



Preddiplomski studij

Računarstvo

Modul:

Telekomunikacije i  
informatika

# Višemedijske usluge

Internetske usluge:  
World Wide Web – terminologija,  
formati i protokoli

Ak.god. 2007./2008.

10.04.2008.

- ♦ osnove World Wide Weba (*podsjetnik, Komunikacijske mreže*)
- ♦ nastanak World Wide Weba
- ♦ izvedba usluge u mreži i programska podrška
- ♦ pojam Uniform Resource Identifier (**URI**)
- ♦ zapis sadržaja na Webu (**HTML**)
- ♦ protokol Hypertext Transfer Protocol (**HTTP**)
- ♦ posrednički poslužitelji i priručna spremišta

- ◆ usluga: globalni hipermedijski informacijski sustav
  - informacijski prostor weba čine *informacijski izvori* ili *resursi* (engl. *resource*) međusobno povezani hipervezama (engl. *hyperlink*)
  
- ◆ model izvedbe usluge: klijent-poslužitelj
  
- ◆ osnovne komponente:
  - zapis izvora: **HTML (XHTML)**
    - jednostavan, prenosiv zapis teksta, mogućnost umetanja hiperveza, korištenje datoteka s drugim medijima (slike, audio, video) u izvornom obliku
  - adresiranje - identifikacija izvora: **URI**
  - način povezivanja i komunikacije: **HTTP**
    - standardni internetski aplikacijski protokol

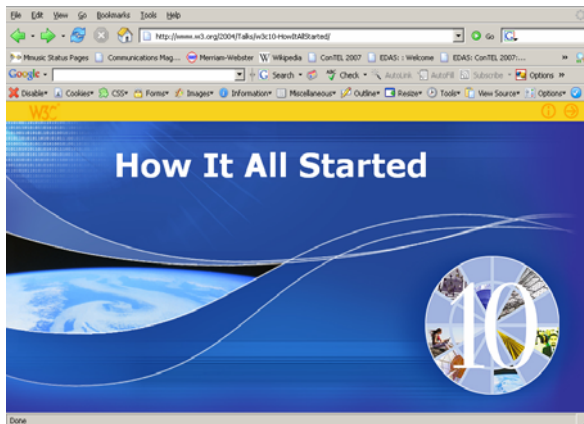
- ◆ pojam hipermedijskog dokumenta proširuje se pojmom *informacijskog izvora* ili *resursa* (engl. *resource*)
  - u općenitom smislu, “bilo što” što daje informaciju i što se može identificirati
  
- ◆ obično promatramo konkretne, automatizirane, mrežno dohvatljive informacijske izvore, npr.:
  - elektronički dokument,
  - slika,
  - izvor informacije jasne namjene (npr. tečaj HNB),
  - usluga (HTTP-SMS prilaz),
  - kolekcija resursa.
  
- ◆ primjer *izvora*: elektronički dokument (“datoteka”)
  - *informacija* koju datoteka pruža je njen *sadržaj* (može biti statički ili promjenjiv)
  - prikaz, odnosno *reprezentacija* informacije se često naziva “Web stranicom”

- ◆ “klasične” Internetske usluge
  - tematski web portali
  - pristup datotekama
  - e-mail, webmail, archive mailing lista
  - news, forumi, blogovi
  - ...
  
- ◆ ostale usluge
  - rezervacijski sustav (avio-karte, hoteli, ...)
  - digitalna knjižnica, on-line publikacije
  - osobno bankarstvo
  - studentska služba (upis, prijava ispita, ...)
  - ...


# Kako je sve započelo..., riječima izumitelja Web-a

- ♦ Tim Berners Lee – prezentacija na 10. godišnjici WWW-a

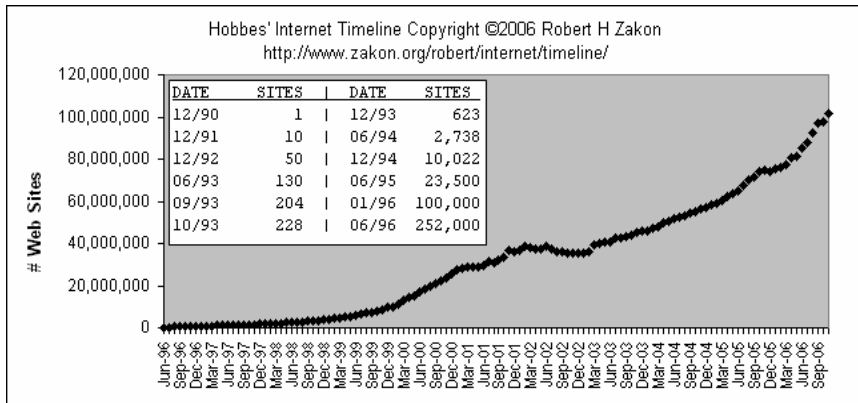
<http://www.w3.org/2004/Talks/w3c10-HowItAllStarted/>



Tim Berners-Lee, W3C  
Director and inventor of  
the World Wide Web

- ♦ prvi priznati autor ideje - Vannevar Bush, 1945.g.
  - **memex** – uređaj koji će proširiti ljudsku sposobnost pamćenja i rukovanja informacijama kroz poveznice među dokumentima  
("As We May Think", The Atlantic Monthly, July 1945)
  
- ♦ počeci realizacije današnjeg WWW-a: CERN  European Organization for Nuclear Research
  - ideja: omogućiti znanstvenicima iz raznih država jednostavan pristup raznim dokumentima vezanim za projekte
  - prvi prijedlog takvog sustava povezanog hipervezama - **Tim Berners-Lee**, 1989. g.
  
- ♦ prva javna demonstracija sustava (s tekstualnim sučeljem) 1991. g. na konferenciji *Hypertext '91*
- ♦ prvi preglednik s grafičkim sučeljem, Mosaic, u veljači 1993. g.
- ♦ u Hrvatskoj: počeci 1993. g., Croatian Home Page (kasnije, [www.hr](http://www.hr)) 1994. g.  
(<http://web.archive.org/web/19961106021324/http://tjev.tel.fer.hr/>)

# Rast broja Web-sjedišta (1990-2006)



Izvor: <http://www.zakon.org/robert/internet/timeline/#Growth>

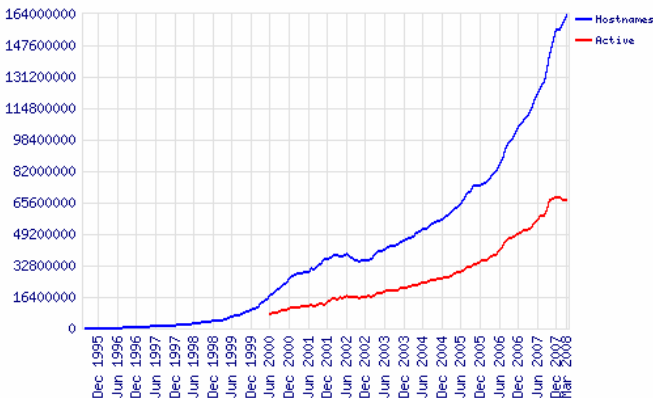


# Trend rasta broja Web-sjedišta (1995-2008)



Zavod za telekomunikacije

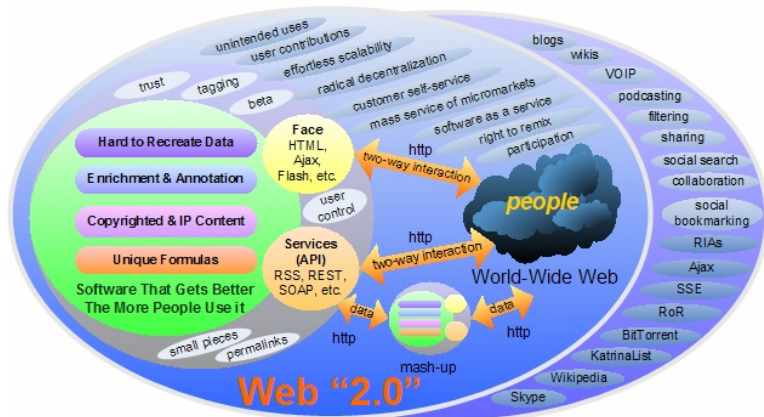
Total Sites Across All Domains August 1995 - March 2008



“In the **March 2008** survey, we received responses from **162,662,052** sites. Growth has continued to rise over the past few months, with this month seeing a gain of four and a half million new sites..”

Source: [http://news.netcraft.com/archives/web\\_server\\_survey.html](http://news.netcraft.com/archives/web_server_survey.html)

## Elements of the Web's Next Generation



Source: <http://web2.wsj2.com>

Tim Berners-Lee: "...Web 2.0 is of course a piece of jargon, nobody even knows what it means. If Web 2.0 for you is blogs and wikis, then that is people to people. But that was what the Web was supposed to be all along..."

