

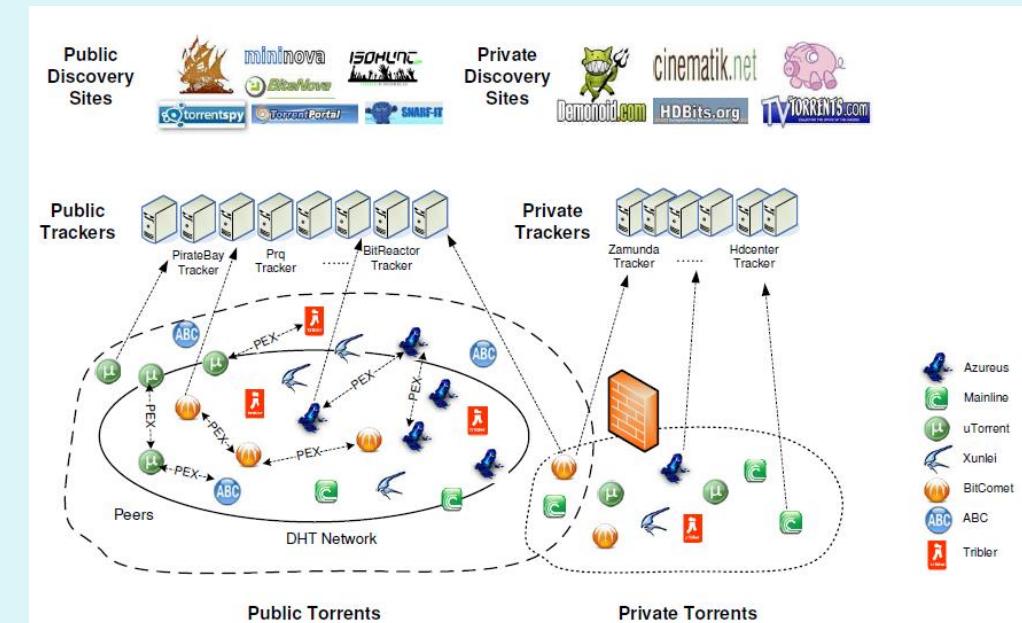
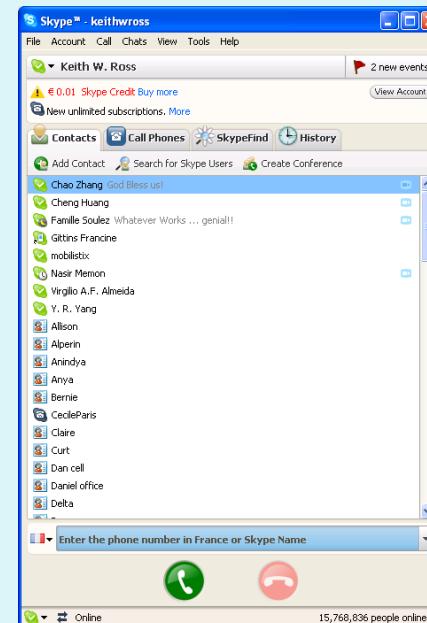
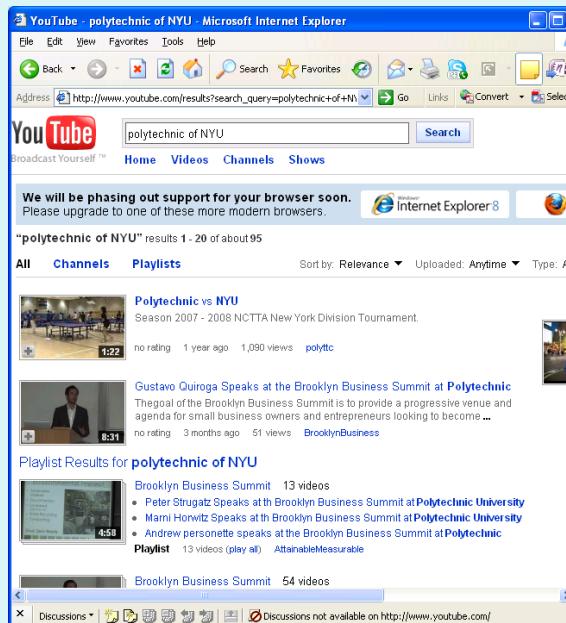
Računalniške komunikacije

