

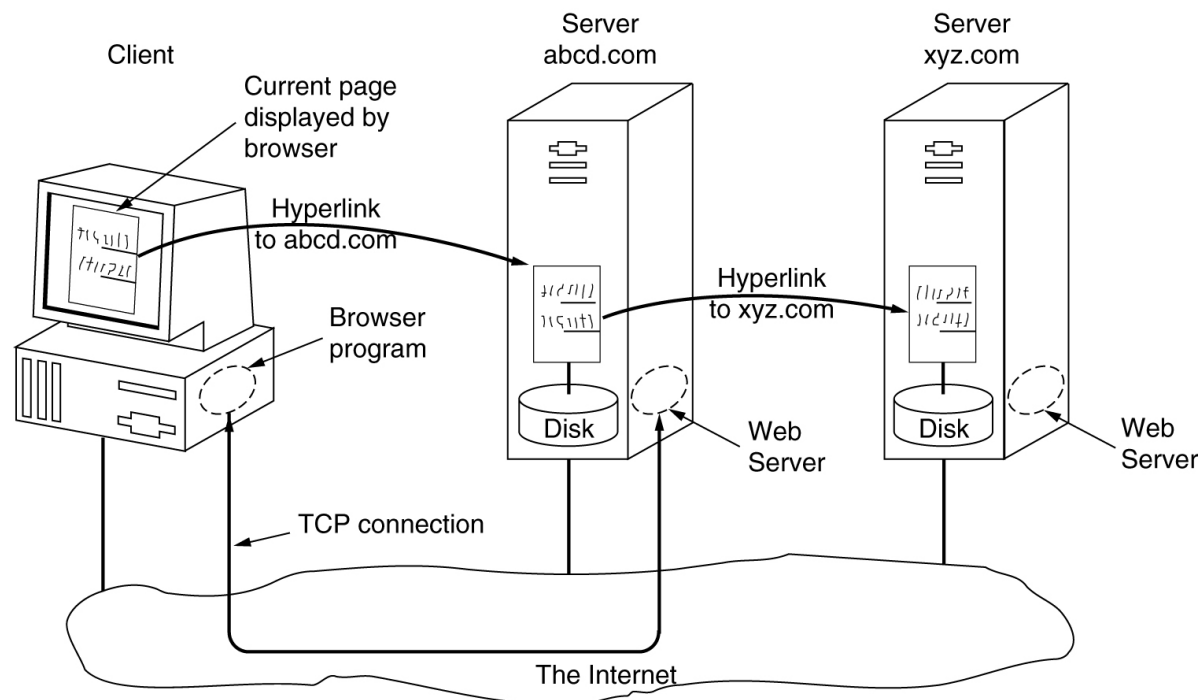


Nivelul Aplicație

World Wide Web

World Wide Web

- Set de documente (pagini) cu legături între ele (hyperlinks)
- Distribuite pe mașini diferite
- Include o pagina de referință (home page)

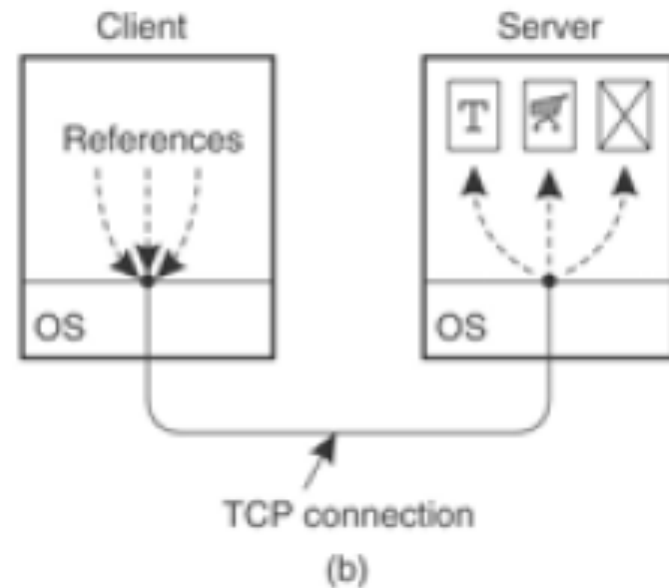
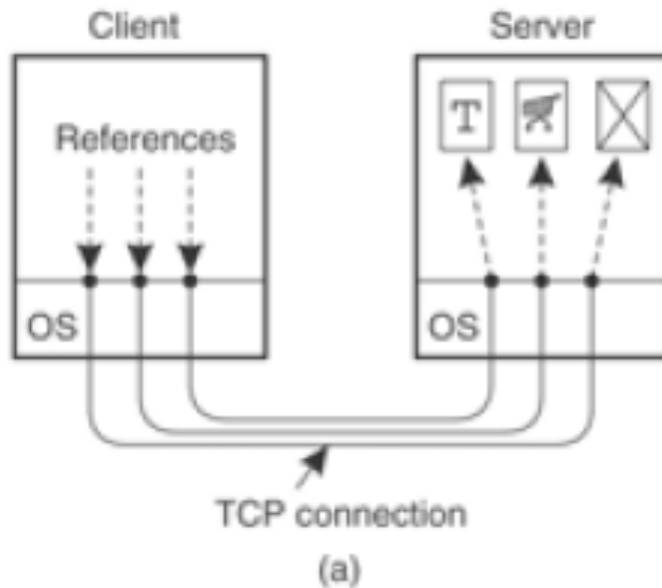


Interacțiunea client - server

- Browser - determina URL
- Browser - cere DNS-ului adresa IP pentru `www.w3.org`
- DNS - raspunde cu `18.23.0.23`
- Browser - deschide o conexiune TCP la port 80 pe `18.23.0.23`
- Browser - trimite o comanda
`GET /hypertext/www/TheProject.html`
- Server `www.w3.org` - trimite fisierul `TheProject.html`
- Conexiunea TCP este inchisa
- Browser - afișează conținutul din `TheProject.html`
- Browser - extrage si afiseaza toate imaginile din `TheProject.html` (se deschide o noua conexiune TCP pentru fiecare imagine)

Conexiuni persistente

- Disponibile in HTTP 1.1
- O singura conexiune persistenta poate fi folosita pentru mai multe cereri-raspunsuri
- Cererile pot fi transmise si in pipeline (fara a astepta raspunsurile)





Trei elemente de baza

- O schema de adresare a documentelor in Internet (**URL – Uniform Resource Locator**)
- Un limbaj de formatare a documentelor (**HTML – HyperText Markup Language**)
- Un protocol pentru transportul mesajelor specializate prin retea (**HTTP – HyperText Transfer Protocol**)

URL – Uniform Resource Locator

scheme://host[:port#]/path/.../[/;url-params][?query-string][#anchor]

scheme protocol (http, ftp etc.)

