDEX, per ricerche di testo nei documenti. Le parole da ricercare sono inserite dal server sulla linea di comando della applicazione CGI. (compatibilità)

Standard input

il server ridirige sull'ingresso della applicazione CGI i dati ricevuti dal client (browser). Il numero di byte è nella variabile d'ambiente CONTENT_LENGTH, il tipo dei dati MIME nella CONTENT_TYPE.

Standard output

l'applicazione CGI manda il risultato dell'elaborazione sullo standard output verso il server, che a sua volta prepara i dati e li spedisce al client.

Client HTTP → server HTTP → CGI

Tipicamente, uso di **form**

```
<TITLE>Esempio di Form </TITLE>
<H1>Esempio di Form </H1>
```

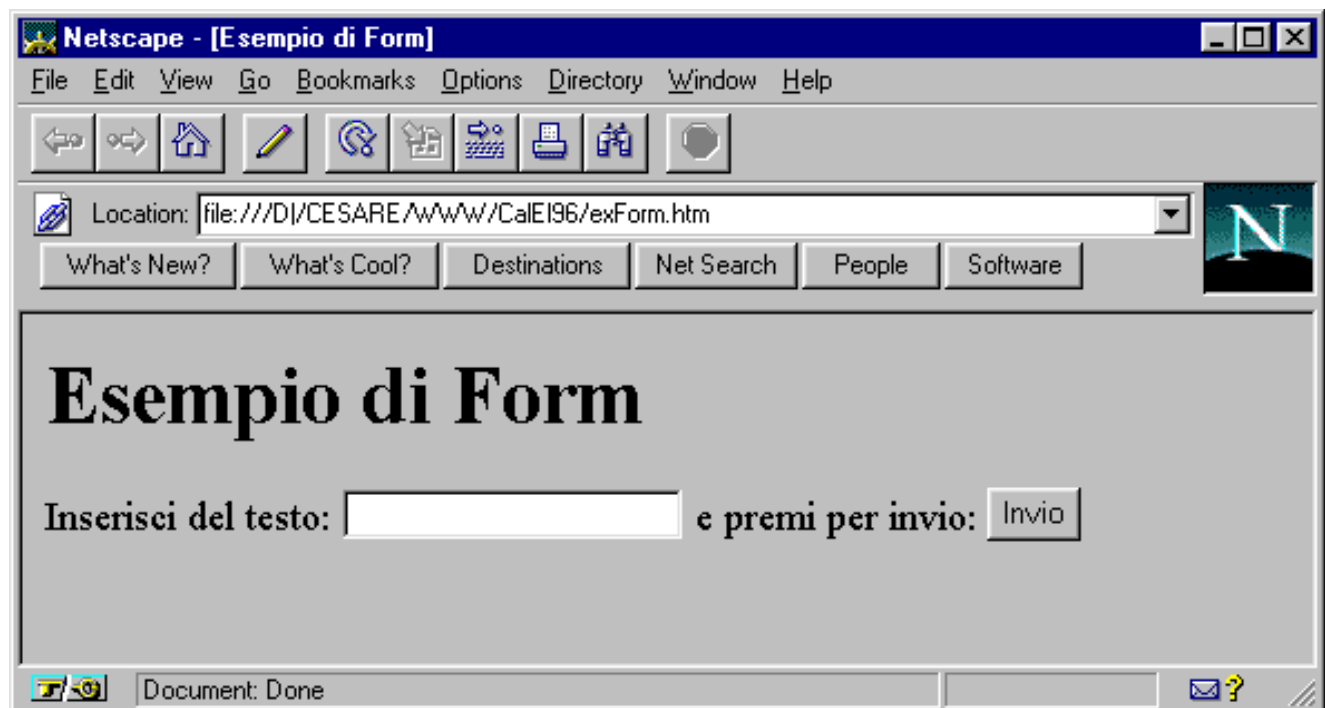
```
<FORM METHOD="POST" ACTION="http://www-
lia.deis.unibo.it/cgi-bin/post-query">
```

Inserisci del testo: <INPUT NAME="entry">

e premi per invio: <INPUT TYPE="submit"
VALUE="Invio">

```
</FORM>
```

Visualizzazione **form**



Client HTTP → server HTTP → CGI

Attributi del **form** tag

```
<TITLE>Esempio di Form </TITLE>
```

```
<H1>Esempio di Form </H1>
```

```
<FORM METHOD="POST" ACTION=  
"http://www-lia.deis.unibo.it/cgi-bin/post-query">
```

```
Inserisci del testo: <INPUT NAME="entry">
```

```
e premi per invio: <INPUT TYPE="submit"  
VALUE="Invio">
```

```
</FORM>
```

Dove:

ACTION URL di chi processa la query

METHOD metodo usato per sottomettere il form:

POST il form con i dati è spedito come data body
(metodo consigliato)

GET il form con i dati è spedito attaccato
all'URL (action?
name=value&name=value)

caso GET

```
http://www-lia.deis.unibo.it  
/cgi-bin/get-query?entry=testo
```

caso POST

```
http://www-lia.deis.unibo.it  
/cgi-bin/post-query
```

e come data body:

```
entry=testo
```

Applicazione CGI → server HTTP

Applicazione CGI usa **standard output** per mandare al server i dati. I dati sono identificati da un header.

