Nivelul Aplicaţie

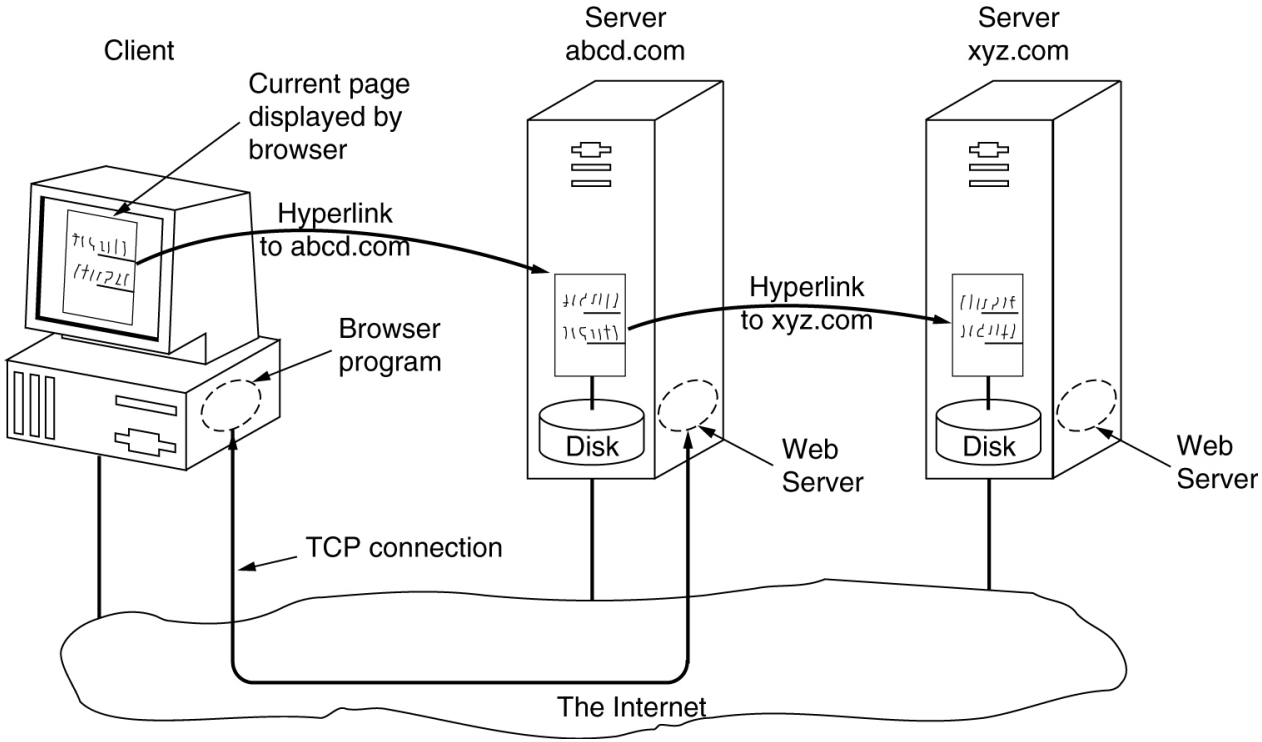
World Wide Web

Cuprins

- Functionarea WWW
- URL
- HTML – marcaje, formulare
- HTTP
- Clientul (Browser)
- Serverul

World Wide Web

- Set de documente (pagini) cu legături intre ele (hyperlinks)
- Distribuite pe maşini diferite
- Include o pagina de referinţa (home page)
  - pagina initiala a unui site Web
  - pagina afisata la pornirea unui browser



Interacţiunea client – server (reluare!)