<http://www-128.ibm.com/developerworks/podcast/dwi/cm-int082206.txt>

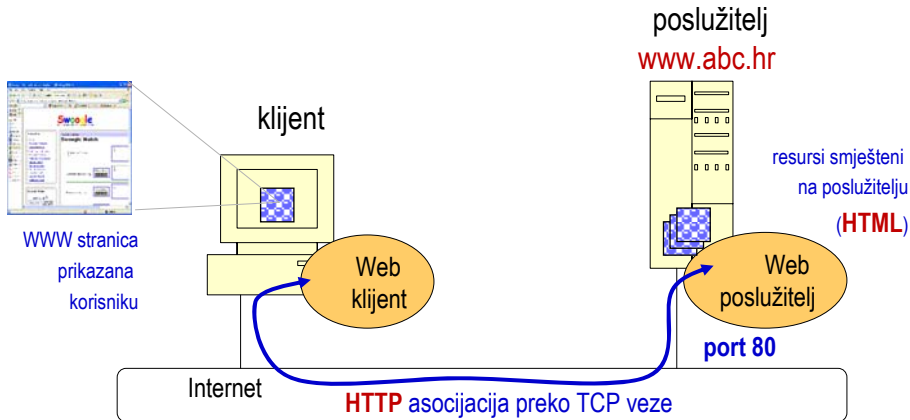
## Semantički Web

- ♦ *"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize."*  
– Tim Berners-Lee, 1999

### Preporučena literatura za zainteresirane:

- ♦ **W3C Semantic Web Activity** [On-line: <http://www.w3.org/2001/sw/>]
- ♦ T. Berners-Lee, Tim; J. Hendler, O.Lassila (May 17, 2001). "The Semantic Web". Scientific American Magazine. [On-line: <http://www.sciam.com/article.cfm?id=the-semantic-web>]
- ♦ T. Berners-Lee, intervju 7.2.2008.  
[On-line: [http://talis-podcasts.s3.amazonaws.com/twt20080207\\_TimBL.html](http://talis-podcasts.s3.amazonaws.com/twt20080207_TimBL.html)]

- ◆ model klijent-poslužitelj
- ◆ resurs identificiran putem **URI**



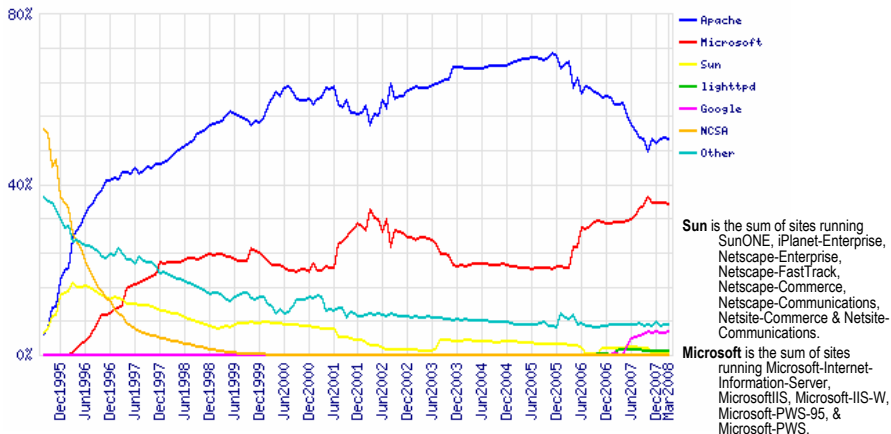
## ♦ Web **klijent**

- korisnički Web klijent – **preglednik** (engl. *browser*)
  - grafičko ili tekstualno korisničko sučelje za prikaz Web stranice i navigaciju; novije verzije donose više mogućnosti
  - popularni preglednici: Netscape, Mozilla, Firefox, Internet Explorer, Opera, Lynx, ... (uglavnom besplatni)
- automatizirani Web klijent – **robot** ili pauk (engl. *spider, crawler*)
  - program koji samostalno pretražuje Web (ili neki njegov dio) radi prikupljanja podataka, npr. za tražilice

## ♦ Web **poslužitelj**

- popularni HTTP poslužitelji: Apache HTTP server (besplatan), Microsoft Internet Information Server
- dodatni aplikacijski poslužitelji

## Market Share for Top Servers Across All Domains August 1995 - March 2008



Source: [http://news.netcraft.com/archives/web\\_server\\_survey.html](http://news.netcraft.com/archives/web_server_survey.html)



Source: <http://marketshare.hitslink.com/report.aspx?qprid=0>

◆ Prva 3 zajedno > 97%

- Microsoft Internet Explorer 74,88%
- Firefox 17,27%
- Safari 5,70%

- ◆ usklađivanje Web stranica
  - Cascading Style Sheets (**CSS**)
  - Server Side Includes (**SSI**)
  
- ◆ klijentske tehnologije
  - **Javascript**
  - Java apleti (*applets*)
  - za tehnologije koje nisu “ugrađene” koriste se *plug-in*-ovi
  
- ◆ poslužiteljske tehnologije
  - Common Gateway Interface, **CGI**
  - Java **servleti**, JavaServer Pages (**JSP**)
  - Active Server Pages (**ASP**) - Microsoft
  - Perl, PHP



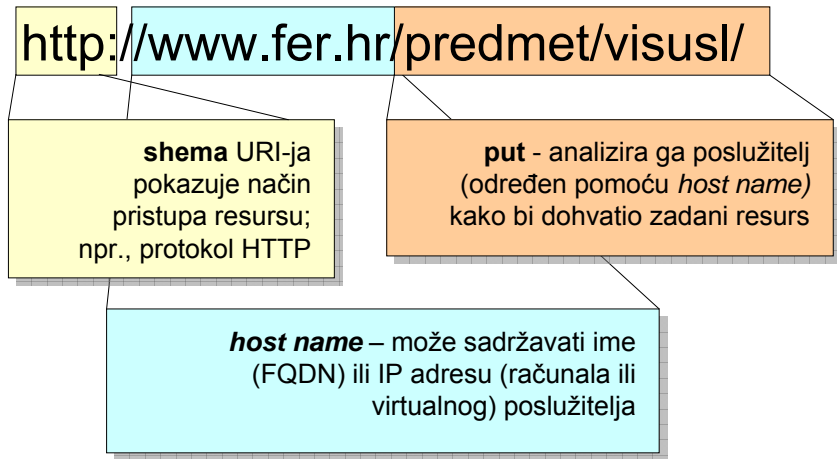
- identifikacija izvora: **URI** ◀
- zapis izvora: HTML (XHTML)
- način povezivanja i komunikacije: HTTP

## URI – Uniform Resource Identifier

*(uniformni identifikator resursa)*

- ♦ **uniformni**: jednoobrazni način zapisa – propisan je oblik
- ♦ **identifikator**: sadrži informaciju nužnu za razlikovanje identificiranog resursa od svih ostalih ( $\neq$  identitet!)
- ♦ **resurs**: informacijski izvor; “*bilo što*” što se može identificirati URI-jem

*Pojam URI-ja je središnji pojam u arhitekturi World-Wide Weba. World Wide Web Consortium (W3C) definira WWW kao “informacijski prostor u kojem su predmeti od interesa identificirani URI-jima”.*



- ◆ Hijerarhijski niz komponenata:
  - **shema** (engl. *scheme*)
  - **autoritet** (engl. *authority*)
  - **put** (engl. *path*)
  - **upit** (engl. *query*)
  - **fragment** (engl. *fragment*).

URI = **scheme** ":" hier-part [ "?" **query** ] [ "#" **fragment** ]

hier-part = "://" **authority** path-abempty  
/ path-absolute  
/ path-rootless  
/ path-empty

- ◆ svaki apsolutan URI je složen ovako:

**<shema>** : *<dio specifičan za shemu>*

- ◆ **shema** je zaseban potprostor imena
  - npr. “**http**”, “**ftp**”, “**mailto**”, “**urn**”, “**file**”, “**news**”
  - neki su dobili ime po protokolu, ali to **ne znači da je shema = protokol**
- ◆ sintaksa ostatka URI-ja posebno se definira za svaku shemu

<http://www.fer.hr/predmet/kommre/>

<http://www.w3.org/TR/webarch/#identification>

<http://www.hr/wwwhr/arts/theatre/index.hr.html>

<http://google.com/search?q=telematika>

<mailto:telemat@tel.fer.hr>

<file:///c:/temp/>

<news:hr.org.fer>

<ftp://jdoe:jdoe@ftp.w3.org/>

<about:blank>

<urn:ietf:rfc:2396>

- ♦ većina shema *<dio specifičan za shemu>* ustrojava ovako:

// *<autoritet>* *<put>* ? *<upit>*

- ♦ **autoritet** – predstavlja (logičkog) poslužitelja
- ♦ **put** – put kroz hijerarhiju (po uzoru na datotečni sustav)
  - počinje kosom crtom i nadalje je sastavljen od segmenata odvojenih kosom crtom, npr:

/

/segment1/2/3

/mark-twain/roman/tom-sawyer

- ♦ **upit** – popis parametara u proizvoljnom redoslijedu

- ♦ shema *http* definira autoritet ovako:

$\langle \text{autoritet} \rangle = \langle \text{host} \rangle : \langle \text{vrata} \rangle$

- ♦ primjer:

`http://www.fer.hr/?@=1d2w9#news_8980`

`http://www.google.com:81/search?q=telematika`

- ♦ broj **vrata** je neobavezan dio autoriteta (podrazumijeva se tcp/80)
- ♦ **put** neobavezan
- ♦ **upit** iza prvog upitnika, neobavezan
  - upit može imati više segmenata
  - segmenti upita se odvajaju znakom **&**
  - svaki segment je tipično par (ime, vrijednost) odvojen znakom **=**