Tipi di dati forniti:

- full document con il corrispondente MIME type (text/html, text/plain per testo ASCII, etc.)

Esempio: per spedire una pagina HTML

Content-type: text/html

```
<HTML><HEAD>
<TITLE>output di HTML da script CGI</TITLE>
</HEAD><BODY>
<H1>titolo</H1>
semplice <STRONG>prova</STRONG>
</BODY></HTML>
```

- reference a un altro documento

Esempio: riferisco un documento gopher

Content-type: text/html

Location: gopher://httprules.foobar.org/0

```
<HTML><HEAD>
<TITLE > Sorry ... it moved </TITLE>
</HEAD><BODY>
<H1>go to gopher </H1>
Now available at
<A HREF="gopher://httprules.foobar.org/0">
    new location</A> of gopher server.
</BODY></HTML>
```

Applicazione CGI

Esempio: generazione della pagina di risposta
(caso full document)

```
#include <stdio.h>
.....

main(int argc, char *argv[]) {
    int cl;

    generazione di un full document in risposta
    printf("Content-type: text/html");

    cl = atoi(getenv("CONTENT_LENGTH"));

    for(x=0; cl && (!feof(stdin)); x++) {
        ...
        elaborazione dell'input (stdin)
        ...
    }

    printf("<H1>Query Results</H1>");
    printf("You submitted ...");

    for(x=0; x <= m; x++)
        printf(".....", ... , ....);
}
```

INTERNET: PRINCIPALI PROBLEMI

Problema del **traffico**

alcuni formati di informazioni possono richiedere un
eccesso di banda
necessità di adeguare le infrastrutture

Problema della **sicurezza**

la trasparenza richiede la integrazione con i sistemi
locali di esecuzione/visualizzazione: intrusioni o virus da
controllare
riservatezza delle transazioni e autenticazione clienti

Problema della **visualizzazione**

alcuni formati richiedono una lunga visualizzazione:
tempi di accesso molto rallentati

sfruttare la possibile concorrenza/asincronismo

Uso di clienti con strategie adatte a

abbreviare il **tempo di risposta**
favorire il browsing delle informazioni
**(web navigation e ricerche mediante robot,
spider, worm, ecc.)**

evoluzione dei protocolli

per garantire una apertura alle esigenze man mano
determinate

La ricerca di informazioni sul WEB

Il web è un ipertesto (un grafo) con ormai milioni di nodi → problema di reperire le informazioni

Esistono **indici del web** (detti anche cataloghi o directories), realizzati per facilitare la ricerca di informazioni su Internet.

Organizzazione indici:

- alfabetica
- per argomento (gerarchici)
- per area geografica (gerarchici)
- con possibilità di ricerca (parole chiave)

Gli indici sono costruiti utilizzando dei programmi che esplorano i site web presenti in rete.

Programmi più diffusi:

- search engine
- spider
- crawler
- worm
- knowbots (knowledge robots)

Esempio:

AltaVista (<http://altavista.digital.com>) Web index:

- 30 milioni pagine da 275,600 servers
- 4 milioni articoli da 14,000 Usenet news groups.

E' acceduto più di 21 milioni di volte al giorno
(dati di Ottobre 96)

I motori di ricerca

Struttura dati tipica:

Due fasi: **ricerca** e **indicizzazione**

Passi della **ricerca** (algoritmi tipo breadth-first, depth-first):

- prelevare un URL
- eseguire hash URL
- Se hash URL presente in Tabella hash allora STOP altrimenti
 - aggiungere hash URL in Tabella hash
 - aggiungere Puntatori a URL e a pagina in Tabella URL
 - aggiungere URL e Pagina (o titolo) in Heap
 - ripetere tutti i passi per ogni link presente nella pagina

Problemi:

- dimensioni del grafo web
- punto di partenza della ricerca
- tipo di ricerca: depth-first → stack overflow
breadth-first → dimensioni heap
- come trattare i link presenti nelle active map (CGI)
- URL obsoleti
- Macchine non raggiungibili rallentano la ricerca

I motori di ricerca

Fase di **indicizzazione**

La procedura di indexing estrae le parole chiave da ogni pagina (o titolo) web memorizzati nell'heap nella fase di ricerca (sintesi delle pagine)

Per trovare le parole chiave:

- si scartano le parole poco significative (articoli, etc.)
- si scelgono parole che nella pagina hanno la frequenza maggiore (es. Lycos)

Per ogni parola ottenuta si memorizza in una tabella la parola e l'URL che la contiene.