2020/21

aplikacijska plast
HTTP, SMTP, POP3

Omrežne aplikacije

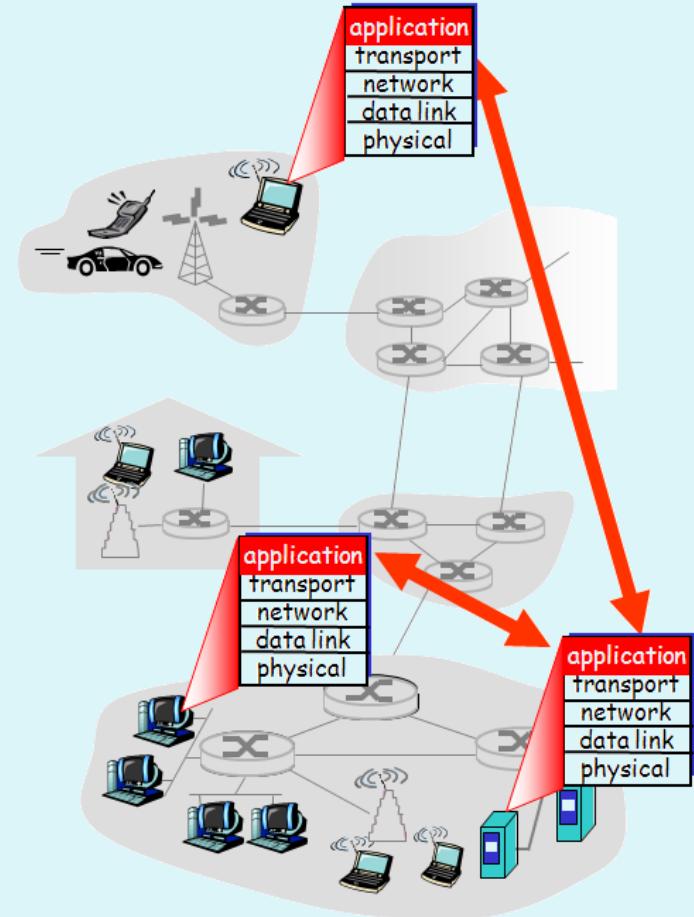
- so glavni razlog za povezovanje računalnikov v omrežja (*raison d'être*)!
- razlikovanje med:
 - aplikacijo v računalniku, ki uporablja omrežje (kos programske opreme)
 - omrežno aplikacijo (porazdeljena aplikacija iz več komponent)
- aplikacijski protokoli so običajno del aplikacije
- ne potrebuje nujno GUI



Pogoste aplikacije

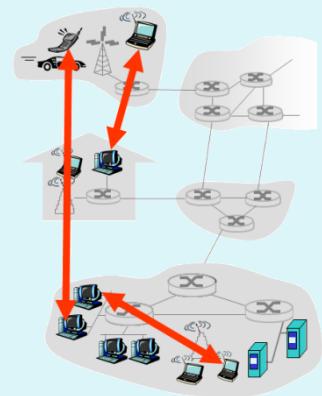
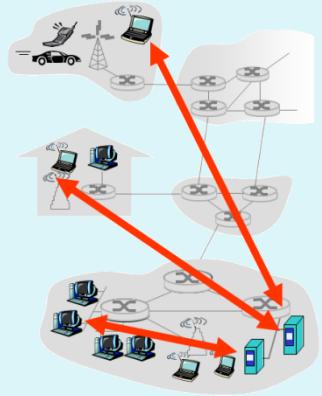
- e-mail
- splet
- takojšnje sporočanje (*instant messaging*)
- oddaljen dostop (*remote login*)
- izmenjava datotek (P2P)
- omrežne igre
- multimedijijske vsebine
- družabna omrežja
- IP telefonija
- konferenčne aplikacije
- porazdeljeno procesiranje

➤ programer mora implementirati aplikacije, ki tečejo na različnih sistemih in komunicirajo med seboj. Podrobnosti omrežja zanj niso pomembne