- ◆ URI-ju se može dodati na kraj identifikator fragmenta entiteta odvojen znakom “#”, npr.:

`http://www.hr/hrvatska/general.hr.shtml#territory`

- u HTML dokumentu “/general.hr.shtml” postoji oznaka s nazivom “territory” na čiji se vrh pozicionira prozor preglednika
- identifikator fragmenta je smislen samo ako se koristi pri akciji dohvaćanja entiteta u kojem se nalazi odgovarajući fragment

URI = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

hier-part = "://" authority path-abempty

/ path-absolute

/ path-rootless

/ path-empty

http: //example. com: 8042/over/there?name=ferret#nose

\_/\	\_____/\	\_____/\	\_____/\	\_/\
shema	authori tet	put	up i t	fragment

- ♦ URI izravno koristi znakove iz ograničenog skupa
- ♦ ostali ASCII znakovi se predstavljaju u posebnom “escaped” obliku: znak “%” i dvoznamenkasti heksadecimalni kôd

povećanje od +30%

- “ć” ima kôd E6, razmak 20, a “%” 25

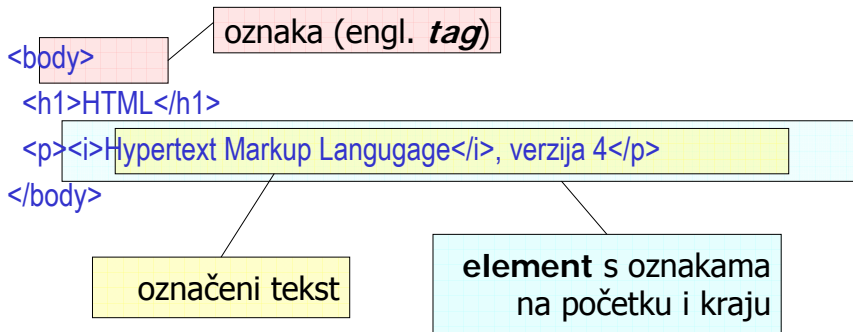
pove%E6anje%20od%20+30%25

- kod **upita** postoji poseban način pretvaranja razmaka u “+” (kôd 2B):

pove%E6anje+od+%2B30%25

- identifikacija izvora: URI
- zapis izvora: **HTML (XHTML)** ◀
- način povezivanja i komunikacije: HTTP

- ♦ prva verzija HTML-a 1992. godine; verzija 4.01 iz 1999. (preporuka W3C-a), osnovica za *Extensible Hypertext Markup Language* XHTML
- ♦ jezik za označavanje (*markup*) – običan tekst s umetnutim oznakama koje utječu na predočavanje teksta i služe za uvođenje hiperveza



```
<html>
```

**DOKUMENT**

```
<head>
```

```
<title>TU: HTML: ustroj dokumenta</title>
```

```
<meta name="author" content="Ivo Ivic">
```

```
</head>
```

**ZAGLAVLJE**

```
<body>
```

```
<h1>Ustroj dokumenta u HTML-u</h1>
```

```
<p>HTML dokument sa sastoji od <b>zaglavljja</b> i
```

```
<b>tijela</b>.</p>
```

```
</body>
```

**TIJELO**

```
</html>
```

<html>

<head>

<title>TU: HTML: ustroj dokumenta</title>

<meta name="author" content="Ivo Ivic">

</head>

<body>

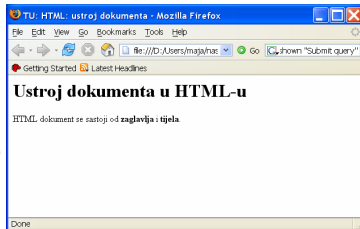
<h1>Ustroj dokumenta u HTML-u</h1>

<p>HTML dokument sa sastoji od <b>zaglavlja</b> i

<b>tijela</b>.</p>

</body>

</html>



- ♦ organizacija dokumenta
  - odjeljak (paragraf) **<p>**
  - naslov poglavlja (*heading*) **<h1>** **<h2>** ...
- ♦ izgled slova
  - podebljanje (*bold*): **<b>**
  - kurziv (*italic*): **<i>**
- ♦ hiperveza (poveznica): **<a href="url">tekst</a>**
- ♦ umetnuta slika: ****
- ♦ tablice: **<table>** **<tr>** **<td>** ...



```
<form action="http://google.com/search" method=GET>
```

Pretraživanje izraza:   
<p>

Broj rezultata na stranici:

```
<select name="num">
```

```
<option value="10" selected>10
```

```
<option value="20">20
```

```
<option value="50">50
```

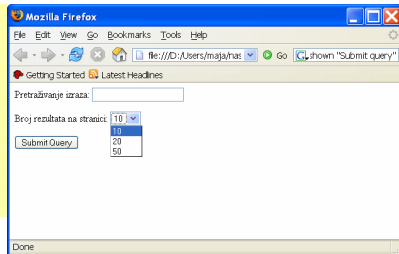
```
</select><p>
```

```
<input type=submit
```

```
name="search"
```

```
value="Pretraži Web!">
```

```
</form>
```



- ◆ često primjenjivani formati dokumenata (na webu i drugdje)
  - umetnute slike: GIF, JPEG, PNG
  - dokumenti: PDF, Postscript
  - multimedijски dodaci: MPEG, QuickTime, WM
  - ...
  
- ◆ razni formati zasnovani na jeziku Extensible Markup Language (XML)
  - XHTML - HTML zapisan pomoću XML-a
  - vektorska grafika: SVG (Scalable Vector Graphics)
  - multimedijске prezentacije: SMIL (Synchronized Multimedia Integration Language)
  - unos elektroničkim perom: Ink Markup Language (InkML)
  - ....

- ♦ standard W3C-a od 1996., CSS1, CSS2.1, CSS 3
- ♦ <http://www.w3.org/Style/CSS/>
- ♦ omogućuje definiranje stilova prikaza
  - izgled slova, boje, organizacija dokumenta
- ♦ CSS ima stupnjeve (engl. *level*) i profile (engl. *profile*)
  - preglednici na stolnim osobnim računalima imaju izvedbu stupnja 1, 2 ili 3; drugi programi imaju izvedbu odgovarajućeg profila (npr. mobitel, PDA, TV, printer, itd.)
- ♦ definicije se čuvaju u jednom dokumentu (*style sheet*), a koriste u više HTML dokumenata
  - npr. stil “komentar” korišten u HTML-u:  
`<p class="komentar">Koristimo CSS</p>`
  - osiguravaju jedinstven izgled svih stranica u sjedištu
  - stilovi se mogu nasljeđivati i nadjačavati (*cascading*)
  - donekle rješava problem odvajanja izgleda od sadržaja

- ◆ tehnički problemi sintakse
  - tolerancija greški, loše performanse, *namespace*-i
- ◆ pomiješan sadržaj sa izgledom
  - teško održavanje (izmjena sadržaja ili izgleda nezavisno od drugog sastojka)
  - pruža samo jedan “pogled” na podatke
- ◆ orijentiran na prikaz
  - nedostatak semantike
  - problematično pretraživanje
- ◆ nedosljedna podrška preglednika
  - odstupanje od standarda
- ◆ rješenje: Extensible HyperText Markup Language (XHTML™)

- ◆ *Extensible Hypertext Markup Language* (XHTML™)
  - <http://www.w3.org/MarkUp/>
  - prva verzija XHTML 1.0, 1/2000, revizija 8/2002
  - u izradi verzija XHTML 2.0
  
- ◆ XHTML zapisuje HTML pomoću XML-a
  - XML = *Extensible Markup Language*
  - novi, univerzalni standard formatiranja na Webu
  - ima strožu sintaksu → brže se parsira → brže se prikazuje u pregledniku
  - proširiv je – korisnik može dodati svoje oznake (“tagove”)
  
- ◆ moguć problem: nedostatak podrške u preglednicima  
<http://www.w3.org/People/mimasa/test/xhtml/media-types/results>
  
- ◆ novi standard za obrasce: **XForms**

- ◆ najvažnije: zapis u XML-u
- ◆ osjetljivost na velika i mala slova
  - sva imena *tagova* moraju biti u malim slovima
- ◆ obavezni zatvarajući *tagovi*
- ◆ svi atributi moraju imati vrijednost
- ◆ vrijednost atributa mora biti u navodnicima
- ◆ nema križanja elemenata (`<b><i>!!!</b></i>`)

- identifikacija izvora: URI
- zapis izvora: HTML (XHTML)
- način povezivanja i komunikacije: **HTTP** ◀

- ◆ internetski protokol aplikacijskog sloja
- ◆ definira format i način razmjene poruka
  - tekstualan zapis, sličan formatu e-mail poruke i MIME standarda
- ◆ vrste poruka:
  - **zahtjev** ("metoda")  
definira operaciju (metodu), resurs, protokol  
naziv "metoda" potječe od terminologije iz područja objektno-orijentiranog programiranja
  - **odgovor** (ishod zahtjeva i rezultat)  
ishod zahtjeva (uspjeh, neuspjeh, greška,...) opisan statusnim kôdom  
za neke vrste zahtjeva, kao rezultat uspješnog ishoda, u tijelu odgovora dostavlja se sadržaj zatraženog resursa