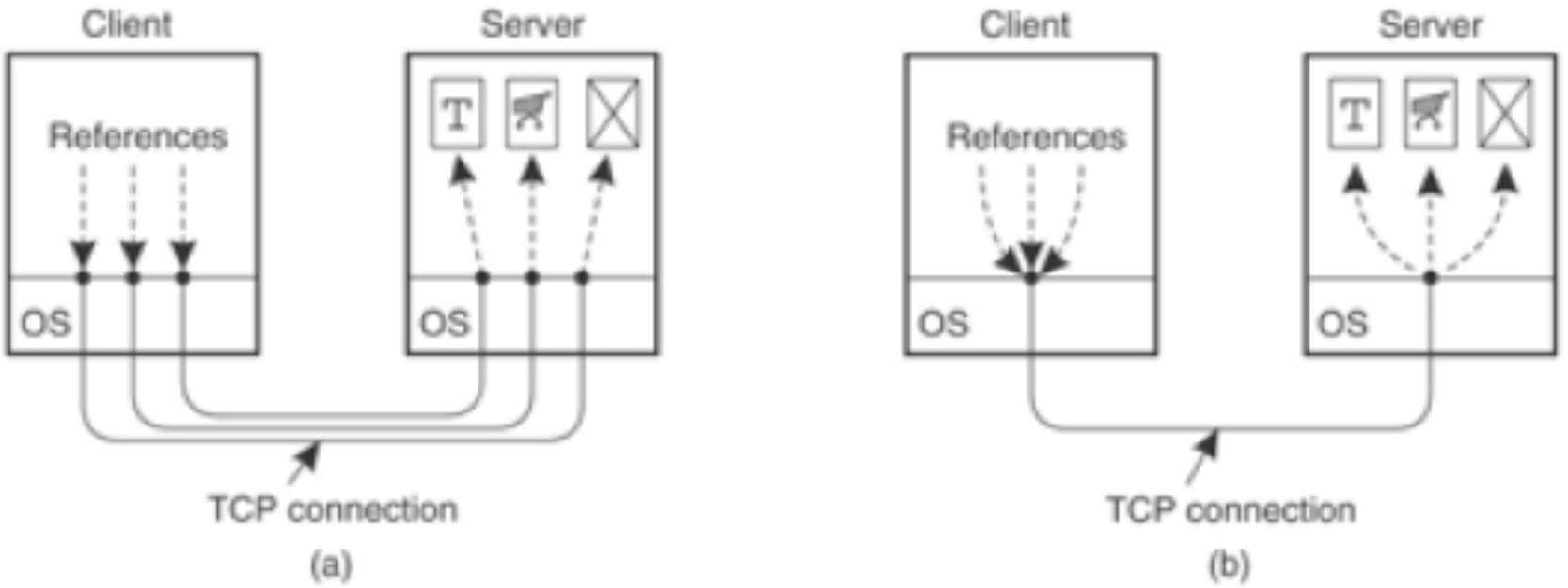
- Browser - determina URL
- Browser - cere DNS-ului adresa IP pentru www.w3.org
  - DNS - raspunde cu 18.23.0.23
- Browser - deschide o conexiune TCP la port 80 pe 18.23.0.23
- Browser - trimite o comanda

```
GET /hypertext/www/TheProject.html
```

  - Server www.w3.org - trimite fisierul TheProject.html
- Conexiunea TCP este inchisa
- Browser - afişează conţinutul din TheProject.html
- Browser - extrage si afiseaza toate imaginile din TheProject.html (se deschide o noua conexiune TCP pentru fiecare imagine)