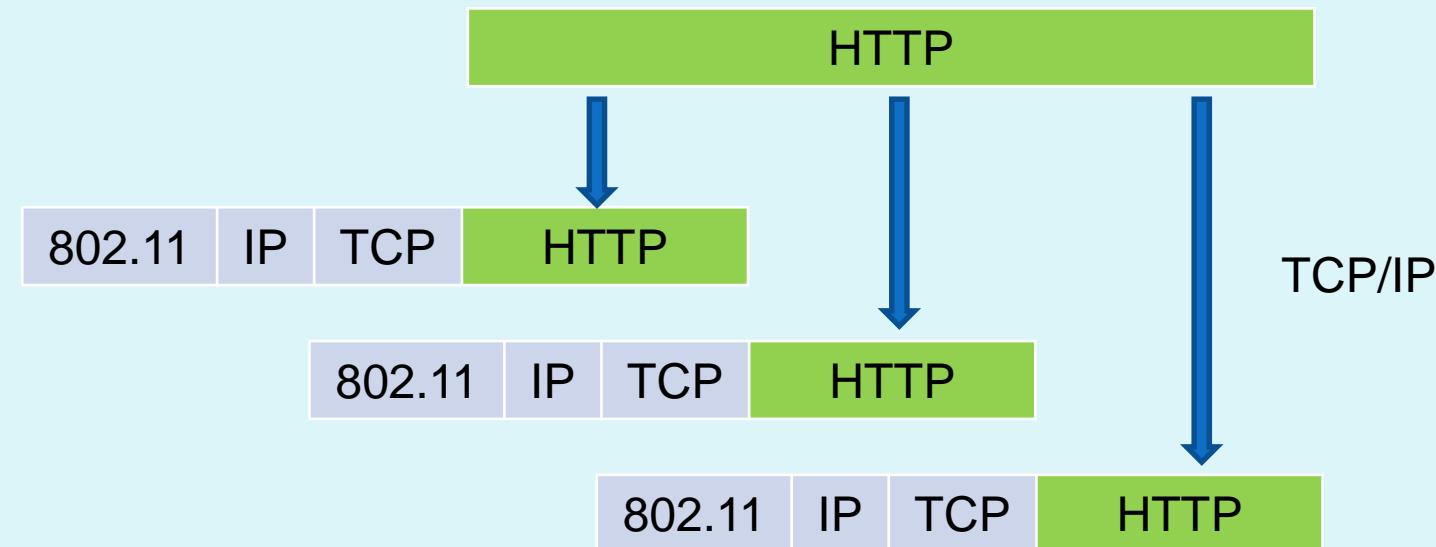
Protokoli aplikacijske plasti

- arhitekture aplikacij:
 - **strežnik in odjemalec**: strežnik ves čas dostopen, odjemalci so priključeni le občasno (http, ftp, telnet, e-mail),
 - **peer-to-peer (P2P)**: končni sistemi direktno komunicirajo med seboj, člani so lahko nedostopni (BitTorrent, eMule)
 - **mešana arhitektura** (Skype, takojšnje sporočanje)
- aplikacijski protokoli so običajno *berljivi* uporabniku
- lastništvo protokolov:
 - **javni** (odprti) protokoli:
 - HTTP (RFC 2616), SMTP, BitTorrent, specifikacije v dokumentih RFC
 - **lastniški** (zaprte, angl. *proprietary*) protokoli, npr. Skype
 - ni javnih specifikacij
 - delovanje ugotavljamo z obratnim inženiringom



Aplikacijska sporočila

- običajno razdeljena in enkapsulirana v več segmentov (TCP/UDP?)



Kateri transportni protokol uporabiti?



Zahteve aplikacij po storitvah transporta

Aplikacija	primeri aplikacijskih protokolov	dovoljena izguba	potrebna minimalna časovna širina	časovna občutljivost	običajen transportni protokol
prenos datotek	FTP	ne	ne (elastična)	ne	TCP
e-pošta	SMTP, POP3, IMAP	ne	ne (elastična)	ne	TCP
splet	HTTP	ne	ne (elastična)	ne	TCP
oddaljen dostop	telnet	da	~10 kb/s	da	TCP
multimedija (realni čas)	RTP, HTTP	da	10 kb/s – 5 Mb/s	da	TCP/UDP
multimedija (shranjena)	RTP, HTTP	da	10 kb/s – 5 Mb/s	da	TCP/UDP
interaktivne igre	HTTP, lastniški	da / ne (?)	~10 kb/s	da	TCP/UDP
IP telefonija	SIP, RTP, lastniški	da	100 kb/s – 3 Mb/s	da	TCP/UDP

Kateri protokol uporabiti?

TCP ali UDP? Zanesljivost, performanse?

Izbira transportnega protokola in zanesljivost dostave

HTTP, FTP

**zaporedja podatkov
spremenljive dolžine,
komunikacija po principu
zahteva/odgovor**

TCP

RTP, Skype

**nezanesljiv prenos
(tokovi v realnem času)
ali kratka sporočila**

UDP

?