# Komunikacija HTTP klijenta i poslužitelja



Zavod za telekomunikacije

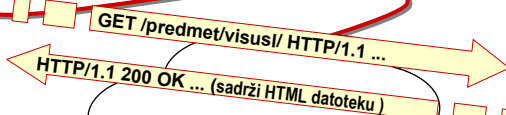
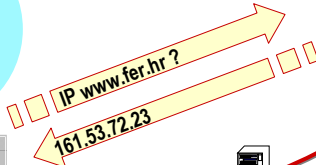
Odabrana "web adresa" (URI):  
<http://www.fer.hr/predmet/visusi>



web klijent



lokalni DNS  
poslužitelj



(dohvaćanje  
datoteke s diska,  
ili dinamičko  
generiranje  
datoteke)



www.fer.hr  
161.53.72.23



disk

web  
poslužitelj

1. proces www poslužitelja (uvijek) osluškuje TCP zahtjeve na dobro-poznatim vratima 80 (ako nije drugačije konfiguriran!)
2. koristeći klijentski program (preglednik), korisnik upisuje adresu traženog izvora (**URI**)
3. preglednik saznaje IP adresu poslužitelja putem upita na DNS
4. preglednik pokreće TCP vezu sa slobodno odabranih vrata na lokalnom računalu na IP adresu poslužitelja i TCP vrata 80 (port je “dobro-poznat”)
5. nakon uspostave TCP veze, preglednik zahtijeva da mu poslužitelj pošalje dokument (**HTTP-zahtjev**)
6. poslužitelj šalje dokument(e) (**HTTP-odgovor**)
7. nakon uspješnog transfera, TCP veza se zatvara
8. preglednik prikazuje dokument (**HTML**) korisniku

- ◆ prva verzija "0.9" - ograničene mogućnosti
  - podržava prijenos samo hipertekstualnih dokumenata
  
- ◆ **HTTP 1.0** - prvi (*Informational*) RFC (RFC 1945), svibanj 1996.
  - podržava prijenos različitih tipova podataka
    - posuđuje koncepte iz MIME standarda
  - zadržava kompatibilnost unazad
  - sredinom 90-tih - HTTP promet dominira - HTTP/1.0 neučinkovit
    - svako sjedište mora biti na drugom poslužitelju
    - preko jedne konekcije ostvaruje se jedan HTTP zahtjev
    - nema podrške za upravljanje performansama - cache, proxy, djelomični dohvat
  - WWW prozvan "World Wide Wait"

- ◆ **HTTP verzija 1.1** - RFC 2616, lipanj 1999.
  - zadržava kompatibilnost unazad prema HTTP 1.1
  - RFC 2617 - autentifikacija i sigurnost
  
- ◆ **poboljšanja:**
  - jedno fizičko računalo - više Web poslužitelja - "virtualni host"
  - trajne konekcije - više zahtjeva preko jedne TCP konekcije
  - djelomični dohvat sadržaja
  - bolja podrška za priručna spremišta (engl. *cache*) i posrednike (engl. *proxy*)
  - pregovaranje o sadržaju datoteke (engl. *content negotiation*)
  - bolji sigurnosni mehanizmi - autentifikacija

- ◆ zahtjev i odgovor moraju biti ispravno formatirani
  
- ◆ HTTP definira opći format poruke
  - tekstualan zapis (kao SMTP)
  - naslanja se na format e-mail poruke (RFC 822) i MIME standarda
    - dijele neka načela, ali ne sasvim i ne potpuno
    - npr. ne koriste se sva MIME zaglavlja
    - npr. tijelo ne mora biti 7-bitni ASCII

## zahtjev

GET /predmet/visusi HTTP/1.1

Host: www.fer.hr

...

Accept-Language: hr, en

Accept-Encoding: gzip, deflate

...

## odgovor

HTTP/1.1 200 OK

Date: Mon, 07 Apr 2008 17:31:09 GMT

Server: Apache/2.2.8 (FreeBSD) ..

Last-Modified: Mon, 30 Jan 2006 16:12:36 GMT

...

Keep-Alive: timeout=3, max=61

Connection: Keep-Alive

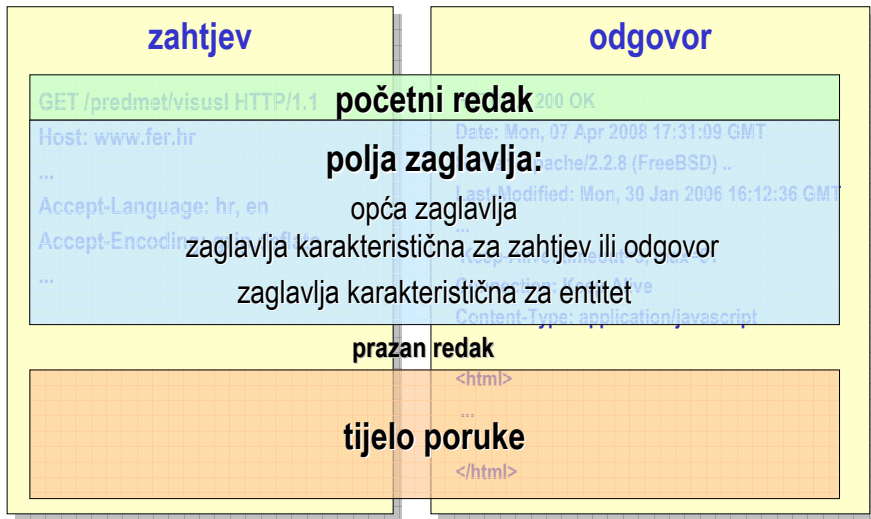
Content-Type: application/javascript

<html>

...

...

</html>



- ♦ početni redak sadrži (*request line*):
  - nad kojim resursom je podnesen zahtjev
  - koja metoda (operacija) se traži nad tim resursom
  - koja se verzija protokola koristi

<metoda> <URI> <verzija>

- ♦ primjeri:
  - GET / HTTP/1.0
  - POST /shop/order HTTP/1.1
  - HEAD /search?q=raspored HTTP/1.0



- ◆ metoda zahtjeva određuje što se traži od resursa
- ◆ HTTP/1.1 definira 8 metoda i omogućuje dodavanje novih metoda (*extensions*):
  - OPTIONS
  - GET
  - HEAD
  - POST
  - PUT
  - DELETE
  - TRACE
  - CONNECT
- ◆ najpoznatije metode: **GET, HEAD i POST**
- ◆ važna svojstva: sigurnost i idempotentnost

- ♦ **GET** – metoda za dohvaćanje
  - znači “Pribavi reprezentaciju tog resursa”
  - aktivira se kod upisivanja adrese u preglednik ili klika na link
- ♦ ako poslužitelj ima zahtjevani resurs, vraća ga u tijelu odgovora, inače vraća grešku
- ♦ ponašanje metode GET se mijenja ako se koristi uvjetni GET (zaglavlje *If-Modified-Since:* ili *If-Match:*)
- ♦ moguće koristiti djelomični GET (partial GET) - dohvaća se samo dio datoteke (u zaglavlju zahtjeva definiran raspon *Range*)

GET / HTTP/1.0

GET /a/b?c HTTP/1.1

Host: www.tel.fer.hr

**prazan redak, bez tijela**

GET /obicni.txt HTTP/1.1

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,  
application/vnd.ms-excel, application/vnd.ms-powerpoint,  
application/msword, \*/\*

Accept-Language: hr

Accept-Encoding: gzip, deflate

User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

Host: www.tel.fer.hr

Connection: Keep-Alive

- ◆ **HEAD** – metoda za dohvaćanje podataka o resursu
  - razlika u odnosu na GET: poslužitelj **ne** vraća sadržaj resursa u tijelu odgovora
  - najčešće se koristi kako bi se provjerilo postoji li entitet na poslužitelju
- ◆ druge uporabe:
  - provjera veličine datoteke prije dohvaćanja
  - pribavljanje metapodataka o entitetu
- ◆ na poslužitelju se HEAD zahtjev obrađuje jednako kao i GET

- ◆ **POST** – metoda za “aktiviranje” resursa
  - znači “Pomoću adresiranog resursa obradi podatke koje šaljem”
  - obično se aktivira se pritiskom na gumb u obrascu
  
- ◆ može se koristiti kod ispunjavanja web-obrasca
  - podaci koje je upisao korisnik, prenose se metodom POST na poslužitelj i tamo se obrađuju
    - primjeri:
      - zahtjev za provedbom narudžbe
      - dodavanje vlastitog komentara tekstu na Webu
      - prijavljivanje na termin laboratorijskih vježbi
  - u ovom primjeru, koristio bi se gumb “*Submit*” na obrascu

## **POST /search HTTP/1.1**

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg...