host nume / adresa IP a serverului Web

port# numar port server Web (80 pentru http)

path calea de la radacina serverului la document

url-params pentru identificarea sesiunii

query-string valori din formular HTML

anchor referinta la un marcaj pozitional din document

exemplu

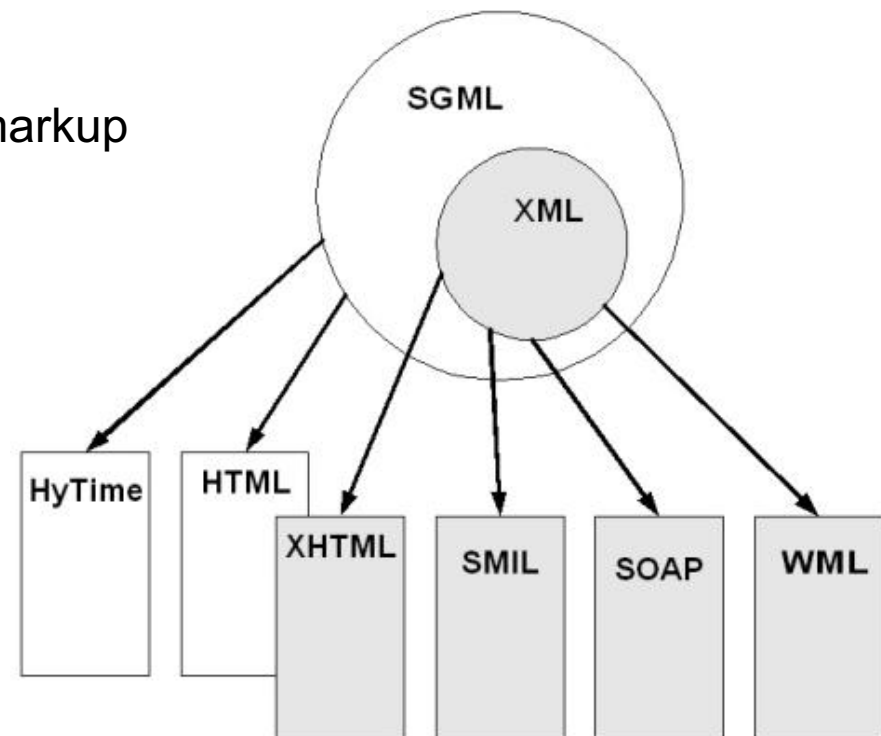
<http://www.situlmeu.ro/cv/test?id=8079?name=valentin&x=true#aici>

Câteva URL-uri obișnuite

Schema	Utilizat pentru	Exemple
http	Hipertext (HTML)	http://www.cs.vu.nl/~ast
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
File	Fișier local	file:///usr/suzanne/prog.c
news	Grup de știri	news:AA0134223112@cs.utah.edu
news	Articol de știri	news:AA0134223112@cs.utah.edu
gopher	Gopher	gopher://gopher.tc.umn.edu/11/libraries
mailto	Trimitere de poșta electronică	mailto:JohnUser@acm.org
telnet	Conectare la distanță	telnet://www.w3.org:80

HTML - HyperText Markup Language

- Definit ca o aplicatie SGML – Standard Generalized Markup Language
- Aplicatia are patru parti
 - Declaratia SGML – caractere si delimitatori
 - DTD (Document Type Definition) – constructiile de marcare valide (sintaxa)
 - Specificarea semanticii asociate
 - Instante de documente cu continut si markup



Structura unei pagini

```
<html>
  <head>
    <title>
      Prima incercare
    </title>
  </head>
  <body>
    Prima incercare

    Nu este greu sa construiesti un
    text urat in html, mai complicat este
    sa construiesti un text care sa arate
    bine.
  </body>
</html>
```

Prima incercare: Nu este greu sa construiesti un text urat in html, mai complicat este sa construiesti unul care sa arate bine.

Ce afiseaza browser-ul

O selecție de marcaje uzuale

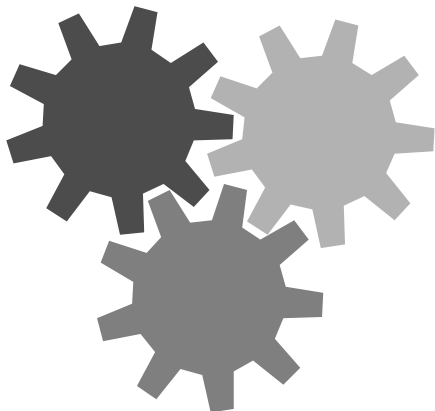
Marcaj	Descriere
<code><html> ... </html></code>	Delimitează textul scris în HTML
<code><head> ... </head></code>	Delimitează zona de antet
<code><title> ... </title></code>	Definește titlul (nu este afișat de programul de navigare)
<code><body> ... </body></code>	Delimitează zona de corp
<code><hn> ... </hn></code>	Delimitează un titlu de nivel n
<code> ... </code>	Text îngroșat
<code><i> ...</i></code>	Text cursiv
<code><center> ... </center></code>	Centrat pe orizontală
<code>
</code>	Trecere la linie nouă
<code><p></code>	Început de paragraf
<code> ... </code>	Delimitează o listă neordonată
<code> ... </code>	Delimitează o listă ordonată (numerotată)
<code> ... </code>	Delimitează un elemente într-o listă ordonată sau neordonată
<code><hr></code>	Linie orizontală
<code></code>	Afișează o imagine în acest loc (sau text-ul specificat)
<code>text</code>	Hiper-legătură la o pagină
<code>text</code>	Declară o ancoră într-un document

HTML – un exemplu

```
<html>
<head> <title> AMALGAMATED WIDGET, INC. </title></head>
<body> <h1> Welcome to AWI's Home Page </h1>
<img SRC="http://www.widget.com/images/logo.gif" ALT="AWI Logo"> <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's</b>
home page. We hope <i> you </i> will find all the information you need here.
<p> Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by FAX. </p>
<hr>
<h2> Product Information </h2>
<ul>
  <li> <a href="http://widget.com/products/big" > Big widgets </a>
  <li> <a href="http://widget.com/products/little" > Little widgets </a>
</ul>
<h2> Telephone Numbers </h2>
<ul>
<li> 1-800-WIDGETS,
<li> 1-415-765-4321
</ul>
</body>
</html>
```

Pagina formatată

Welcome to AWI's Home Page



We are so happy that you have chosen to visit **Amalgamated Widget's** home page. We hope *you* will find all the information you need here.

Below we have links to information about our many fine products. You can order electronically (by WWW), by telephone, or by FAX.