Conexiuni persistente

- Introduse 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 obişnuite

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

HTML - HyperText Markup Language

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>
```

Ce afiseaza browser-ul

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

O selecţie de marcaje uzuale

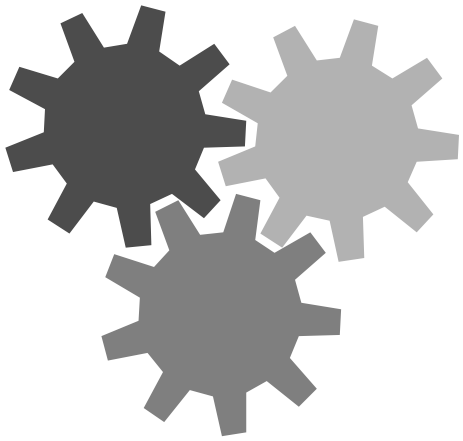
Marcaj	Descriere
<html> ... </html>	Delimitează textul scris în HTML
<head> ... </head>	Delimitează zona de antet
<title> ... </title>	Defineşte titlul (nu este afişat de programul de navigare)
<body> ... </body>	Delimitează zona de corp
<h*n*> ... </h*n*>	Delimitează un titlu de nivel *n*
...	Text îngroşat
...	Text cursiv
...	Centrat pe orizontală
 	Trecere la linie nouă
<p>	Început de paragraf
...	Delimitează o listă neordonată
...	Delimitează o listă ordonată (numerotată)
...	Delimitează un element într-o listă ordonată sau neordonată
<hr>	Linie orizontală
<img src="URL" ALT="text">	Afişează o imagine în acest loc (sau text-ul specificat)
text	Hiper-legătură la o pagină
text	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

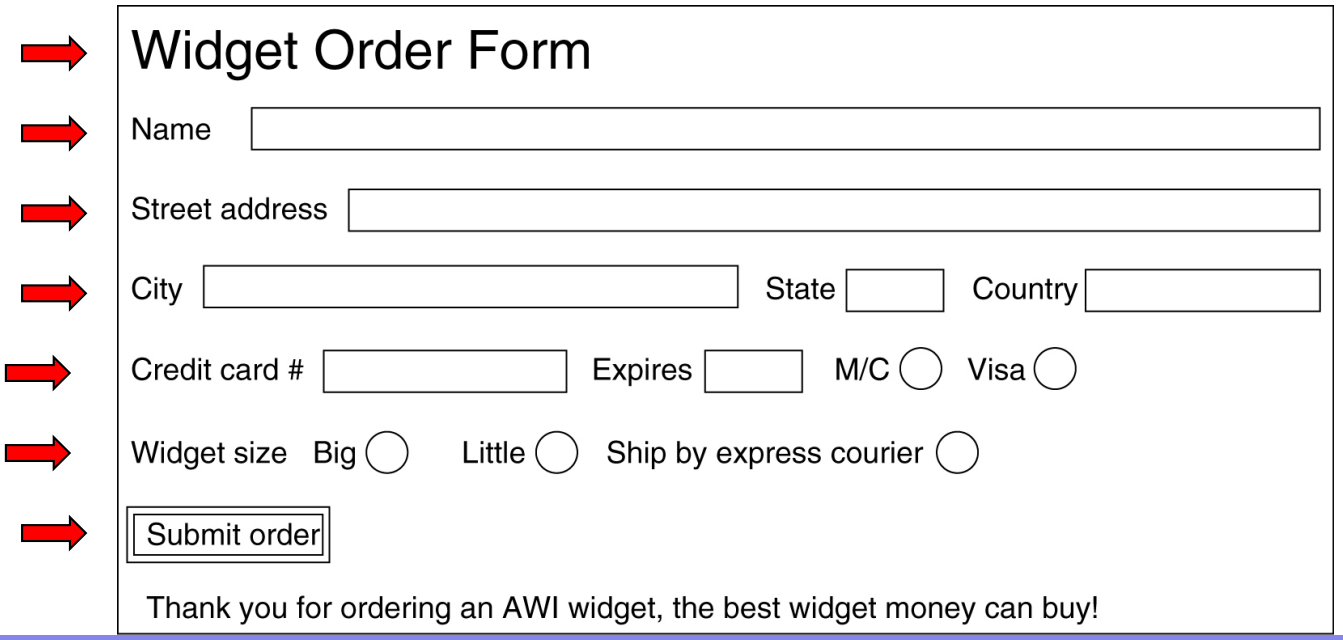
- 1-800-WIDGETS
- 1-415-765-4321

Formulare – marcaje specifice

element HTML	Parametri	Semnificaţie
<INPUT>, TYPE=text	NAME, SIZE, MAXLENGTH	câmp de intrare (tipul implicit)
<INPUT>, TYPE=radio	NAME, VALUE	buton radio
<INPUT>, TYPE=checkbox	NAME, CHECKED	casetã de selecţie
<INPUT>, TYPE=password	NAME, SIZE, MAXLENGTH	câmp de parolã
<INPUT>, TYPE=reset sau submit		buton de acţiune
<INPUT>, TYPE=image	NAME, ALIGN, SRC	hartã (imagine) activã
<INPUT>, TYPE=hidden	NAME,	element ascuns
<SELECT>	NAME, OPTION, MULTIPLE	listã de selecţie
<TEXTAREA>	NAME, COLS, ROWS, WRAP	zonã de editare

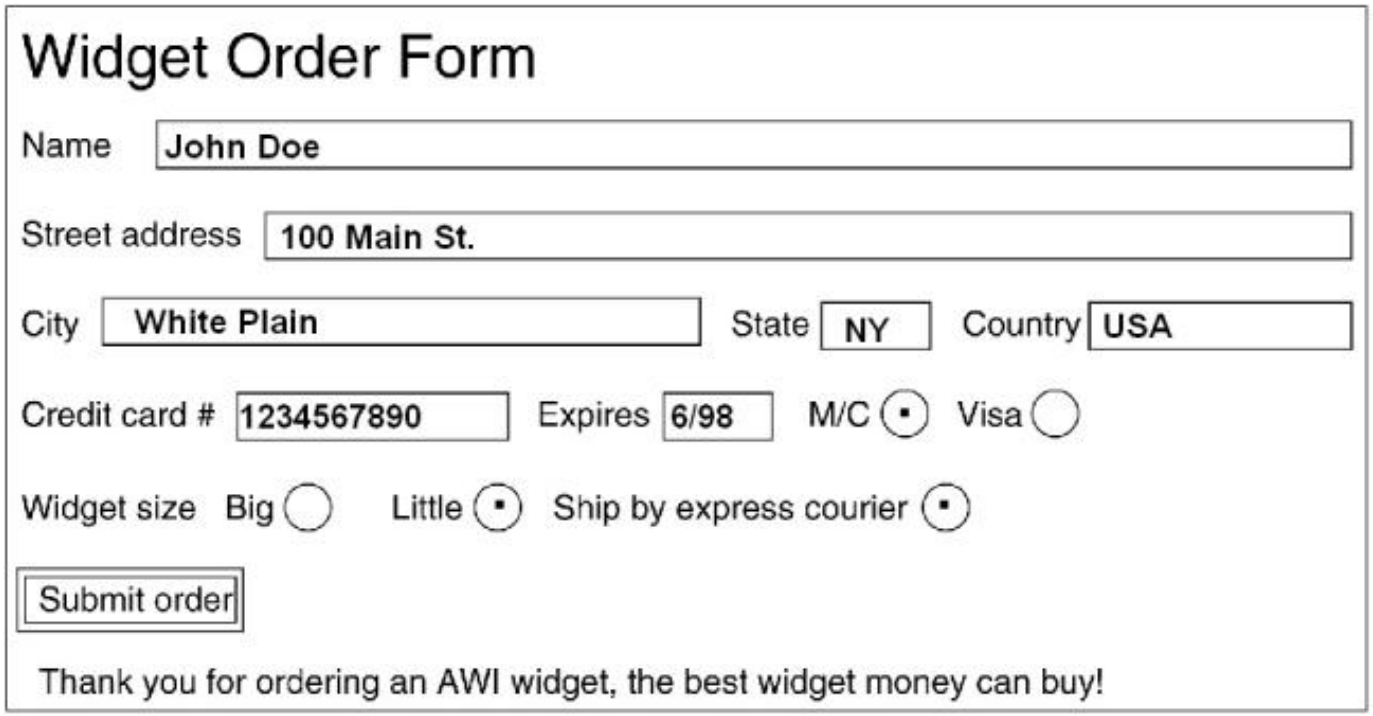
Formulare – un exemplu