**kaj narediti, če
potrebujemo
zanesljivost (DNS,
DHCP) ?**

UDP

APLIKACIJSKI PROTOKOLI

HTTP

FTP

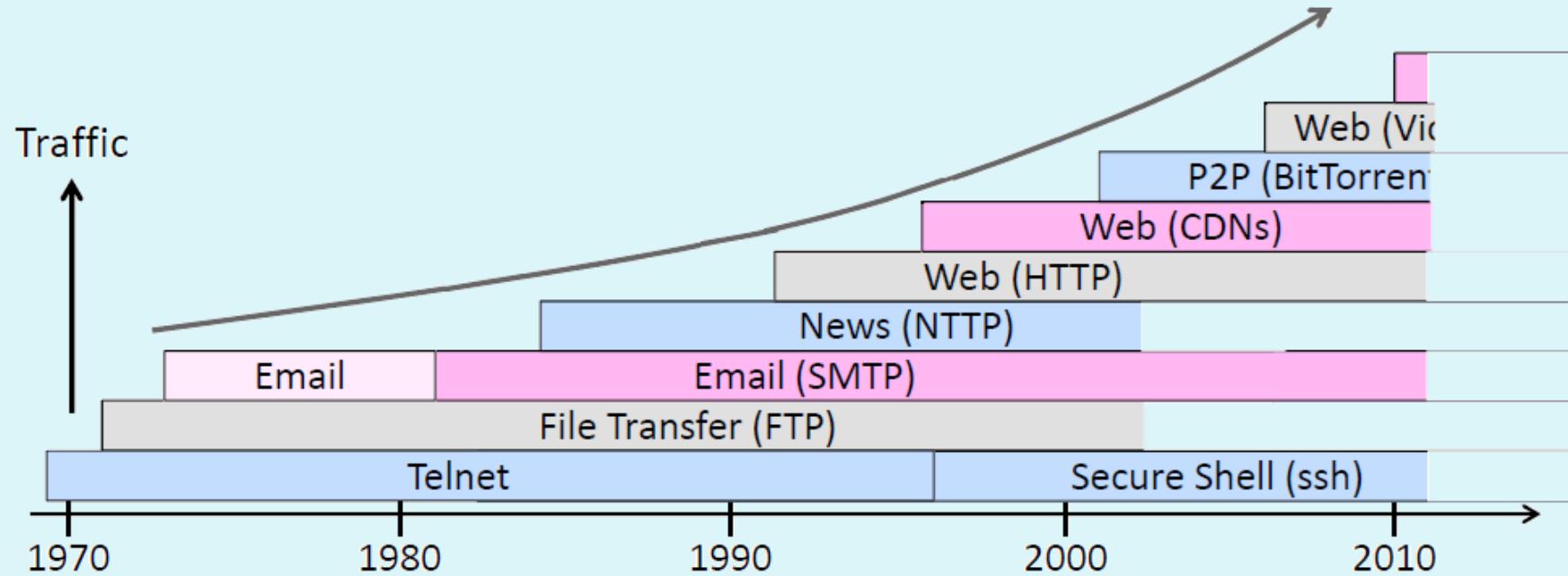
SNMP

DNS

P2P

1	ISO 802.3 z Gigabit Ethernet	ISO 802.2 Ethernet	SDH-DCC 2M TSn	Physical
2	ARP Address Resolution Protocol ISO 802.2 LLC ISO 802.3 MAC	RARP Reverse Address Resolution Protocol	Frame Relay X.25 ISDN PPP	Data Link
3	IP Internet Protocol	ICMP Internet Control Message Protocol		Network
4	TCP Transmission Control Protocol	UDP User Datagram Protocol		Transport
	SMPT / IMAP4 (E-MAIL)	Telnet (Terminal)	RIP Route Information Protocol TFTP Trivial File Transfer Protocol SNMP Network Management	OSPF Open Shortest Path First DHCP Dynamic Host Configuration Protocol BOOTP Bootstrap Protocol