Product Information

- [Big widgets](#)
- [Little widgets](#)

Telephone numbers

- 1-800-WIDGETS
- 1-415-765-4321

Formulare – marcaje specifice

element HTML	Parametri	Semnificație
<INPUT>, TYPE=text	NAME, SIZE, MAXLENGTH	câmp de intrare (implicit)
<INPUT>, TYPE=radio	NAME, VALUE	buton radio
<INPUT>, TYPE=checkbox	NAME, CHECKED	casetă de selecție
<INPUT>, TYPE=password	NAME, SIZE, MAXLENGTH	câmp de parolă
<INPUT>, TYPE=reset sau submit		buton de acțiune
<INPUT>, TYPE=image	NAME, ALIGN, SRC	hartă (image) activă
<INPUT>, TYPE=hidden	NAME,	element ascuns
<SELECT>	NAME, OPTION, MULTIPLE	listă de selecție
<TEXTAREA>	NAME, COLS, ROWS, WRAP	zonă de editare

Formulare – un exemplu

```
<html>
<head><title> AWI CUSTOMER ORDERING FORM </title></head>
<body>
<h1> Widget Order Form </h1>
<form ACTION="http://widget.com/cgi-bin/widgetorder" method=POST>
  <p> Name <input name="customer" size=46> </p>
  <p> Street Address <input name="address" size=40> </p>
  <p> City <input name="city" size=20> State <input name="state" size=4> Country
      <input name="country" size=10> </p>
  <p> Credit card # <input name="cardno" size=10> expires <input name="expires"
      size=4> M/C <input name="cc" type=radio value="mastercard"> VISA <input
      name="cc" type=radio value="visacard"> </p>
  <p> Widget size Big <input name="product" type=radio value="expensive"> Little
      <input name="product" type=radio value="cheap"> Ship by express courier
      <input name="express" type=checkbox> </p>
  <p> <input type=submit value="submit order"> </p>
</form>
</body>
</html>
```

Thank you for ordering an AWI widget, the best widget money can buy!

Widget Order Form

Name

Street address

City State Country

Credit card # Expires M/C ☐ Visa ☐

Widget size Big ☐ Little ☐ Ship by express courier ☐

Thank you for ordering an AWI widget, the best widget money can buy!

Formulare

Un răspuns posibil cu informațiile completate de utilizator

Widget Order Form

Name

Street address

City State Country

Credit card # Expires M/C ☒ Visa ☐

Widget size Big ☐ Little ☒ Ship by express courier ☒

Thank you for ordering an AWI widget, the best widget money can buy!

(împărțit aici în trei linii din motive de aliniere in pagină)

HTTP

- Protocol “**stateless**”
- Foloseste paradigma **request/response**
 - clientul si serverul comunica direct sau prin proxy-uri
 - structura mesajelor:
 - linia de comanda / raspuns
 - linii de antet
 - linie blank
 - corp mesaj

Structura mesaj **request**

```

METHOD /path-to-resource
      HTTP/version-number
Header-name-1: value
Header-name-2: value
...
[ optional request body ]
  
```

Exemplu

```

GET /sj/index.html HTTP/1.1
Host: www.mywebsite.com
  
```

Structura mesaj **response**

```

HTTP/version-number status-code message
Header-name-1: value
Header-name-2: value
...
[ response body ]
  
```

Exemplu

```

HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 9934
...
<HTML>
<HEAD> ... </HEAD> ...
... </HTML>
  
```


Metode HTTP

Metoda	Descriere
GET	Cerere de citire a unei pagini Web
HEAD	Cerere de citire a antetului unei pagini de Web
POST	Adăugarea la resursa specificată (de exemplu o pagină de Web)
PUT	Cerere de memorare a unei pagini de Web
DELETE	Ștergerea unei pagini de Web
TRACE	Transmite in ecou cererea care a sosit
OPTIONS	Interogarea anumitor opțiuni
CONNECT	Rezervat pentru o utilizare viitoare

Exemplu GET

- Formular HTML

```
<HTML>
<HEAD><TITLE>Formular simplu</TITLE></HEAD>
<BODY>
<H2>Formular simplu</H2>
<FORM ACTION="http://finance.yahoo.com/q" METHOD="get">
Ticker: <INPUT SIZE="25" NAME="s">
<INPUT TYPE="submit" VALUE="Get Quote">
</FORM>
</BODY>
</HTML>
```

Formular simplu

Ticker:

- URL construit de browser pentru intrarea **YHOO**

`http://financiar.yahoo.com/q?s=YHOO`

- Cerere HTTP

`GET /q?s=YHOO HTTP/1.1`

`Host: financiar.yahoo.com`

`User-Agent: Mozilla/4.75 [en] (WinNT; U)`

Raspuns

HTTP/1.1 200 OK

Date: Sat. 03 May 2005 17:48:35 GMT

Connection: close

Content-Type: text/html

Set-Cookie: B=dfaosiu534qjnfretk&b=2;expires=Thu, 15
Aug 2011 20:00:00 GMT; path=/; domain=.yahoo.com

<HTML>

<HEAD><TITLE>Yahoo! financiar - YHOO</TITLE></HEAD>

<BODY>

...

</BODY>

</HTML>



Exemplu POST

- Aceeasi cerere, formulata cu metoda POST

```
POST /q HTTP/1.1
```

```
Host: financiar.yahoo.com
```

```
User-Agent: Mozilla/4.75 [en] (WinNT; U)
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: 6
```

```
s=YHOO
```

- Raspunsul este identic

Exemplu HEAD

Cerere

HEAD http://www.cs.pub.ro/~ionescu/ HTTP/1.1

Host: www.cs.pub.ro

User-Agent: Mozilla/4.75 [en] (WinNT; U)

Raspuns

HTTP/1.1 200 OK

Date: Mon, 05 Feb 2005 04:33:19 GMT

Server: Apache/1.2.5

Last-Modified: Mon, 05 Feb 2005 04:30:19 GMT

Content-Length: 2234

Content-Type: text/html

Coduri de stare

Cod	Semnificație	Exemple
1xx	Informație	100 = serverul acceptă continuarea tratării cererii de la client (asociat cu un antet Expect din cerere)
2xx	Succes	200 = cerere reușită; 204 = nu există conținut
3xx	Redirectare	301 = pagină mutată definitiv; 302 = pagina mutata temporar; 304 = pagina din memoria ascunsă este încă validă
4xx	Eroare la client	400 = cerere incorecta; 401 = ne-autorizat 403 = interzis 404 = pagina nu a fost găsită
5xx	Eroare la server	500 = eroare internă la server; 501 = ne-implementat 503 = încearcă mai târziu

Antete Mesaje HTTP

Antet	Tip	Descriere
User-Agent	Cerere	Informație asupra programului de navigare și a platformei
Accept	Cerere	Tipul de pagini pe care clientul le poate trata
Accept-Charset	Cerere	Seturile de caractere care sunt acceptabile la client
Accept-Encoding	Cerere	Codificările de pagini pe care clientul le poate trata
Accept-Language	Cerere	Limbajele naturale pe care clientul le poate trata
Host	Cerere	Numele DNS al serverului (folosit pentru virtual hosting)
Authorization	Cerere	O listă a drepturilor clientului
Cookie	Cerere	Trimite (la server) un cookie setat anterior
Set-Cookie	Răspuns	Serverul vrea să salveze un cookie la client
Server	Răspuns	Informație despre server (ex. Server: Apache/1.2.5)
Content-Encoding	Răspuns	Cum este codat conținutului (de exemplu, gzip)
Content-Length	Răspuns	Lungimea paginii în octeți
Content-Type	Răspuns	Tipul/subtipul MIME al paginii
Last-Modified	Răspuns	Ora și data la care pagina a fost ultima dată modificată
Location	Răspuns	O indicație pentru client pentru redirectarea cererii
Accept-Ranges	Răspuns	Serverul va accepta cereri în anumite limite de octeți
Date	Ambele	Data și ora la care mesajul a fost trimis
Connection	Ambele	Intenția de a păstra sau nu conexiunea (ex. Connection: Close)