Accept-Language: hr

## **Content-Type: application/x-www-form-urlencoded**

Accept-Encoding: gzip, deflate

User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

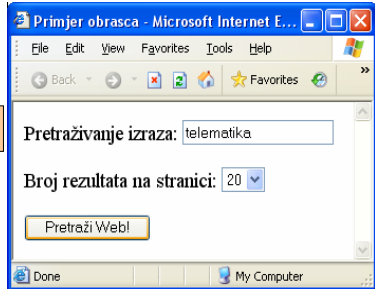
Host: localhost:4744

## **Content-Length: 44**

Connection: Keep-Alive

Cache-Control: no-cache

q=telematika&num=20&search=Pretra%9Ei+Web%21



## ◆ OPTIONS

- informiranje o mogućnostima resursa i poslužitelja (URI = \*)

## ◆ PUT

- postavljanje entiteta uključenog u tijelu zahtjeva na zadani URI
  - POST u URI navodi resurs koji mora obraditi podatke koji se šalju u tijelu zahtjeva
  - PUT navodi naziv resursa u kojeg treba pohraniti podatke koji se šalju u tijelu
- najčešće se ne koristi - sigurnosni rizik

## ◆ DELETE – brisanje odabranog resursa - ne koristi se

## ◆ TRACE – klijent dobiva kopiju zahtjeva kojeg je uputio poslužitelju, služi za dijagnostiku

## ◆ CONNECT - za buduću uporabu, ne implementira se

- ◆ općenita zaglavlja
  - odnose se na poruku, a ne na njen sadržaj
  - mogu se pojaviti i u zahtjevu i u odgovoru
  
- ◆ zaglavlja karakteristična za zahtjev
  - poslužitelju daju više informacija o prirodi zahtjeva
  - omogućuju klijentu da kontrolira kako će se zahtjev ostvariti
    - npr. uvjetni zahtjev - samo ako su ispunjeni određeni uvjeti
  - klijent obavještava poslužitelja koje vrste podataka može obraditi
  
- ◆ zaglavlja karakteristična za entitet
  - opisuju entitet koji se nalazi u tijelu poruke, ako ga ima



## METHOD:

- GET
- POST
- HEAD
- PUT
- OPTIONS
- TRACE
- CONNECT
- DELETE

**Method Request-URI HTTP-Version**  
**parametar1: *vrijednost***  
**parametar2: *vrijednost***  
**parametar3: *vrijednost***  
...  
**<prazni redak>**

**HTTP/1.0**  
**HTTP/1.1**

## general\_header

Cache-Control:  
Connection:  
Date:  
Pragma:  
Trailer:  
Transfer-Encoding:  
Upgrade:  
Warning:

## request\_header

Accept:	If-Modified-Since:
Accept-Charset:	If-None-Match:
Accept-Encoding:	If-Range:
Accept-Language:	If-Unmodified-Since:
Authorization:	Max-Forwards:
Expect:	Proxy-Authorization:
From:	Range:
Host:	Referer:
If-Match:	TE:
	User-Agent:

## entity\_header

**Allow:**  
Content-Encoding:  
Content-Language:  
Content-Length:  
Content-Location:  
Content-MD5:  
Content-Range:  
Content-Type:  
Expires:  
Last-Modified:  
**extension-header**

- ◆ **početni redak** sadrži:

- verziju protokola
- statusni kôd
- opisnu frazu

HTTP/1.1 303 See Other

HTTP/1.1 200 OK

HTTP/1.1 404 Not Found

- ◆ u **tijelu** odgovora se obično prenosi reprezentacija resursa (“entitet”) koju preglednik treba prikazati korisniku
- ◆ neka polja zaglavlja:
  - **Content-Type**: format entiteta
  - **Content-Length**: duljina entiteta u tijelu u oktetima

- ◆ opća zaglavlja
  - odnose se na poruku, a ne na njen sadržaj
  - mogu se pojaviti i u zahtjevu i u odgovoru
  
- ◆ zaglavlja karakteristična za odgovor
  - klijentu daju više informacija o odgovoru
    - dio informacija može se prenijeti i u tijelu poruke (npr. opis pogreške)
  
- ◆ zaglavlja karakteristična za entitet
  - opisuju entitet koji se nalazi u tijelu poruke, ako ga ima
  - češće se koriste kod odgovora nego kod zahtjeva

- ◆ sastoji se od tri dekadске znamenke
- ◆ slično kao kod protokola FTP i SMTP
  
- ◆ pet kategorija:
  - 1xx – **Informativne** - ne naznačuju ni uspjeh, ni neuspjeh
  - 2xx – **Uspjeh** - poslužitelj je primio, razumio i ispunio zahtjev
  - 3xx – **Preusmjeravanje** - potrebno poduzeti dodatne akcije
  - 4xx – Greška na **klijentu** - zahtjev je neispravan
  - 5xx - Greška na **poslužitelju** - zahtjev je ispravan, ali poslužitelj ga ne može ispuniti

**HTTP/1.0**  
**HTTP/1.1**

**HTTP-Version Status-Code Reason-Phrase**

parametar1: vrijednost

parametar2: vrijednost

parametar3: vrijednost

....

<prazna linija>

<html>

- tijelo dokumenta -

</html>

## **general\_header**

Cache-Control:

Connection:

Date:

Pragma:

Trailer:

Transfer-Encoding:

Upgrade:

Warning:

## **response\_header**

Accept-Ranges:

Age:

ETag:

Location:

Proxy-Authenticate:

Retry-After:

Server:

Vary:

WWW-Authenticate:

## **entity\_header**

Allow:

Content-Encoding:

Content-Language:

Content-Length:

Content-Location:

Content-MD5:

Content-Range:

Content-Type:

Expires:

Last-Modified:

**extension-header**

## **Informacija**

100 Continue

## **Uspjeh**

200 OK

## **Preusmjerenje**

300 Multiple Choices

301 Moved permanently

302 Found

304 Not Modified

## **Greška kod klijenta**

400 Bad Request

401 Unauthorized

403 Forbidden

404 Not Found

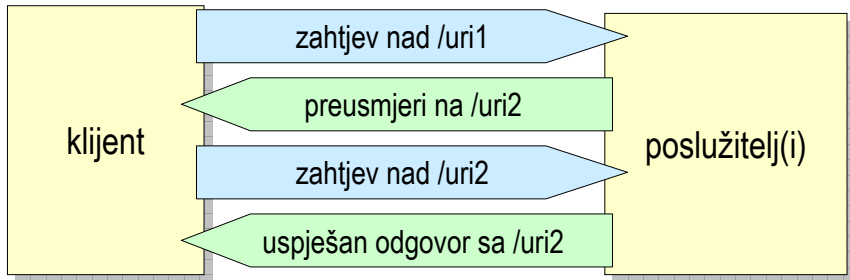
## **Greška na poslužitelju**

500 Internal Server Error

- ◆ poslužitelj je uspješno primio, razumio i ispunio zahtjev
- ◆ najčešći kôd **200** (*OK*)
  - za zahtjev **GET** znači da je dostavljeni entitet u tijelu odgovora sadržaj resursa
  - za **POST** znači da je resurs primio podatke i dostavljeni entitet opisuje ishod akcije
- ◆ najčešći odgovor na Webu

- ◆ 201 (Created) - rezultat metode PUT
- ◆ 204 (*No Content*)
  - poslužitelj je ispunio zahtjev, ali nema potrebe da vraća ikakav entitet u tijelu odgovora (npr. anketa)
  - korisnički agent ne mijenja prikaz prošlog entiteta
- ◆ 205 (*Reset Content*)
  - korisnički agent treba sadržaj postojećeg entiteta postaviti na početne vrijednosti
  - namijenjeno za višestruki unos podataka preko obrasca
- ◆ 206 (*Partial Content*)
  - vraćena je djelomična manifestacija resursa
  - zahtjev je sadržavao polje zaglavlja Range:

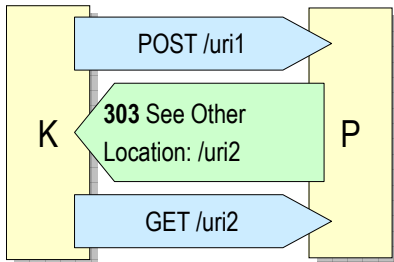
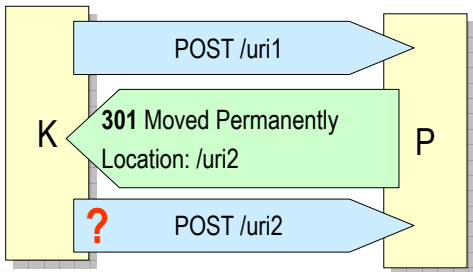
- ◆ klijent treba poduzeti dodatne korake kako bi ispunio izvorni zahtjev
- ◆ novi URI se nalazi u polju zaglavlja **Location**:
- ◆ sigurne metode se mogu izvršiti bez sudjelovanja korisnika
  - sigurne: GET, HEAD, OPTIONS, TRACE





- ◆ ponekad se resursu dodjeli drugi URI, pa se poslužitelj može konfigurirati da preusmjerava zahtjeve na novu adresu
- ◆ 301 (*Moved Permanently*)
  - resurs je trajno premješten i svi bi budući zahtjevi trebali biti usmjereni na novi URI
- ◆ 302 (*Found*)
  - resurs privremeno koristi drugi URI, klijent i dalje može koristiti stari URI
- ◆ 307 (*Temporary Redirect*)
  - resurs je samo privremeno premješten, pa se treba sačuvati stara adresa
- ◆ klijent ponavlja identičan zahtjev nad novim URI-jem
- ◆ ako metoda nije sigurna, traži se potvrda korisnika
- ◆ u tijelu odgovora se može nalaziti poruka za “ručno” preusmjeravanje