```
<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>
Thank you for ordering an AWI widget, the best widget money can buy!
</form>
</body>
</html>
```



Formulare

Un text cu informaţiile completate de utilizator



customer=John+Doe&address=100+Main+St.&city=White+Plain&
state=NY&country=USA&cardno=1234567890&expires6/98&cc=mastercard&
product=cheap&express=on

(împărţit 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>
```

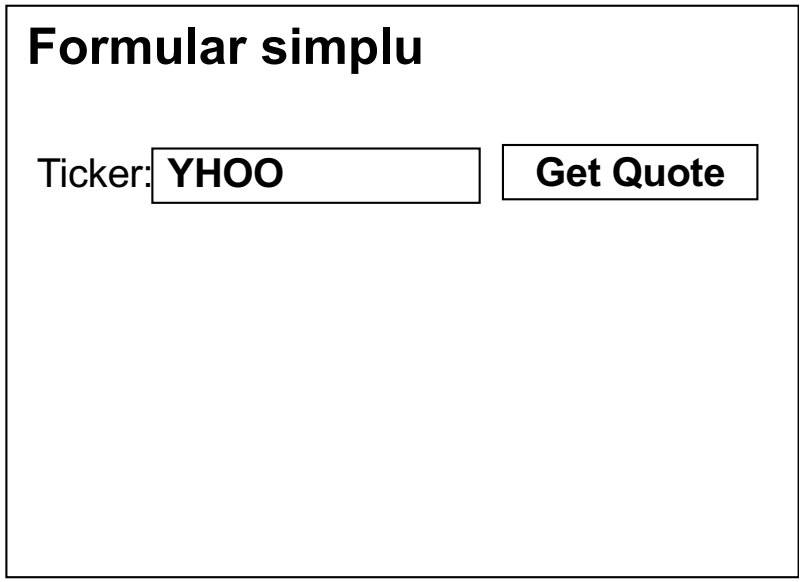
Metode HTTP

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	Ştergerea unei pagini de Web
TRACE	Transmite in ecou cererea care a sosit
OPTIONS	Interogarea anumitor opţiuni
CONNECT	Folosit ptr conectare prin proxy server pe conexiune tunel

Exemplu GET

- **Formular HTML**

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



URL construit de browser pentru intrarea YHOO

http://financiar.yahoo.com/q?

Cerere HTTP

GET /q?s=YHOO HTTP/1.1

Host: financiar.yahoo.com

User-Agent: Mozilla/4.75 [en]

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	Semnificaţie	Exemple
1xx	Informaţie	100 = serverul acceptă continuarea tratarii cererii de la client (asociat cu un antet Expect din cerere)
2xx	Succes	200 = cerere reuşită; 204 = nu există conţinut
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	Informaţie asupra programului de navigare şi 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	Informaţie despre server (ex. Server: Apache/1.2.5)

Antete Mesaje HTTP (2)

Antet	Tip	Descriere
Content-Encoding	Răspuns	Cum este codat conţinutului (de exemplu, gzip)
Content-Length	Răspuns	Lungimea paginii în octeţi
Content-Type	Răspuns	Tipul/subtipul MIME al paginii
Last-Modified	Răspuns	Ora şi data la care pagina a fost ultima dată modificată
Location	Răspuns	O indicatie pentru client pentru redirectarea cererii
Accept-Ranges	Răspuns	Serverul va accepta cereri în anumite limite de octeţi
Date	Ambele	Data şi ora la care mesajul a fost trimis
Connection	Ambele	Intentia de a pastra 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 de server prin antet Cache-Control cu valorile
  - public - nici o restrictie 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
Content-Length: 2289
...
```

Consistenta cache-urilor (1)

- Asigura ca documentul din cache este acelasi cu cel din server
- **Solutie 1**: Folosind comanda HEAD
  - clientul transmite **HEAD**
  - primeste raspuns si verifica antet **Last-Modified**
  - transmite **GET** daca document din server este mai nou dacat 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
. . .
```