Antete referitoare la tipul continutului

- Sistem de tipuri imprumutat din MIME (Multipurpose Internet Mail Extensions)
- Doua niveluri (reprezentate de doua antete in raspuns)

– Content-Encoding

- gzip (GNU zip)
- compress (UNIX)
- deflate (zlib format definit in RFC 1950 si 1951)

– Content-Type

- Tip, subtip si (optional) perechi *atribut = valoare*
- *Exemple*

`Content-Type: text/plain; charset = 'us-ascii'`

`Content-Type: text/xml`

`Content-Type: application/pdf`

`Content-Type: video/x-mpeg`

Exemplu mesaje multipart

Cerere

```
GET /cgi-bin/doit.cgi HTTP/1.1
Host: cgi-bin.netscape.com
Date: Sun, 18 Feb 2004 06:22:33 GMT
```

Raspuns

```
HTTP/1.1 200 OK
Server: Netscape-Enterprise-3.6 SP1
Date: Sun, 18 Feb 2004 06:22:35 GMT
Content-Type: multipart/x-mixed-replace; boundary="ThisRandomString"
```

```
--ThisRandomString
```

```
Content-Type: image/gif
```

```
...
```

```
--ThisRandomString
```

```
Content-Type: image/gif
```

```
...
```

```
--ThisRandomString
```

```
Content-Type: image/gif
```

```
...
```



Antete pentru control caching

- Trei tipuri de caching:
 - la client – cache privat
 - la proxy, server – cache-uri partajate
- Control caching – introdus in HTTP/1.1
 - se face prin antet **Cache-Control** cu valorile
 - **public** - nici o restricție pentru caching
 - **private** – nu in *shared caches*
 - **no-cache** – nici in browser, nici in proxy
- Exemplu

```
HTTP/1.1 200 OK
```

```
Date: Mon, 05 Feb 2005 04:33:19 GMT
```

```
Server: Apache/1.2.5
```

```
Last-Modified: Mon, 05 Feb 2005 04:30:28 GMT
```

```
Cache-Control: private
```

```
Pragma: no-cache
```

```
Content-Length: 2289
```

```
...
```



Consistenta cache-urilor (1)

- Asigura ca documentul din cache este consistent cu cel din server
- Solutie 1: Folosind comanda HEAD
 - clientul transmite HEAD
 - primește raspuns si verifica antet Last-Modified
 - transmite GET daca document mai nou decat copia din cache
- Cerere
 - HEAD http://www.cs.pub.ro/~ionescu/ HTTP/1.1
 - Host: www.cs.pub.ro
 - User-Agent: Mozilla/4.75 [en] (WinNT; U)
- Raspuns
 - HTTP/1.1 200 OK
 - Date: Mon, 05 Feb 2005 04:33:19 GMT
 - Server: Apache/1.2.5
 - Last-Modified: Mon, 05 Feb 2005 04:30:19 GMT
 - Content-Length: 2234
 - Content-Type: text/html

Consistenta cache-urilor (2)

- Solutie 2: Folosind comanda GET cu antet If-Modified-Since

- GET /~ionescu/ HTTP/1.1
 - Host: www.cs.pub.ro
 - If-Modified-Since: Mon, 04 Feb 2005 04:30:28 GMT

- serverul transmite

- HTTP/1.1 304 Not Modified
 - Date: Mon, 05 Feb 2005 04:33:19 GMT
 - Server: Apache/1.2.5

- sau

- HTTP/1.1 200 OK
 - Date: Mon, 05 Feb 2005 04:33:19 GMT
 - Server: Apache/1.2.5
 - Last-Modified: Mon, 05 Feb 2005 04:30:28 GMT
 - Content-Length: 2289

Solutie pentru performanta

- Clientul nu contacteaza serverul pentru orice cerere
 - Raspunsul unui server poate include data expirarii, care este memorata de client
 - HTTP/1.1 200 OK
 - Date: Mon, 05 Feb 2005 04:33:20 GMT
 - Content-Type: image/jpeg
 - Content-Length: 35782
 - Cache-Control: private
 - Expires: Tue, 06 Feb 2005 04:33:20 GMT
 - Last-Modified: Mon, 05 Feb 2005 04:33:18 GMT
 - Clientul verifica existenta paginii in cache
 - Nu exista – cere resursa neconditionat
 - Exista expirata- adauga la cerere antet If-Modified-Since
 - daca server raspunde cu 304 Not Modified foloseste intrarea din cache
 - Exista ne-expirata – foloseste intrarea din cache

Antete pentru autentificare si autorizare

- Autentificare de baza

- prin antet `de autorizare`
- nume si parola transmise codat Base64 (nu criptat) – atentie HTTPS

- Secventa de actiuni

- Cerere resursa restrictionata
- Server raspunde cu 401

```
HTTP/1.1 401 Authenticate
```

```
Date: Mon, 05 Feb 2005 04:33:19 GMT
```

```
Server: Apache/1.2.5
```

```
WWW-Authenticate: Basic realm="Capitol3"
```

- Browser retrimite cererea cu antet suplimentar de autorizare

```
GET /carte/capitol3/index.html HTTP/1.1
```

```
Date: Mon, 05 Feb 2005 04:33:20 GMT
```

```
Host: www.cs.pub.ro
```

```
Authorization: Basic eNCoDEd-userID:PaSSwoRd
```

- Server verifica credentialele de autorizare si satisface cererea (sau refuza cu 403)
- Browser foloseste credentiale si in viitoarele cereri la `URL dependente`

Antete de suport sesiune

- Informatia de stare transmisa prin mesajele HTTP
- Intelegerea initiata de server prin antet Set-Cookie

Set-Cookie: <nume>=<valoare>[; expires=<data>][; path=<cale>]
[;domain=<nume_domeniu>][; secure]

<nume>=<valoare> pereche atribut/valoare de trimis de browser

path, domain identifica cererile care sunt calificate

secure browser-ul trebuie sa transmita info pe legatura securizata

Cookie: <nume>=<valoare>

Inclus de browser pentru cererile referitoare la URL in care domeniul si calea corespund cu cele din Set-Cookie

Exemplu:

HTTP/1.1 200 OK

Set-Cookie: client=lon; path=/carte/capitol3/; domain=.pub.edu

GET /carte/capitol3/index.html HTTP/1.1

Host: www.cs.pub.edu

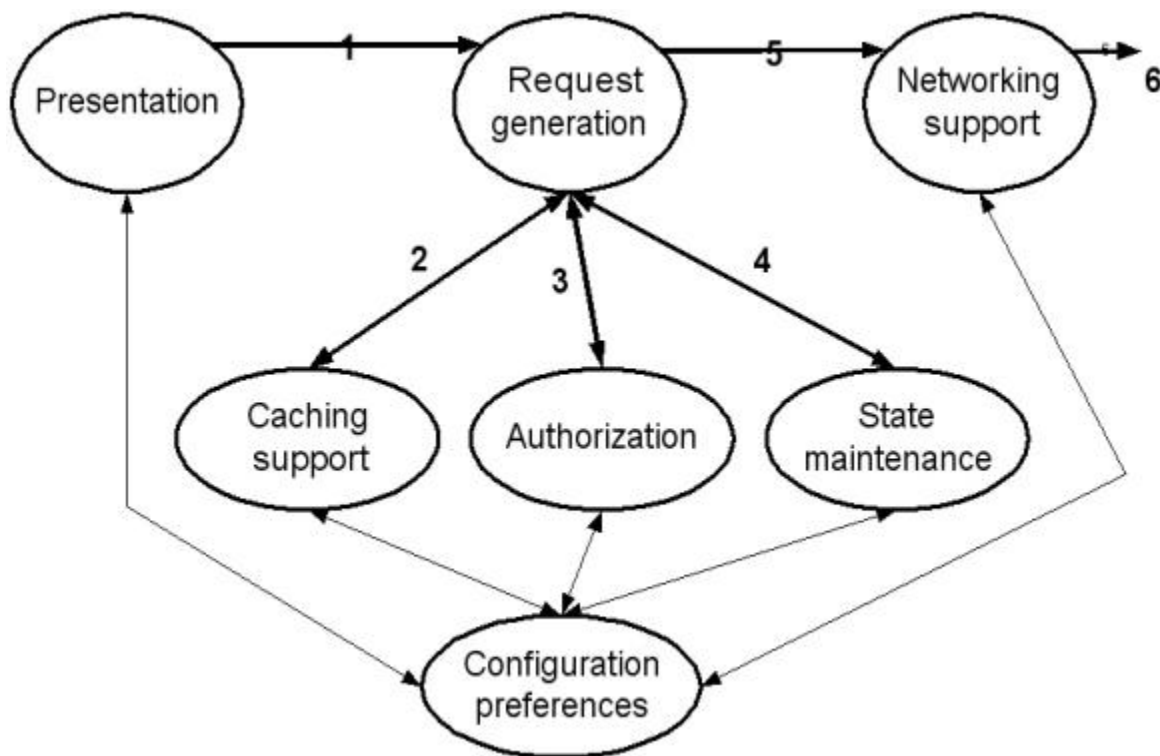
Cookie: client=lon



Browsere Web - Componente

- **Interfata utilizator:** dialogul cu utilizatorul
- **Generator cereri:** construiește cererile pentru server
- **Suport caching:** pastrează copii resurse găsite
- **Autorizare:** compune credențiale de autorizare când sunt cerute de server
- **Management stare:** pastrează stare browser între cereri și răspunsuri corelate
- **Procesare răspuns:** parsează răspunsul, face verificări și pasează rezultatul modului de interfață utilizator
- **Interpretare conținut:** procesare suplimentară pentru a înțelege obiecte multimedia, imagini, applets, cod JavaScript și informații de stil - style sheet information)
- **Suport rețea:** comunicare prin rețea
- **Configurare:** menține opțiunile de configurare pentru browser și permite utilizatorilor să le modifice

Generarea cererii



1. Identifica evenimentul si paseaza legatura

2. Verifica daca o copie a resursei este in cache

3. Verifica daca sunt necesare credentiale

4. Verifica daca trebuie incluse antete Cookie

5. Cererea este pasata modulului de retea

6. Transmite prin retea

Funcțiile modulelor din browser

- Interfata utilizator

- Afiseaza fereastra browser pentru renderizarea continutului primit de la [Interpretare continut](#)
- Permite accesul utilizator la funcțiile browser prin meniu, taste speciale etc
- Raspunde evenimentelor initiale de utilizator
 - Selectare/introducere URL
 - Umplere formulare
 - Activare butoane de navigare (ex. Back)
 - Vizualizare sursa paginii, info resurse etc.
 - Setare optiuni configurare
 - **Nu descarca imagini referite in pagina HTML**
 - **Rejecteaza cookies**
- Paseaza informatia de cerere la [Generator cereri](#)

Generator cereri

- Primește informația pentru cereri de la **Interfața utilizator** sau de la **Interpretare continuă**;
- Rezolvă **URL relativ**
 - URL relativ la locația curentă afișată (calea din HREF **nu începe cu /**)

Ex:

URL curent: `http://www.myserver.com/mydirectory/index.html`

Link: `...`

Rezolvat la: `http://www.myserver.com/mydirectory/altdirector/pag2.html`

- URL relativ la rădăcina Web server-ului corespunzător locației curente (calea din HREF **începe cu /**)

Ex:

URL curent: `http://www.myserver.com/mydirectory/index.html`

Link: `...`

Rezolvat la: `http://www.myserver.com/rootdirector/homepage.html`

– Construiește linia de cerere

METHOD

Implicit (la activare hyperlink) GET

In formular (specificat explicit) GET sau POST

/cale-resursa

Numai calea in HTTP/1.1

Tot URL in HTTP/1.0

HTTP/versiune

– Construiește antetele de baza

Host: www.cs.pub.ro

User-Agent: Mozilla/4.75 [en] (WinNT; U)

Referer: <http://www.cs.pub.ro/~ionescu/index.html>

Accept: text/html, text/plain, type/subtype

Accept-Charset: ISO-8859-1

...

Content-Type: mime-type/mime-subtype

Content-Length: xxx

Date:

- Intreaba **Support caching** daca exista intrare in cache
 - Nu exista – cere resursa neconditionat
 - Exista expirata - adauga la cerere antet **If-Modified-Since**
 - daca server raspunde cu **304 Not Modified**
 - » paseaza intrarea din cache la **Interpretare continut**
 - Exista ne-expirata – intoarce intrarea din cache
- Intreaba **Autorizare** daca e nevoie de autorizare pentru domain/path
 - Exista credentiale – adauga antet **Authorization**
- Intreaba **Management stare** despre cookies (domain/path)
 - Da – adauga antet **Cookie**
- Paseaza intreaga cerere la **Suport retea**
- Preferintele utilizatorului (**Configurare**) pot modifica fluxul cererii
 - nu se cer imaginile referite in pagina
 - nu se includ Cookies