- ♦ zahtjev je ispunjen, ali sada treba prikazati sadržaj nekog drugog resursa (treba izvršiti **GET** nad resursom identificiranom u polju **Location**)
- ♦ budući da je GET siguran, preusmjeravanje se vrši automatski



- ◆ **uvjetni** (*conditional*) GET
- ◆ koristi polja zaglavlja
  - If-Modified-Since:
  - If-Unmodified-Since:
  - If-Match:
  - If-Unmatch:
- ◆ **304** (*Not Modified*)
  - traženi resurs nije promijenio sadržaj od zadnjeg zahtjeva
  - klijent može koristiti kopiju entiteta iz *cachea*

- ◆ namijenjena za slučajeve kad se čini da je pogreška nastupila na klijentovoj strani
- ◆ odgovor treba sadržavati poruku namijenjenu korisniku u kojem se opisuje situacija i nude rješenja, npr:
- ◆ **400** (*Bad Request*)
  - pogreška u sintaksi zahtjeva
- ◆ **401** (*Unauthorized*)
  - zahtjevu nedostaje autorizacija korisnika
- ◆ **404** (*Not Found*)
  - resurs nije dostupan, ali se ne ulazi u detalje zašto
  - obično je pogreška pri utipkavanju URI-ja

- ◆ **401** (*Unauthorized*)
- ◆ zahtjevu nedostaje autorizacija korisnika
- ◆ odgovor sadrži izazov klijentu u polju **WWW-Authenticate:**
- ◆ klijent ponavlja zahtjev sa dodanim poljem **Authorization:** koje sadrži npr. šifrirano korisničko ime i zaporku
- ◆ poslužitelj izvršava zahtjev ako on sadrži potrebnu autorizaciju
- ◆ nije isto što i autorizacija preko Cookieja (ZZT Web)!

## ♦ 403 (Forbidden)

- poslužitelj odbija ispuniti taj zahtjev, bez obzira na autorizaciju
- primjer: klasični poslužitelj nema pristup datoteci

## ♦ 405 (*Method Not Allowed*)

- nije dozvoljena tražena metoda nad tim resursom

## ♦ 410 (*Gone*)

- resurs na tom URI-ju nije više dostupan i neće ni biti, pa je uzaludno ponavljati zahtjeve

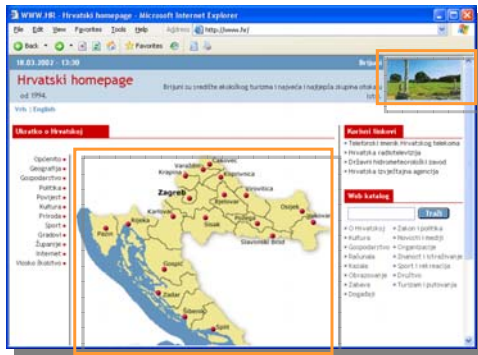
- ◆ kod ovakvih odgovora poslužitelj je “svjestan” da zahtjev nije ispunjen zbog njegove greške
- ◆ **500** (*Internal Server Error*)
  - obično programska greška u resursu
- ◆ **503** (*Service Unavailable*)
  - poslužitelj je preopterećen
  - odgovor može sadržavati **Retry-After**: polje

- ◆ HTTP-ov mehanizam izbora odgovarajuće prezentacije sadržaja resursa
- ◆ klijent predlaže prihvatljive oblike
  - polja zaglavlja: **Accept**, **Accept-Charset**, **Accept-Encoding**, **Accept-Language**, **User-Agent**
- ◆ primjer:
  - resurs podržava višejezične reprezentacije, uključujući i onu na hrvatskom
  - klijent predlaže **Accept-Language: hr**
  - poslužitelj vraća entitet na hrvatskom jeziku



- ◆ HTTP/1.1 omogućuje postojane konekcije (engl. *persistent connection*)
  - **Connection: Keep-Alive**
- ◆ dva načina korištenja konekcije:
  - *pipelining* – pošalju se svi zahtjevi i tek onda se čekaju odgovori
  - *chaining* – novi zahtjev se šalje tek nakon primitka odgovora

# Primjer uporabe *pipelininga*



(1) <http://www.hr>

za prikaz su potrebni 2, 3 i 4

(2) <http://www.hr/style.css>

(3) <http://www.hr/hrvatska/image/hrvatska.gif>

(4) <http://www.hr/hrvatska/image/banners/brijuni1.jpg>

K

P

GET /

200 OK ... (1)

GET (2)

GET (3)

GET (4)

200 OK ... (2)

200 OK ... (3)

200 OK ... (4)

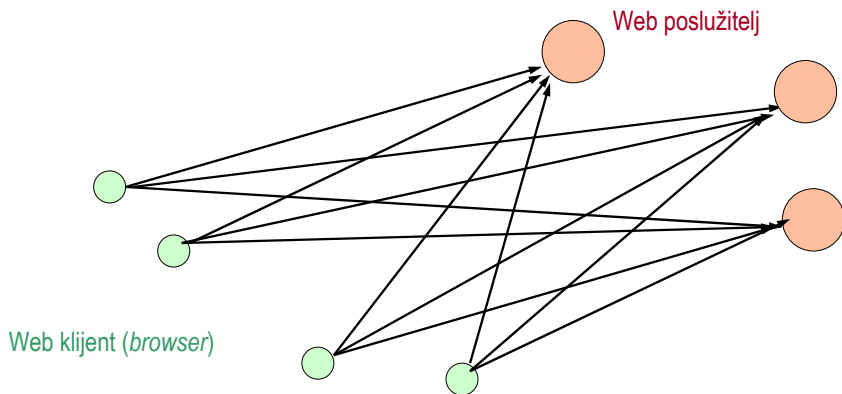
## **World Wide Web: posrednički poslužitelji i priručna spremišta**

- ◆ parametri kvalitete Web usluga
  - raspoloživost
  - propusnost
  - vrijeme čekanja
  
- ◆ pristupi poboljšanju performansi
  - **povećanje kapaciteta** u infrastrukturi – povećava se propusnost i raspoloživost
  - **uravnotežavanje opterećenja** – povećava se propusnost, smanjuje vrijeme čekanja
  - **uvodenje priručnih spremišta** (engl. *cache*) – povećava se propusnost, smanjuje vrijeme čekanja
  
- ◆ postoje i softverska i hardverska rješenja

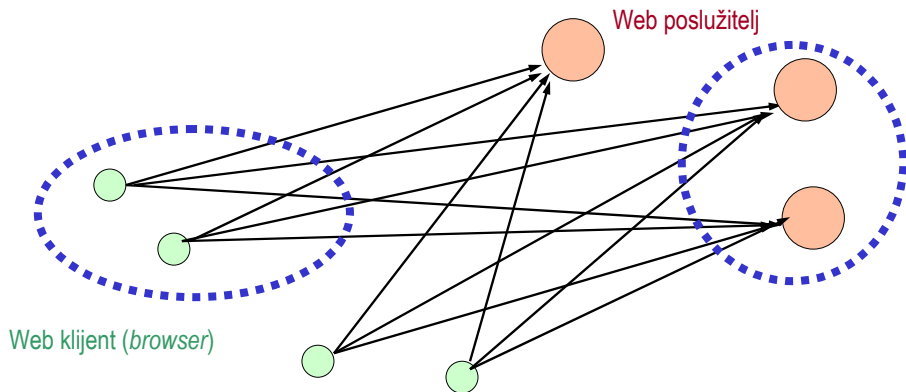
- ◆ **eksplicitna podjela sadržaja** među poslužiteljima
  - statički:
    - svaka usluga na drugom poslužitelju (\*.mail.com, \*.shop.com,...)
  - dinamički
    - osnova za mreže za dostavu sadržaja (engl. *Content Distribution Network*); preusmjeravanje prema geografskom položaju
    - mreža poslužitelja s višeodredišnom distribucijom sadržaja
  
- ◆ **transparentno razlučivanje imena**
  - potpuno replicirana "farma poslužitelja"
  - uravnoteživanje putem DNS-a
  - prilagodba prema dobu dana, zemljopisnom položaju i sl.
  