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
. . .
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
. . .
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

Autentificare si autorizare

- Autentificare de baza
  - permite accesul la pagini protejate
  - prin antet de autorizare
  - nume si parola transmise codat Base64 (nu criptat)
    - atentie, trebuie folosit HTTPS
- **Secventa de actiuni**
  - Clientul cere 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"
```

  - realm defineste domeniul protejat

Autentificare si autorizare (2)

- 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)

- Odata trimise credentialele, browserul retrimite automat antetul de autorizare si credentiale in viitoarele cereri la URL dependente (fisiere din subdirectoare)

Ex. **http://cs.pub.ro/~popescu/clase/**

depinde de **http://cs.pub.ro/~popescu/**

care este un prefix al primului

Suport sesiune

Cookie este un mecanism ce permite transmiterea unor informatii de stare prin mesaje HTTP

- ex. info stare - identificatorul unei sesiuni

- Intelegerea este 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 inapoi de browser

path, domain identifica cererile care sunt calificate

- pentru domeniul .pub.edu domeniile calificate au forma *.pub.edu
- pentru calea /test/ caile calificate sunt de forma /test/*

secure browser-ul trebuie sa transmita info pe legatura securizata

Suport sesiune (2)

- Intelegerea este acceptata de client prin antet Cookie

  Cookie: <nume>=<valoare>

  inclus de browser in cererile referitoare la URL in care domeniul si / sau calea sunt calificate

Exemplu:

HTTP/1.1 200 OK

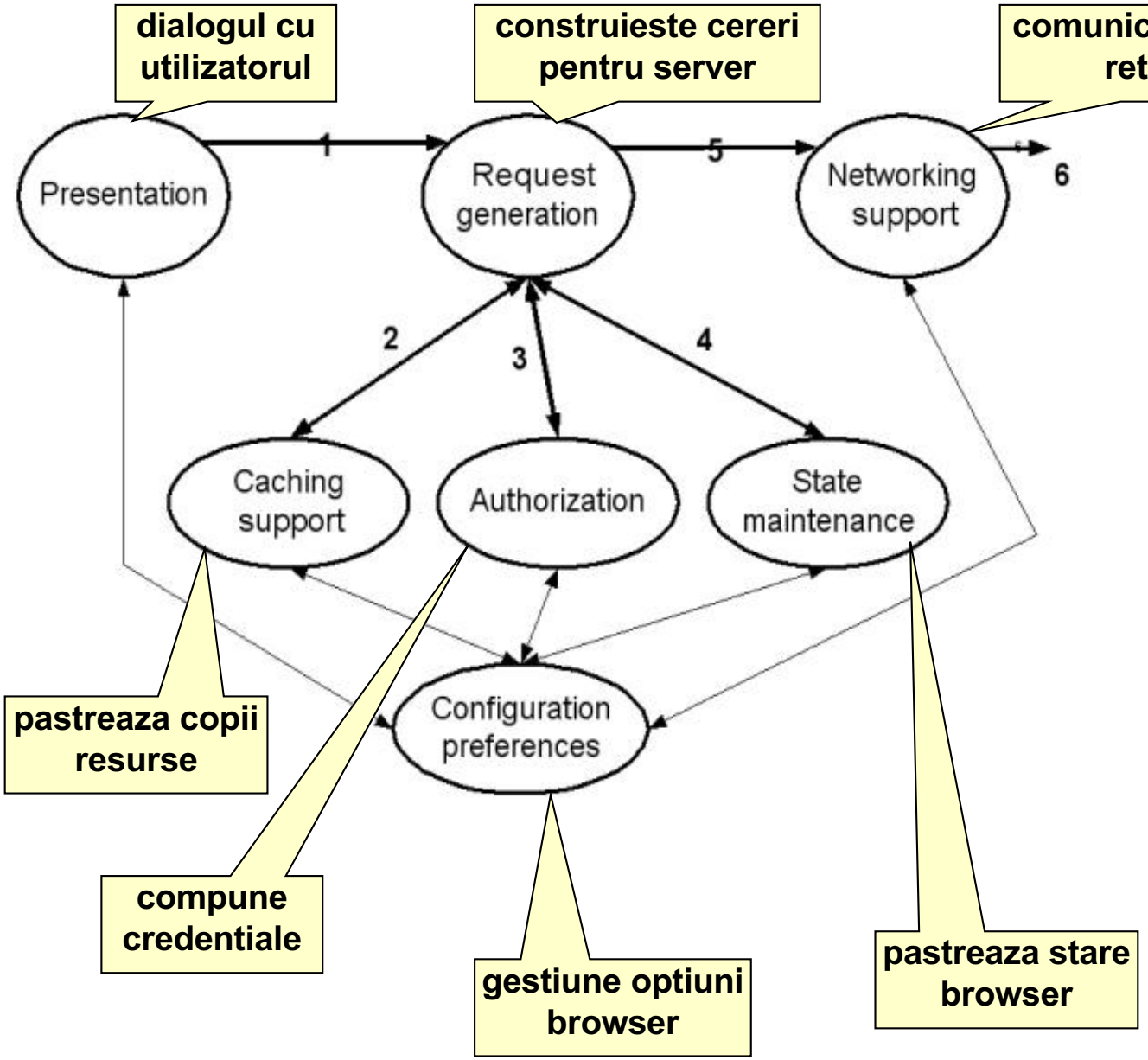
Set-Cookie: client=Ion; path=/carte/capitol3/; domain=.pub.edu

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

Host: www.cs.pub.edu

Cookie: client=Ion

Schema browser - Generarea unei cereri



Functionarea modul prezentare este bazata pe evenimente (ex. selectie URL afisat)

1. Identifica evenimentul si paseaza legatura (URL)