Alla fine della indicizzazione si ordina la tabella sulle parole e si salva su file che verrà consultato per le ricerche da parte degli utenti

problemi:

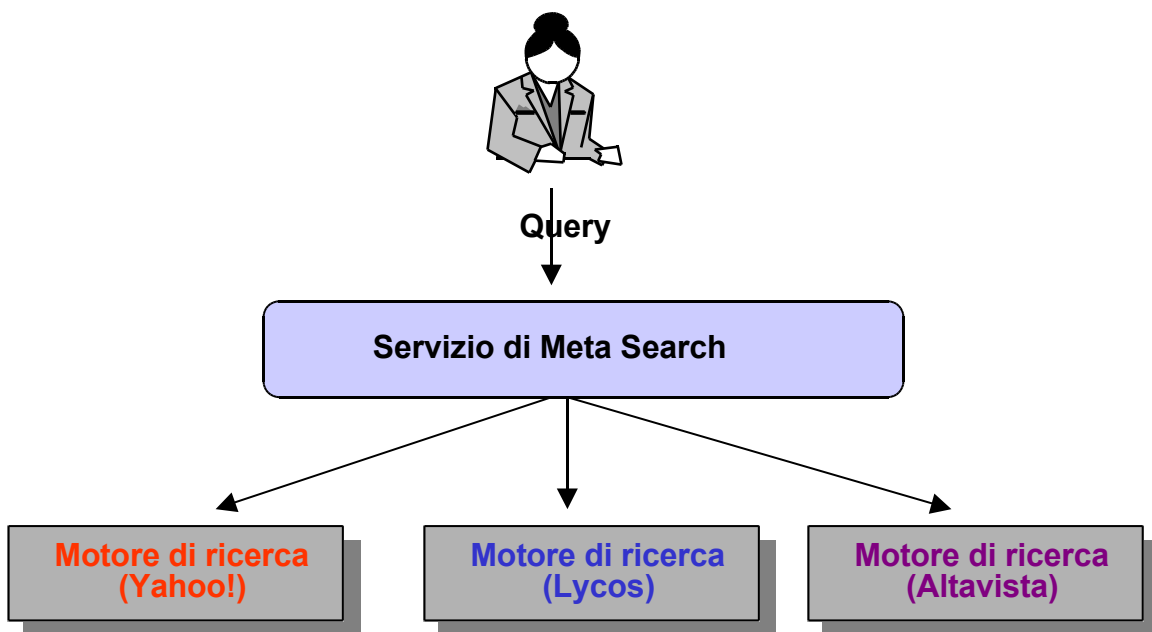
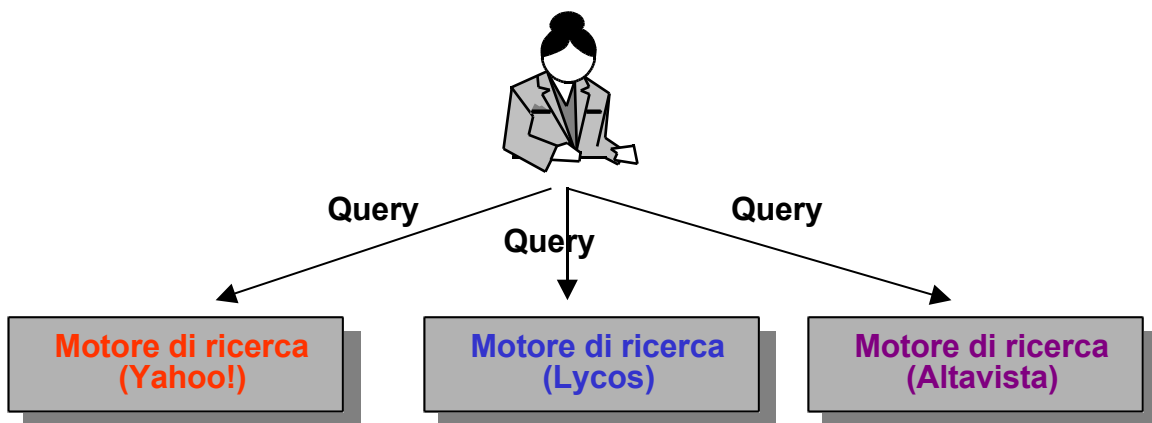
- titoli pagine spesso poco significativi
- analisi intere pagine costosa
- pagine solo video o audio, oppure active map

Ricerche e indicizzazioni cooperative: Harvest è un motore di ricerca che richiede a tutti i server www di eseguire una applicazione per indicizzare localmente la macchina. Un motore centrale raccoglie tutti i risultati
Problema della sicurezza

I meta-searcher

Effettuano la ricerca delle informazioni su più indici del web contemporaneamente.