- Construiește corp cerere

- se aplica pentru POST, PUT

- POST

- parametrii din formulare in corp comanda

- `Content-Type: application/x-www-form-urlencoded`

- `Content-Length: 6`

- `s=YHOO`

- PUT sau POST

- folosind MIME

- `Content-Type: multipart/multipart_subtype; boundary="ThisRandomString"`

- `--ThisRandomString`

- `Content-Type: tip/subtip partea 1`

- `Content-Transfer-Encoding: schema codificare partea 1`

- `continut partea 1`

- `--ThisRandomString`

- `Content-Type: tip/subtip partea 2`

- `Content-Transfer-Encoding: schema codificare partea 2`

- `continut partea 2`

Suport retea

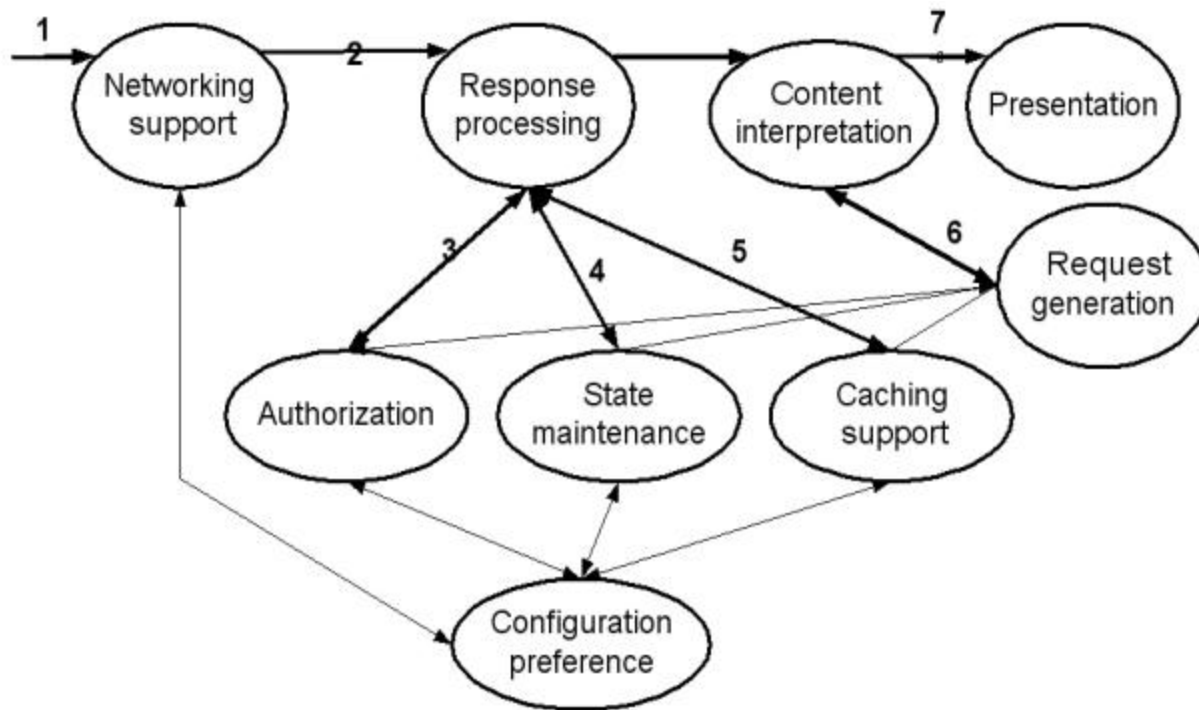
– Transmite cererea

- Primește cereri de la **Generator cereri** și le pune în coada transmisiei
- Întreabă **Configurare** pentru a determina dacă ținta este un proxy și alte opțiuni de rețea
- Deschide socket pentru a transmite cereri din coadă
 - transmite mai multe cereri la o conectare

– Tratează răspuns

- Așteaptă răspunsuri la cereri
- Pasează la **Procesare răspuns**

Procesarea raspunsului



1. Primește răspuns

2. Pasează răspuns

3. Cererea a fost rejectată – verifică dacă pot fi folosite credențiale

4. Dacă se cere info cookie, contactează modulul management stare

5. Contactează suport caching pentru memorarea răspunsului;
apoi pasează răspuns la interpretare conținut

6. Decodifică corp răspuns, procesează diferite tipuri MIME și parsează conținut ptr determinare resurse adiționale necesare (în răspuns sunt referințe la alte resurse)

7. Conținut pasat la modul prezentare

Procesare raspuns

- Verifica stare 401 (ne-autorizat)
 - Cere modulului de **Autorizare** credentiale ptr domeniul din antet **WWW-Authenticate**
 - Exista – retransmite cerere cu credentiale adaugate
 - Nu – cere credentiale de la utilizator (prin **Interfata utilizator**) si retransmite
 - Credentialele sunt memorate pe durata unei sesiuni
- Verifica stare redirectare (301/302/307)
 - Daca
 - HTTP/1.1 301 Moved Permanently**
 - Location: <http://www.alta-locatie.com/pagina.html>**
 - Retransmite cerere la URL din antet **Location**
 - GET /pagina.html HTTP/1.1**
 - Host: www.alta-locatie.com**
 - ...**
 - Daca 301, memoreaza in *persistent lookup table* pentru redirectare automata a cererilor urmatoare

- Verifica antet **Set-Cookie**
 - Cere **Management stare** sa memoreze cookie in browser
 - Memorarea: pe sesiune / pentru o durata specificata
- Verifica optiuni caching si transmite cerere la **Suport caching** de a memora resursele obtinute
 - raspunsul poate include data expirarii
 - HTTP/1.1 200 OK**
 - Date: Mon, 05 Feb 2005 04:33:20 GMT**
 - Content-Type: image/jpeg**
 - Content-Length: 35782**
 - Cache-Control: private**
 - Expires: Tue, 06 Feb 2005 04:33:20 GMT**
 - Last-Modified: Mon, 05 Feb 2005 04:33:18 GMT**
 - ...**
- Paseaza rezultat la **Interpretare continuut**