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

Functiile modulelor din browser

Interfata utilizator

- Afiseaza pe ecran rezultatul primit de la modulul Interpretare continut (din grupul de procesare a raspunsului)
- Executa actiunile initiate de utilizator, prin meniu, taste speciale etc.
  - Selectare/introducere URL
  - Completare 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

- Primeste legatura (URL) la pagina care va fi ceruta
- URL poate fi:
  - absolut (ex. introdus manual) – este complet http://domeniu/cale → nu trebuie prelucrat
  - relativ (ex. preluat din pagina curenta de la modulul Interpretare continut) → trebuie rezovat!

Sunt doua cazuri:

(1) - URL **relativ** la directorul curent al paginii afisate (calea din HREF nu incepe cu /)

Ex:

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

Link in pagina: **<A HREF ="altdirector/pag2.html>…</A>**

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

Generator cereri (2)

(2) - URL **relativ** la directorul radacina al serverului Web cu numele din URL-ul paginii curente (calea din HREF incepe cu /)

Ex:

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

Link in pagina: **<A HREF ="/rootdirector/homepage.html>…</A>**

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

– Construieste linia de cerere, care are 3 componente

METODA

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

- Construieste 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:

Referer – pagina in care se afla link-ul activat de utilizator

- Intreaba Suport 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
- Construieste corp cerere (vezi slide-uri urmatoare)
- Paseaza intreaga cerere la Suport retea
- Preferintele utilizatorului (Configurare) pot modifica fluxul cererii
  - nu se cer imaginile referite in pagina
  - nu se includ Cookies

- Construieste 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

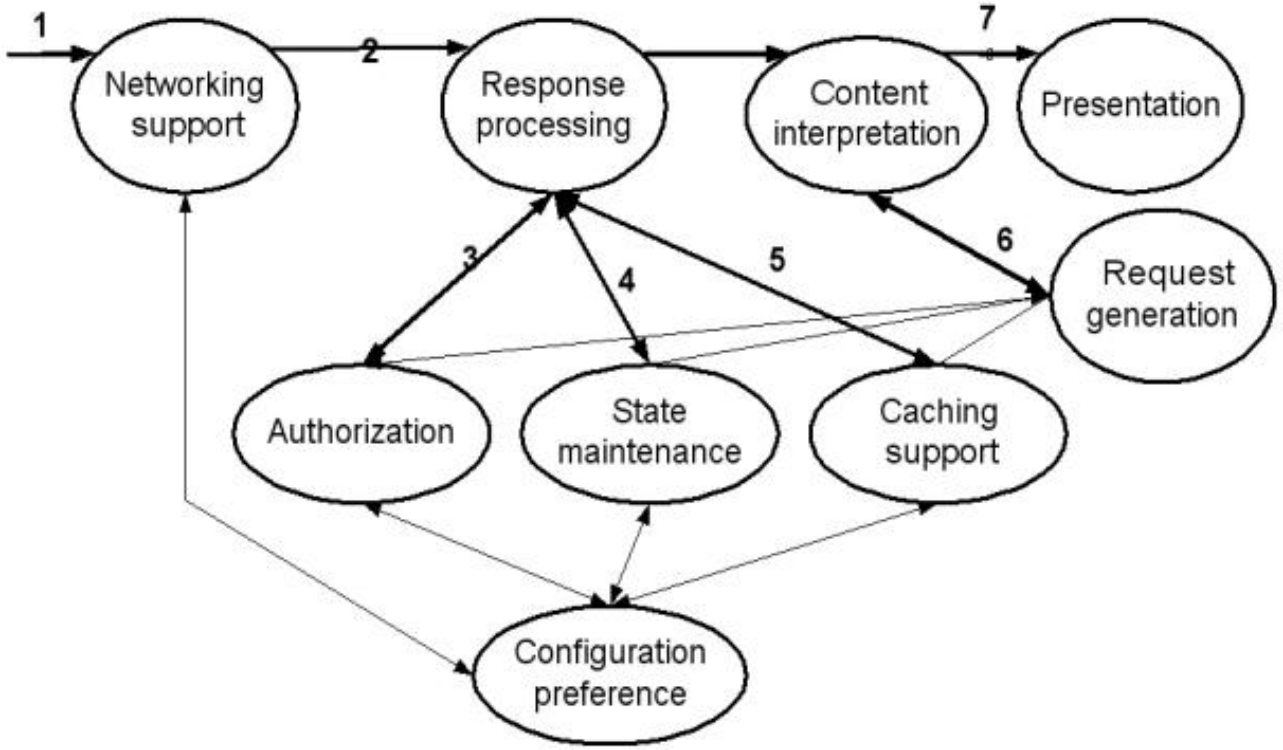
```
Content-Type: multipart/multipart_subtype;
  boundary="ThisRandomString”