Razvoj protokolov skozi čas

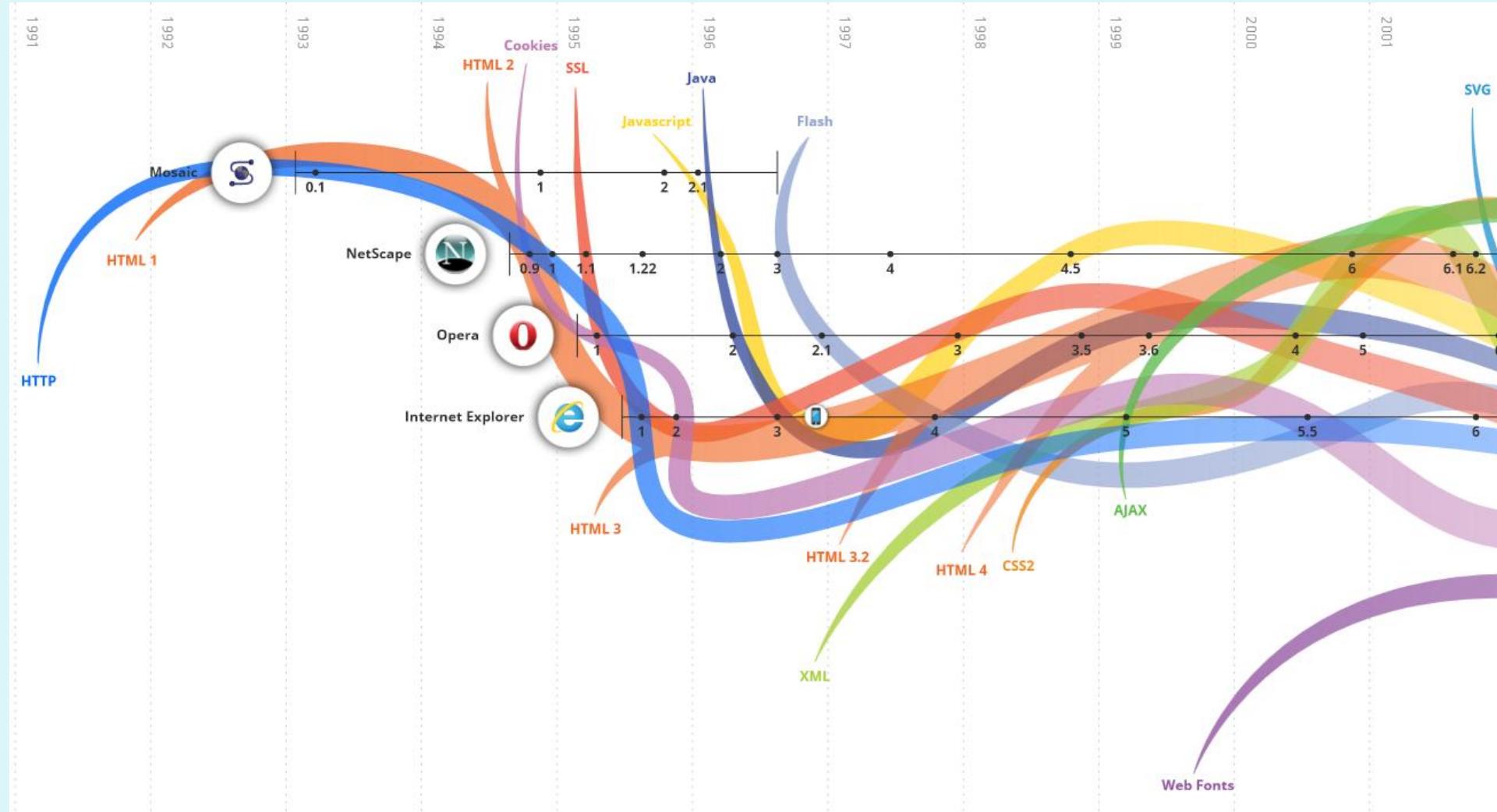


- napovedi za prihodnja leta:
 - kmalu bo 90% prometa v internetu vsebovalo prenos slike/videoa
 - brezžičnega prometa bo kmalu več kot prometa po žici,
 - 15% prometa naredijo mobilni uporabniki,
 - naraščajoča količina škodoželnega prometa (napadi) iz Kitajske, Rusije in ZDA