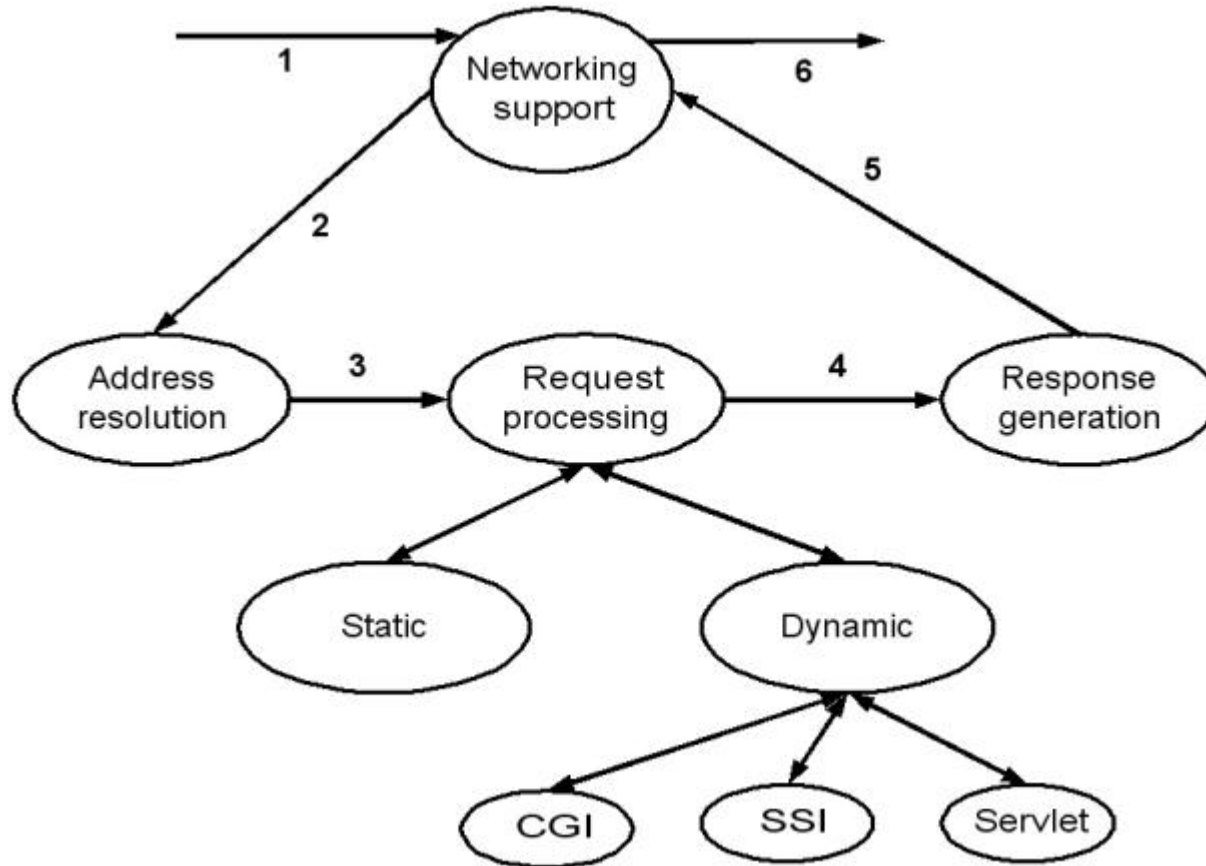
- Interpretare continuu

- Primește conținut de la [Procesare raspuns](#) (Uneori de la [Suport caching](#))
- Examinează antete codificare și, eventual, decodifică conținut
 - [Content-Transfer-Encoding: chunked](#)
 - [Content-Encoding: compress | gzip](#)
- Pasează conținut decodificat la module specifice tipului MIME pe baza antet [Content-Type](#)
- Dacă referințe la alte resurse, pasează URL la [Generator cereri](#)
- Pasează fiecare modul prelucrat la [Interfața utilizator](#)

- Configurare

- Furnizează mecanisme de persistență pentru setările din browser
- Interfața utilizator pentru setări preferințe
- Primește cereri de la alte module pentru a determina acțiunile în funcție de preferințele utilizatorilor

Operații Server



1. serverul primește o cerere
2. Pasează la modulul de **rezolvare a adresei** care (a) determină target-ul; (b) determină dacă cererea conține conținut static / dinamic; (c) examinează credențialele de autorizare.
3. Pasează la modulul **procesare cerere**, care apelează sub-module necesare

4. Rezultat este pasat **generatorului de răspuns**
5. Este pasat modulului **suport rețea**
6. Transmite clientului



- Rezolvarea adresei

- selectează virtual host

- nu exista antet Host: -> eroare 400 Bad request
 - exista -> determina domeniul
 - > determina parametrii config. logica (proprii virtual host)

<VirtualHost www.ceva.com>

ServerAdmin webmaster@calculatoare.com

Alias /test/servlet/test

Alias /images /static/images

DocumentRoot /www/docs/ceva

ServerName www.ceva.com

ErrorLog logs/ceva-error-log

CustomLog logs/ceva-access-log common

</VirtualHost>

- rezolva alias-uri

- **http://www.ceva.com/test?a=1&b=2 /test -> /servlet/test**
 - **http://www.ceva.com/images/nou.gif /images -> /static/images**



- mapare adresa

- pagina statica

- URL `http://www.ceva.com/pagini/cucu.html`
- configurare `DocumentRoot /www/docs/ceva`
 - » `/pagini/cucu.html -> /www/docs/ceva/pagini/cucu.html`

- pagina dinamica

- **sufix nume fisier** sau **prefix URL** decide cine proceseaza
 - » prefix URL `/servlet/ /cgi-bin/`
 - » sufix nume `.cgi .php`

- verifica autentificare

- cod eroare daca resursa ceruta este protejata



• Procesare cerere

- regaseste continut
- seteaza tipul MIME conform configurare server

text/css	css
text/html	html htm
text/plain	asc txt
text.xml	xml
video/mpeg	mpeg mpg mpe

- seteaza alte antete (Content-Length, Last-Modified etc.)
- antet transfer pe bucati (chunked)

HTTP/1.1 200 OK

Content-Type text/plain

Content-Transfer-Encoding: chunked

1b; comentariu

qwertyuiopasdfghjklzxcvbnm12

10

1234567890asdfgh

0

antet-suplimentar: valoare

footer: alta-valoare

• Conexiune persistenta

- cozi de cereri si de raspunsuri
 - cerere mutata din intrare in iesire la luarea in considerare
 - raspunsuri trimise in ordinea cererilor din iesire



Funcționare server

- server HTTP = set de thread-uri care procesează cererile clientilor
- pot fi pastrate continuu în execuție
- serverul poate suporta conexiuni persistente
- cu un nr maxim cereri în așteptare
- cu timeout așteptare cerere nouă
- număr maxim de cereri procesate fără repornire
- timp maxim de procesare a unei cereri

- **Fisier configurare fizica (Apache pentru Windows)**

ServerName demo

ServerRoot "C:/Program Files/Apache Group/Apache"

ServerType Standalone

Port 80

KeepAlive On

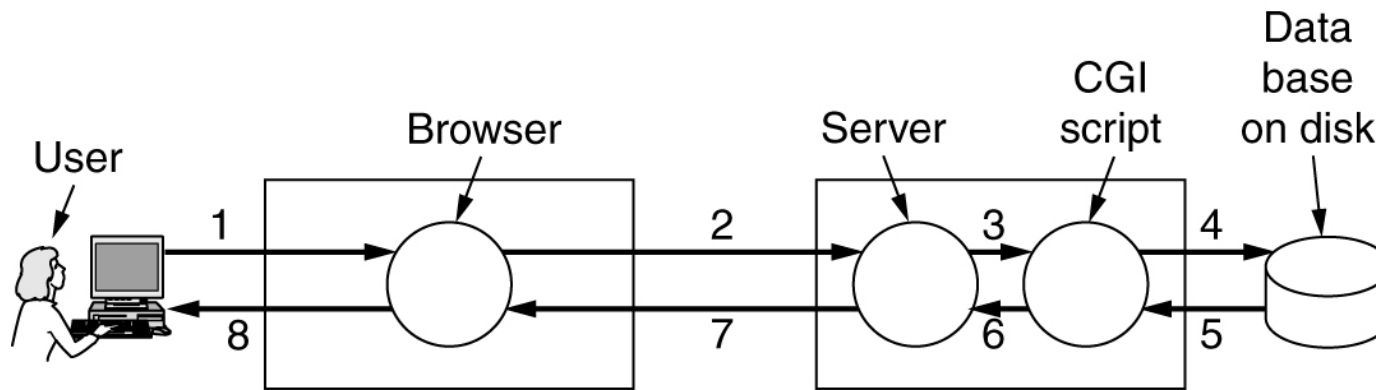
MaxKeepAliveRequest 100

KeepAliveTimeout 15

MaxRequestsPerChild 200

Timeout 300

Documente Web Dinamice



1. User fills in form
2. Form sent back
3. Handed to CGI
4. CGI queries DB
5. Record found
6. CGI builds page
7. Page returned
8. Page displayed

CGI = Common Gateway Interface – primul standard de facto ptr. creare dinamica pagini