--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
  - Primeste cereri de la Generator cereri si le pune in coada transmisie
  - Intreaba Configurare ptr a determina daca tinta este un proxy si alte optiuni retea
  - Deschide socket pentru a transmite cereri din coada
    - transmite mai multe cereri la o conectare

- Trateaza raspuns
  - Asteapta raspunsuri la cereri
  - Paseaza la Procesare raspuns

Procesarea raspunsului



1. Primeste raspuns

2. Paseaza raspuns la modulul de procesare

3. Cererea a fost rejectata – verifica daca pot fi folosite credentiale

– verifica redirectare

4. Daca se cere info cookie, contacteaza modulul management stare

5. Contacteaza suport caching pentru memorarea raspunsului;
apoi paseaza raspuns la interpretare continut

6. Decodifica corp raspuns, proceseaza diferite tipuri MIME si parseaza continut ptr eventuale referinte la resurse aditionale

7. Continut 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
- Daca 302 (pagina mutata temporar) - nu memoreaza

- 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 pentru a memora resursele obtinute
  - raspunsul poate include data expirarii

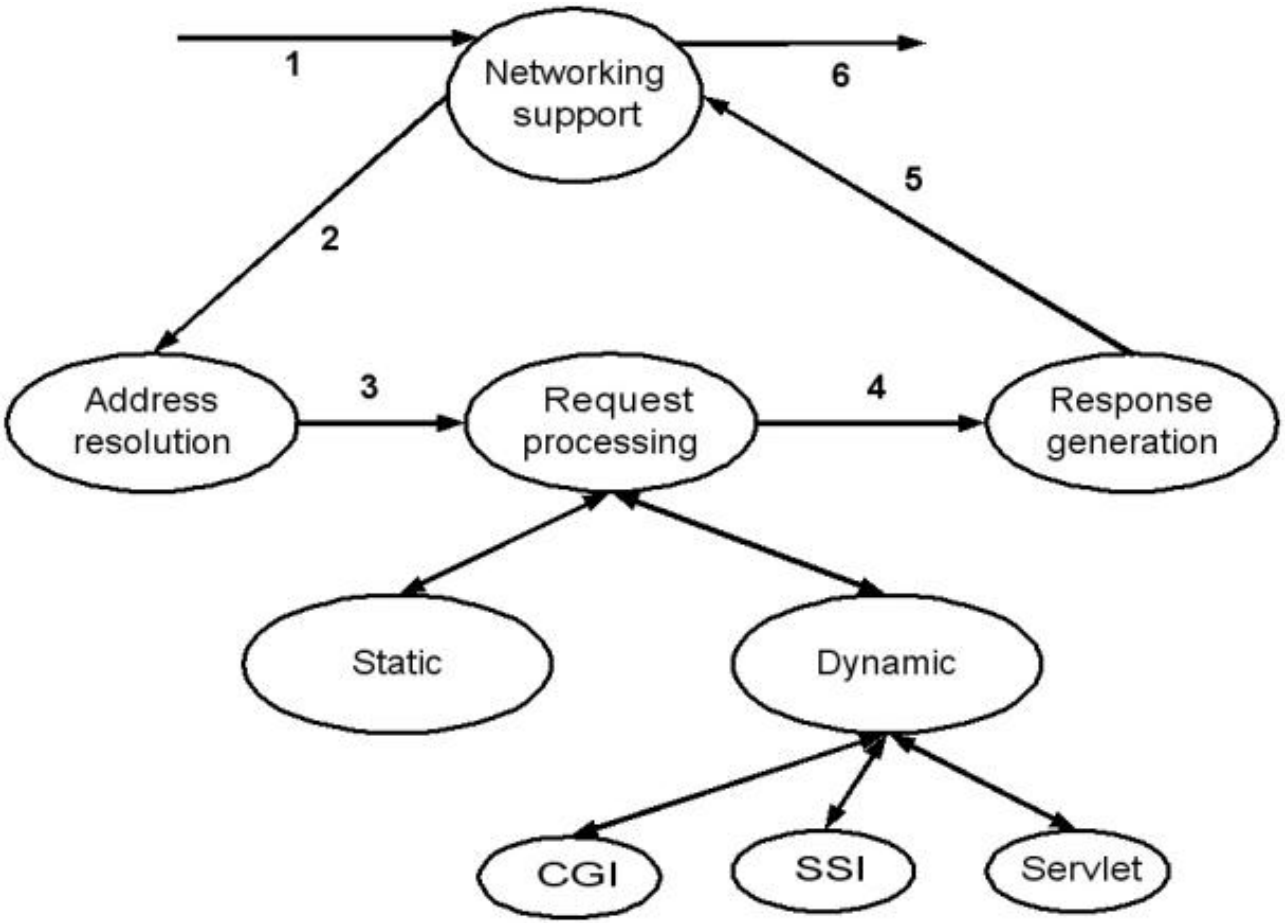
```
HTTP/1.1 200 OK
.  .  .
Cache-Control: private
Expires: Tue, 06 Feb 2005 04:33:20 GMT
Last-Modified: Mon, 05 Feb 2005 04:33:18 GMT
…
```