- ◆ **virtualni poslužitelj**
  - jedan virtualni poslužitelj = farma poslužitelja s uravnoteženjem opterećenja

- ◆ Što je *cache*, općenito?
  - relativno manji skup podataka s “povoljnijim” pristupom, napravljen sa svrhom izbjegavanja “nepovoljnijeg” pristupa većem skupu podataka
    - “povoljniji” pristup: manje novaca, manje komunikacije na veću udaljenost, kraće vrijeme čekanja (zadovoljniji korisnici!)
    - “nepovoljniji” pristup: (i) ISP-ovi plaćaju komunikaciju, sadržaj se “vuče iz daleka”, dulje vrijeme čekanja, neučinkovito
    - primjeri: memorija, disk, Web
- ◆ Ideja:
  - neke stranice su “popularnije”, tj. koriste se više i češće od drugih
  - ako možemo ustanoviti koje su to stranice, možemo ih staviti tamo gdje je jeftinije doći do njih
  - znamo što je “popularnije” prema statistici pristupa (dnevnik prometa, odnosno *log*)

- ♦ u osnovnoj izvedbi klijent-poslužitelj, klijenti komuniciraju s poslužiteljima izravno, bez posrednika

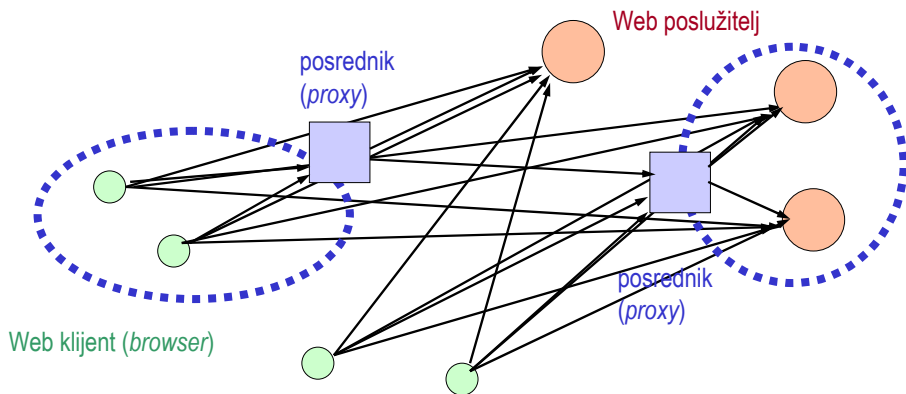


- ♦ klijenti (poslužitelji) se mogu grupirati

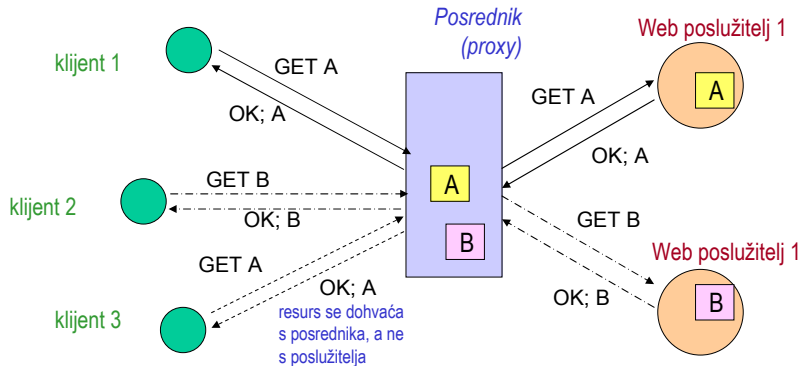




- ♦ posrednički poslužitelj (*proxy*) na kojem je smješteno priručno spremište (*cache*)
- ♦ bolja učinkovitost, raspoređivanje opterećenja

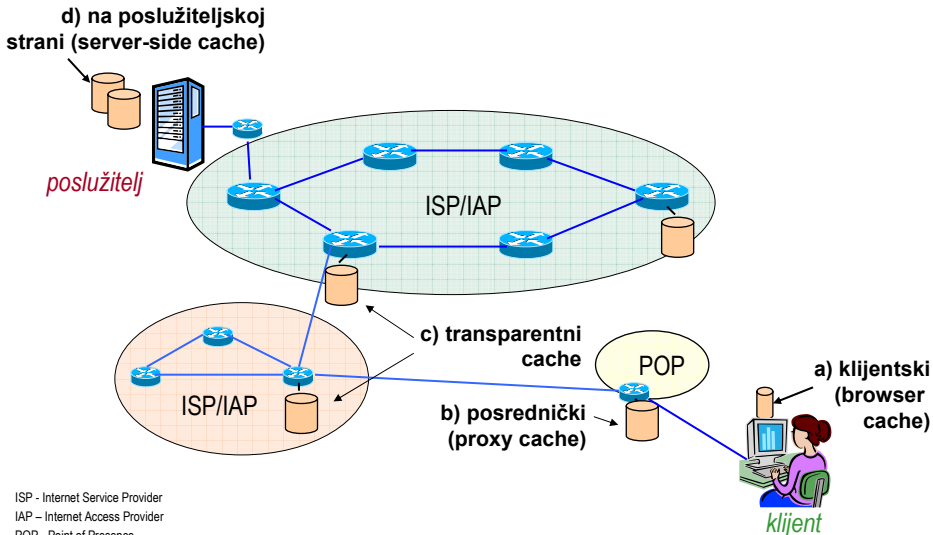


- ♦ primjer: *caching proxy*



Posrednički poslužitelj (engl. *proxy*) može imati i druge primjene:

- posredovanje za skupinu klijenata (ISP)
- “anonimiziranje” klijenta
- transformiranje zahtjeva i odgovora
- prilaz prema drugim (ne-HTTP) sustavima, npr. mail, FTP, i dr.
- vatrozid (*firewall*) – sigurnost
- filtriranje zahtjeva i odgovora, filtriranje sadržaja



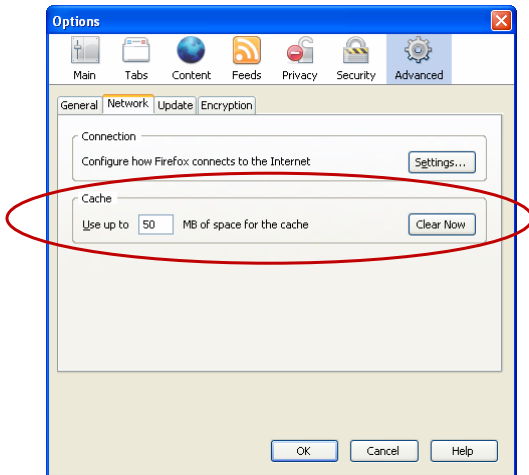
ISP - Internet Service Provider  
IAP - Internet Access Provider  
POP - Point of Presence

- ◆ a) **klijentski** *cache*
  - preglednik (*browser*) pohranjuje sadržaj URL-ova za budući pristup
- ◆ b) **posrednički** *cache* poslužitelj
  - posrednički poslužitelj (*proxy*) je smješten u mreži, obično “blizu” korisnika, npr. kod modemskih ili ADSL ulaza
  - pohranjuje sadržaj najčešće traženih URL-ova
- ◆ c) **transparentni posrednički** *cache* poslužitelj
  - transparentni posrednički poslužitelj je smješten u mreži, obično na rubu mreže pružatelja internetske usluge (*Internet Service Provider, ISP*)
  - ponaša se kao čvor u mreži za distribuciju sadržaja
  - sadržaj se obično distribuira višeodredišno (na razini aplikacije)
- ◆ d) *cache* na **poslužiteljskoj** strani
  - pohrana na strani poslužitelja (engl. *server-side cache*) smanjuje opterećenje poslužitelja

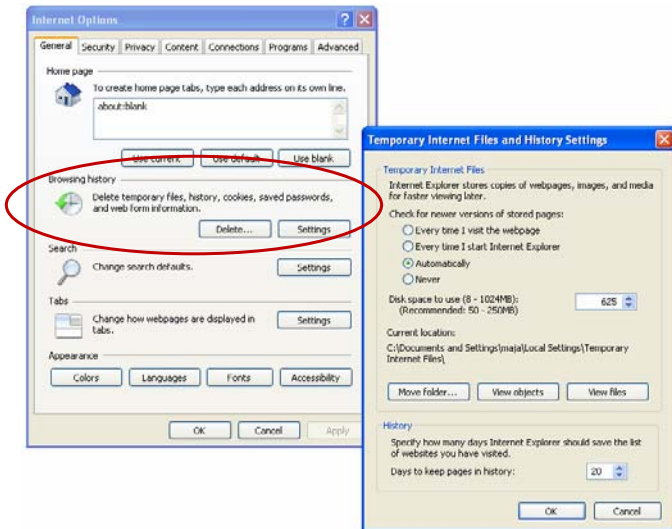
- ♦ primjer: Firefox 2.0



- ♦ preglednik može nuditi razne mogućnosti konfiguracije, npr:
  - prostor na disku
  - položaj
  - osvježavanje



- ♦ primjer: IE 7



- ◆ “klasični” **posrednički** *cache* poslužitelj
  - ponaša se kao Web poslužitelj
    - čeka da klijent pošalje HTTP zahtjev
    - ako ima pohranjeni dokument, vraća ga klijentu (kao poslužitelj), inače ga traži od izvornog poslužitelja
  - klijent, odn. preglednik “zna” da koristi proxy
  
- ◆ **transparentni posrednički** *cache* poslužitelj
  - klijent, odn. preglednik “ne zna” za proxy
  - HTTP zahtjevi klijenta se presreću na putu prema poslužitelju i, ako je dokument pohranjen, vraća ga se klijentu kao da dolazi od izvornog poslužitelja

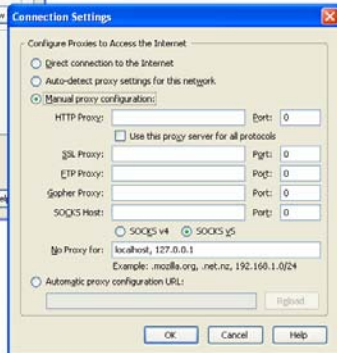
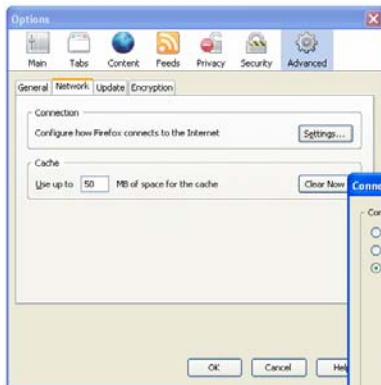


- ◆ Preglednik “zna” za *proxy*
  - korisnik mora konfigurirati preglednik za korištenje *proxy-ja*
  - svi HTTP zahtjevi šalju se *proxy-ju* umjesto na stvarnu adresu poslužitelja
  
- ◆ ako *proxy* nema dokument...
  - *proxy* dohvaća dokument s poslužitelja
  - vraća ga klijentu
  - odlučuje hoće li pohraniti dokument za evt. buduće zahtjeve
  
- ◆ najveća korist od *proxy-ja*: kod korisničke zajednice sa sličnim interesima!