Definit in Unix, extins la alte sisteme (Windows, Macintosh)

Bazat pe **variabile de mediu** la care au acces toate programele CGI

unele contin info independente de cereri

altele sunt actualizate la fiecare cerere



CGI - Variabilele de mediu

Variabilă	Descriere
SERVER_SOFTWARE	numele și versiunea software-ului serverului
SERVER_NAME	numele sau adresa IP a serverului
GATEWAY_INTERFACE	versiunea CGI folosită de server
SERVER_PROTOCOL	numele și versiunea protocolului folosit de server
SERVER_PORT	portul la care serverul primește cereri
REQUEST_METHOD	metoda folosită (de ex. POST)
PATH_INFO	informația suplimentară de cale furnizată de client
PATH_TRANSLATED	versiunea tradusă a informației suplimentare de cale
SCRIPT_NAME	calea în URL, după numele serverului
QUERY_STRING	informația aflată după ? în URL
REMOTE_HOST	numele gazdei clientului (browser-ului)
REMOTE_ADDR	adresa IP a gazdei clientului (browser-ului)
AUTH_TYPE	metoda de autentificare folosită de server
REMOTE_USER	numele utilizatorului
REMOTE_IDENT	folosit pentru login
CONTENT_TYPE	tipul datelor atașate (pentru metode de comunicare POST)
CONTENT_LENGTH	lungimea datelor

Actiuni server

la primirea unei comenzi POST <http://mysite.org/cgi-bin/zip.cgi> HTTP/1.1

1. determina ca [/cgi-bin/zip.cgi](#) trebuie tratat ca CGI
2. mapeaza calea la nume fisier [/www/cgi-bin/zip.cgi](#)
3. verifica legalitate director CGI [/www/cgi-bin/](#)
4. seteaza **variabile de mediu** pe baza cererii si antetelor asociate
5. In Unix
 - **creaza un proces** pentru executia programului CGI (spawn)
 - paseaza corpul cererii prin intrarea standard **input**
 - dirijeaza iesirea standard **output** catre modulul din server care primeste raspunsul
6. parseaza raspunsul si adauga antete implicite (stare, tip continut etc.)

Limbajul utilizat pentru CGI

- C, C++ FORTRAN
- Perl, TCL sau UNIX shell

Un exemplu de program CGI

```
#!/bin/sh  
#  
# script CGI care afisaza data si ora  
# scrie antetul documentului urmat de o linie alba
```

```
echo Content-type: text/plain  
echo
```

```
#scrie data
```

```
echo Document creat la data de 'date'
```

O ieșire posibilă:

```
Content-type: text/plain  
Document creat la data de Sun Sep 26 10:35:12 EST 2005
```

Ce se afișează:

```
Document creat la data de Sun Sep 26 10:35:12 EST 2005
```

Transmiterea informației de stare

Solutii care nu folosesc cookies

păstrarea stării în URL – numărul de vizitări ale paginii curente

```
#!/bin/sh
echo Content-type: text/html
Echo

N=$QUERY_STRING
echo "<HTML>"
case "x$N" in
x)      N=1
        echo "Pagina initiala.<br><br>"
        ;;
x[0-9]*) N='expr $N + 1'
        echo "Ati cerut aceasta pagina de $N ori. <br> <br>"
        ;;
*)      echo "URL-ul folosit este incorect.</HTML>"
        exit 0
        ;;
esac
echo "<a href=\"http://$SERVER_NAME$SCRIPT_NAME?$N\">"
echo "Apasati aici pentru readucerea paginii.</a></HTML>"
```

Ex: server www.personal.com

scriptul din exemplu în fișierul [/cgi/ex](#)

la primul apel <http://www.personal.com/cgi/ex>

scriptul va genera:

Content-type: text/html

<HTML>

Pagina initiala.

Apasati aici pentru readucerea paginii.</HTML>

Browser-ul va afișa liniile:

Pagina initiala.

[Apasati aici pentru readucerea paginii.](#)

Altă posibilitate: păstrarea stării într-un *fișier*

Scriptul înregistrează adresele IP ale clienților de la care a fost contactat

```
#!/bin/sh
FILE=adreseIP

echo Content-type: text/plain
Echo

#Verifica daca adresa IP apare in fisier
if grep -s $REMOTE_ADDR $FILE >/dev/null 2>&1
then
    echo Calculatorul $REMOTE_ADDR a cerut acest URL anterior
else
    #Adauga adresa noua in fisier
    echo $REMOTE_ADDR >> $FILE
    echo Primul contact de la calculatorul $REMOTE_ADDR
fi
```

Exemplu: invocarea vine de la **128.30.4.57** și
nu este prima încercare.

Răspunsul alcătuit de program are forma:

Content-type: text/plain

Calculatorul 128.30.4.57 a cerut acest URL anterior

La primirea lui, browser-ul afișează

Calculatorul 128.30.4.57 a cerut acest URL anterior

SSI - ServerSide Includes

- Facilități de includere, la server, în pagini HTML, a unor fișiere auxiliare sau rezultate ale unor scripturi CGI

- Utilizare

Creare tipare de pagini HTML (templates)

Completare cu rezultatele unor scripturi sau fișiere auxiliare

```
<HTML>
```

```
<HEAD><TITLE>Hello</TITLE></HEAD>
```

```
<BODY>
```

```
<!--#exec cgi="http://mysite.org/cgi-bin/zip-ssi.cgi"-->
```

```
</BODY>
```

```
</HTML>
```

- Avantaje

Faciliteaza scrierea

Separa designul paginii de proiectarea aplicatiei

SSI - directive

Un server poate folosi directive de includere inserate în comentarii, având forma:

```
<!--#aCommand name=value -->
```

unde *aCommand* poate fi una din următoarele:

echo: – insereaza în document valoarea variabilei specificate de argument

```
<!--#echo var="LAST_MODIFIED"-->
```

include: – include documentul identificat de argument.

```
<!--#include file="adresa.html"-->
```

```
<!--#include virtual="/public/adresa.html"-->
```

exec: – plasează în document rezultatul execuției unui script;
argumentul specifică un script shell (*cmd*) sau un program CGI (*cgi*)

```
<!--#exec cgi="http://mysite.org/cgi-bin/zip-ssi.cgi"-->
```

fsize: – include în document dimensiunea fișierului specificat de argument

```
<!--#fsize file="cgi-refguide.ps"-->
```

config: – directivă de configurare pentru server



Aplicatii Web

- Aplicatie Web
 - aplicatie client-server bazata pe mediul Web, in particular pe utilizarea browser Web drept client
- Solutii de dezvoltare a aplicatiilor Web
 - scripturi / cod executabil (CGI, servleti Java)
 - sabloane (SSI, Cold Fusion, WebMacro/Velocity)
 - solutii hibride (PHP=hypertext Preprocessor, ASP=active server pages, JSP=java server pages)
 - Frameworks
 - Caracteristici
 - **Paradigma MVC - model-view-controller**
 - **Suport management stare si autentificare**
 - **Suport acces la date**
 - Abordari
 - **JSP Model 2 (Sun) -> Struts framework**
 - **Bazate pe XML**