- Paseaza rezultat la Interpretare continut

- Interpretare continut
  - Primeste continut de la Procesare raspuns (Uneori de la Suport caching)
  - Examineaza antete codificare si, daca e cazul, decodifica continut
    - Content-Transfer-Encoding: chunked
    - Content-Encoding: compress | gzip
  - Paseaza continut decodificat la module specifice tipului MIME pe baza antet Content-Type
  - Daca link-uri la alte resurse, paseaza URL la Generator cereri
  - Paseaza fiecare modul prelucrat la Interfata utilizator

- Configurare
  - Foloseste Interfata utilizator pentru setari preferinte
  - Primeste cereri de la alte module, pentru a determina actiunile in functie de preferintele utilizatorilor

Operaţii Server



1. serverul primeste o cerere

2. Paseaza la modulul de rezolutie a adresei care
(a) determina server-ul;
(b) determina daca cerere continut static / dinamic;
(c) examineaza credentiale autorizare.

3. Paseaza la modul procesare cerere, care apeleaza sub-module necesare

4. Rezultat pasat generatorului de raspuns
5. Pasat modulului suport retea
6. Transmite clientului

Rezolvarea adresei

- selecteaza virtual host
  - nu exista antet Host: -> eroare 400 Bad request
  - exista -> determina domeniul

    -> determina parametrii de config. logica (proprii virtual host)

    Obs. fragment din fisier configurare Apache

```
<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 folosind info de configurare logica

Alias	/test	/servlet/test
Alias	/images	/static/images

  - **http://www.ceva.com/test?a=1&b=2**

    /test -> /servlet/test

  - **http://www.ceva.com/images/nou.gif**

    /images -> /static/images

- mapeaza adresa
  - **pagina statica** – adauga calea la radacina serverului la calea din URL
    - URL **http://www.ceva.com/pagini/cucu.html**
    - in configurare **DocumentRoot** **/www/docs/ceva**
      - calea devine **->** **/www/docs/ceva/pagini/cucu.html**
  - **pagina dinamica** – este creata de un program
    - mecanismul folosit se determina pe baza:
      - **prefix cale URL** **/servlet/** target = servlet Java
        /cgi-bin/ target = CGI script
      - **sufix nume** **.cgi** **.php**
- verifica autentificare
  - **cod eroare daca resursa ceruta este protejata**

- Procesare cerere
  - regaseste continut (sau primeste de la programul care-l genereaza)
  - 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 (de ex. chunked)

HTTP/1.1 200 OK

Content-Type text/plain

Content-Transfer-Encoding: chunked

- Conexiunea persistenta
  - cozi de cereri si de raspunsuri – pastreaza ordinea:
    - raspunsurile din coada de raspunsuri pastreaza ordinea cererilor din coada de cereri

Functionare server

- server HTTP = set de thread-uri care proceseaza cererile clientilor

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

ServerName demo

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

ServerType Standalone	pastrat continuu in executie
Port 80	
KeepAlive On	suporta conexiuni persistente
MaxKeepAliveRequest 100	nr maxim cereri in asteptare
KeepAliveTimeout 15	taie conex. daca nu cerere noua in 15 sec
MaxRequestsPerChild 200	nr max cereri procesate fara repornire child
Timeout 300	timp maxim de procesare a unei cereri

Studiu individual

A. S. Tanenbaum Reţele de calculatoare, ed 4-a, BYBLOS 2003

7.3 WORLD WIDE WEB
- 7.3.1 Aspecte arhitecturale
- 7.3.2 Documente Web statice
- 7.3.3 Documente Web dinamice
- 7.3.4 HTTP – HyperText Transfer Protocol

A. S. Tanenbaum Computer networks, 5-th ed. PEARSON 2011

7.3 THE WORLD WIDE WEB
- 7.3.1 Architectural Overview
- 7.3.2 Static Web Pages
- 7.3.3 Dynamic Web Pages and Web Applications
- 7.3.4 HTTP—The HyperText Transfer Protocol