- ♦ primjer: Firefox



- ♦ preglednik omogućuje konfiguraciju proxy-a
  - IP adresa i port
  - izuzeće za lokalne sadržaje
  - mogućnost automatske konfiguracije



- ◆ o čemu ovisi hoće li *proxy* pohraniti dokument?
  - “popularnost” dokumenta
  - relativna traženost u odnosu na već pohranjene dokumente (ako se jedan dokument dodaje, nešto drugo mora van!)
  
- ◆ kako mjeriti popularnost dokumenta?
  - vrijeme proteklo od zadnjeg pristupa
  - minimum broja pristupa dokumentu u zadanom vremenskom razdoblju
  
- ◆ što obrisati?
  - dokument koji je najdulje vrijeme bez potražnje
  - dokument s najvećim produktom  $vrijeme \times veličina\_dokumenta$ 
    - vrijeme otkako je dokument zadnji put zatražen pomnoženo s veličinom dokumenta
  - prema učestalosti potražnje, povijesnoj statistici pristupa (dugoročni uzorci), te kombinacijom navedenih parametara

- ◆ u HTTP-u postoje zaglavlja posebno namijenjena za podršku posredničkim poslužiteljima
  - zaglavlja kojima klijenti mogu postaviti svoje zahtjeve prema *proxy-ju*
  - zaglavlja kojima Web poslužitelji mogu označiti “pohranjive” dokumente i njihov “rok trajanja” (tj. vrijeme do osvježavanja)
  - zaglavlja kojima se *proxy* identifikira prema Web poslužitelju
  - zaglavlja kojima se može upozoriti klijente da je dokument zastario

- ◆ **uvjetni** (*conditional*) GET
- ◆ koristi polja zaglavlja
  - If-Modified-Since:
  - If-Unmodified-Since:
  - If-Match:
  - If-Unmatch:
- ◆ **304** (*Not Modified*)
  - traženi resurs nije promijenio sadržaj od zadnjeg zahtjeva
  - klijent može koristiti kopiju entiteta iz *cachea*

- ◆ preglednik “nije svjestan” da koristi transparentni *cache* poslužitelj
- ◆ transparentni *cache* poslužitelj “presreće” HTTP zahtjev na putu kroz Internet
  - ako ima dokument već pohranjen, vraća ga klijentu
    - transparentni *cache* “glumi” poslužitelja, odn. mijenja povratnu adresu na paketima koji nose dokument
  - inače, ako nema dokument, zahtjev propušta prema odredištu
- ◆ u HTTP-u nema podrške za transparentni *cache* (zašto? - izvan protokola)

- ◆ što znači “presretanje” zahtjeva?
  - filtriranje svakog TCP paketa kako bi se otkrilo one koji evt. nose HTTP zahtjeve
  - kriteriji:
    - dobro-znani broja porta
    - strukturirani format HTTP zahtjeva
  - transparentni cache se zato naziva i “*Layer-4 cache*” (prema TCP-u koji je protokol 4. sloja)
- ◆ interes za postavljanje transparentnog cache-a
  - općenito, ISP-ovi
  - cilj: što manje prometa preko svoje “granice”, smanjeni troškovi prema drugim ISPovima

- ◆ neki sadržaji nisu “pohranjivi”
  - često promjenjivi sadržaj (npr. cijene dionica)
  - osobni sadržaj (npr. osobni podaci, personalizirane usluge)
  - šifrirani sadržaj (npr. broj kreditne kartice)
  
- ◆ HTTP ima zaglavlja kojima davatelji usluge mogu u HTTP-odgovor postaviti oznaku da sadržaj nije “pohranjiv”, i od programske podrške *cachea* se očekuje se da to poštuje
  - problem: lažno označavanje sadržaja kao “ne-pohranjivog” (banneri, reklame!) od strane poslužitelja kako *cache* ne bi smanjivao promet prema poslužitelju
  - neki transparentni poslužitelji nude brojanje zahtjeva (“*hit count*”) kao alternativu

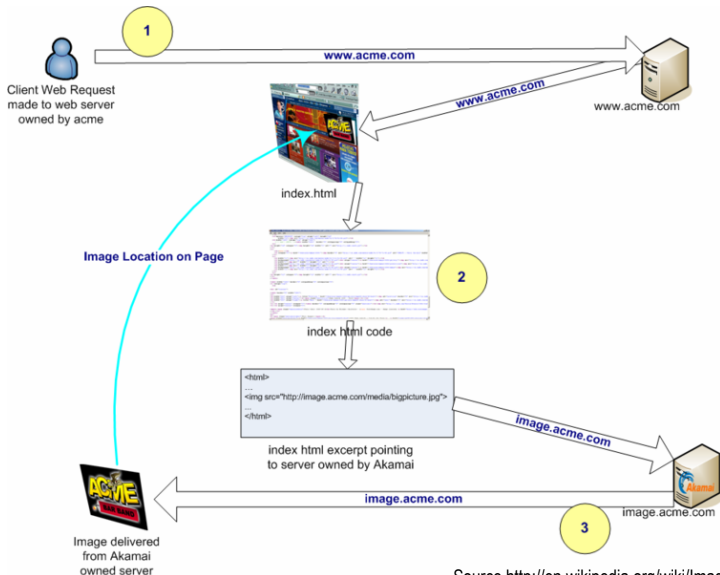


- ◆ hijerarhijski *cache*-ovi
  - stablasta hijerarhijska struktura kao proširenje osnovne ideje
  - višeodredišni protokoli za koordinaciju sadržaja po razinama
  
- ◆ djelomično pohranjivanje
  - većina Web sjedišta kombinira isti sadržaj unutar stranica (npr. pozadina, slike)
  - ideja je pohraniti taj sadržaj, a s izvornog poslužitelja dohvaćati samo razlike (XML?)
  
- ◆ primjeri:
  - Akamai ([www.akamai.com](http://www.akamai.com)), CDNetworks (<http://www.us.cdnetworks.com/>)
  - Coral CDN (<http://www.coralcdn.org/>), Squid (<http://sourceforge.net/projects/squid/>)

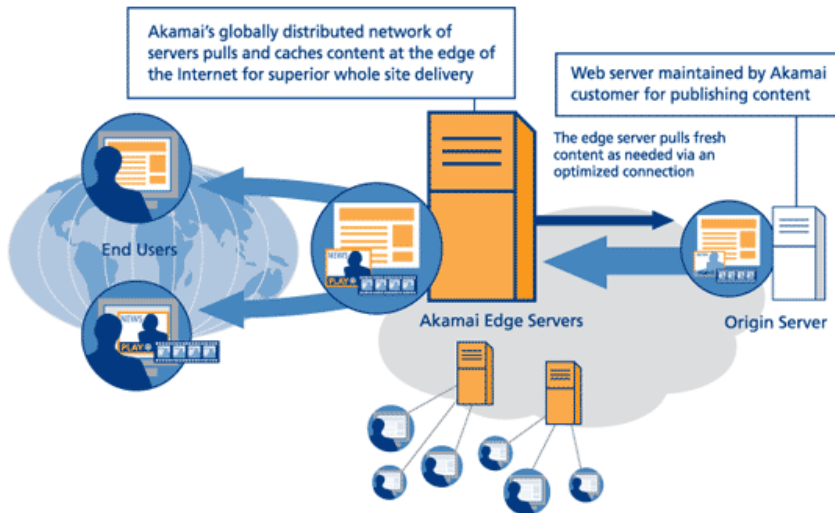
# Primjer CDN: Akamai



Zavod za telekomunikacije



Source <http://en.wikipedia.org/wiki/Image:Akamaiprocess.png>



Izvor: [www.akamai.com](http://www.akamai.com)

- ◆ **World Wide Web Consortium (W3C)**

<http://www.w3c.org/>

*“The World Wide Web Consortium (W3C) develops interoperable technologies (specifications, guidelines, software, and tools) to lead the Web to its full potential.”*

- ◆ **Internet Engineering Task Force**

<http://www.ietf.org>

- **RFC-Editor**

<http://www.rfc-editor.org>

- **RFC Search**

<http://www.rfc-editor.org/rfcsearch.html>

- **RFC Index**

<http://www.rfc-editor.org/rfc-index.html>