Un meta-searcher invia la stessa query contemporaneamente a più indici del web, NON contiene un database



Evoluzioni HTTP

Necessità di **evoluzione** ==> (HTTP 1.1)

- stato associato al cliente (cookies)
- stato nel servitore
- uso di **una** connessione per più richieste

Anche apertura da parte del cliente di più connessioni contemporaneamente

Estensione dei metodi di richiesta:

- operazioni tipo **FTP**

Evoluzione verso un passaggio di informazioni in unico **stream** con **ritrovamento asincrono** dalla parte del cliente

Limiti della **banda**

- uso di caching su servitori intermedi
- cache configurate come agenti con capacità di aggiornare reciprocamente informazioni

Non adeguatezza degli **strumenti**

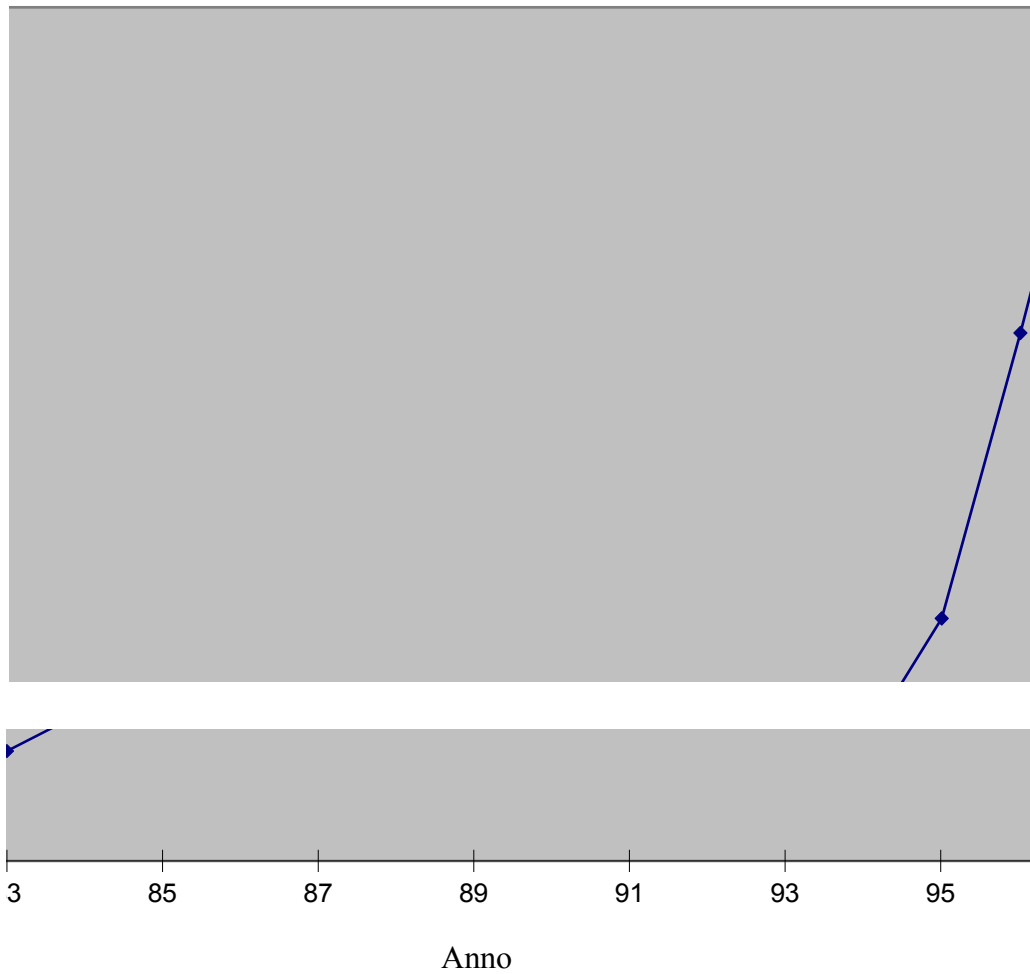
- capacità di integrare tool per scrivere e usare informazioni multimediali
- uso di protocolli Real-Time

Modifica dei **modelli**

- superare i limiti di interazione, interfacce user-friendly non appena bande più ampie disponibili

Processo di standardizzazione diffuso e veloce

La crescita di Internet



numero di **macchine** collegate ad Internet
(stima utenti = No. Host x 10)

La crescita di Internet

numero di **domini** Internet

<NIC.MERIT.EDU> /nsfnet/statistics/history.hosts

Internet Connectivity