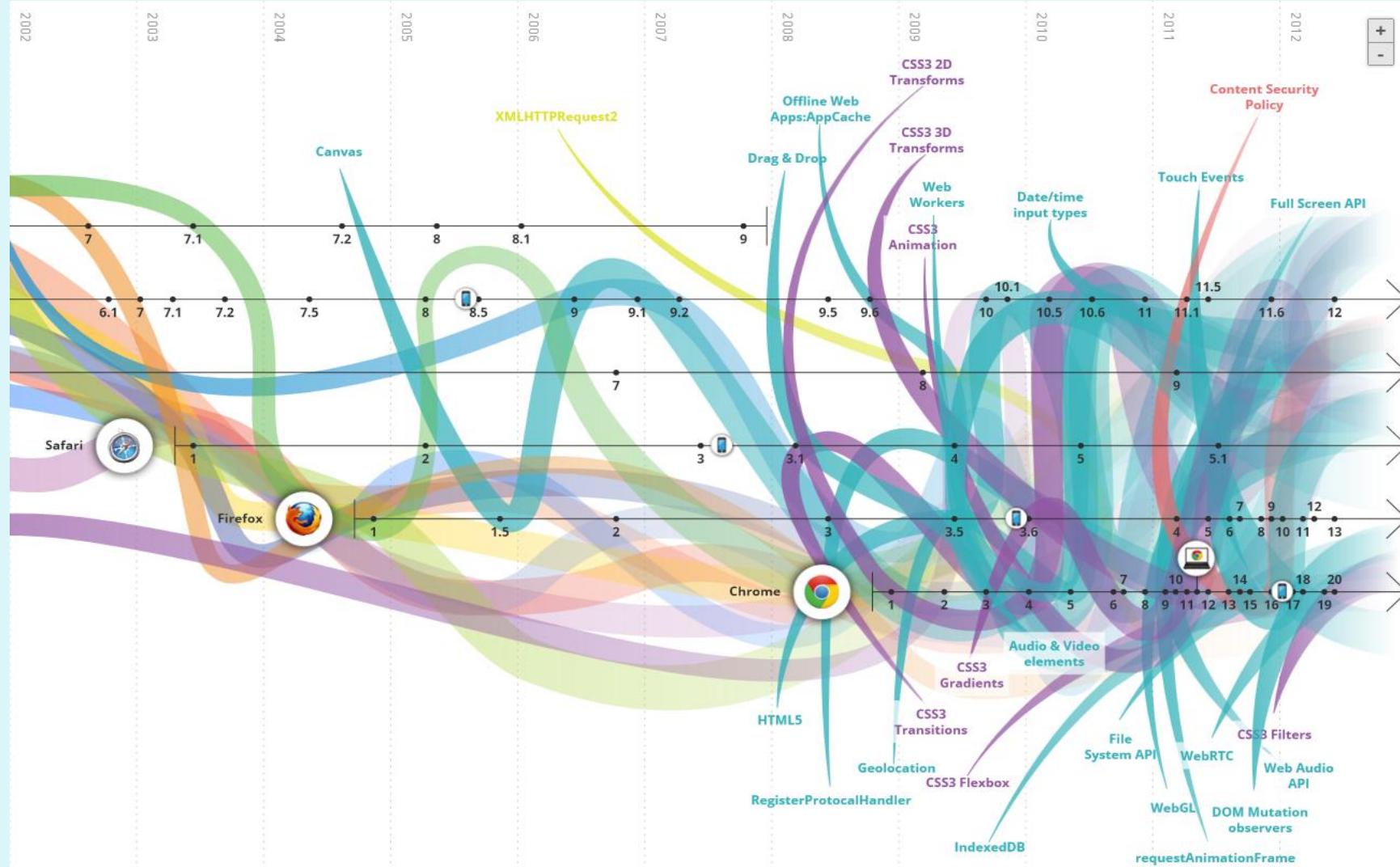
PROTOKOL HTTP



Razvoj spletnih tehnologij 1/2

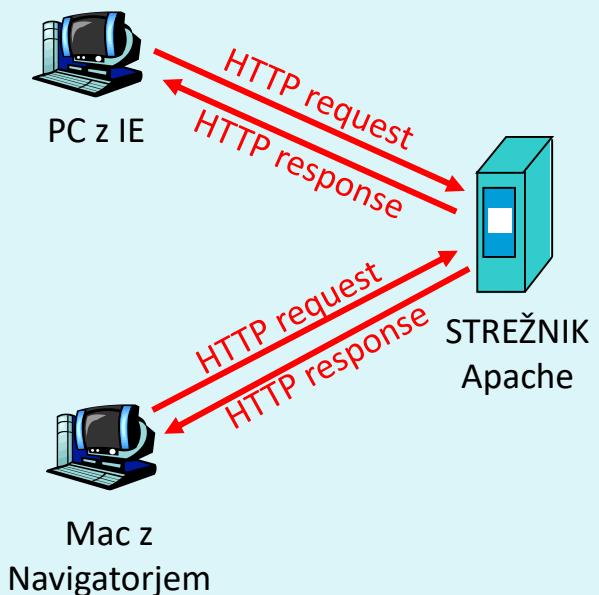
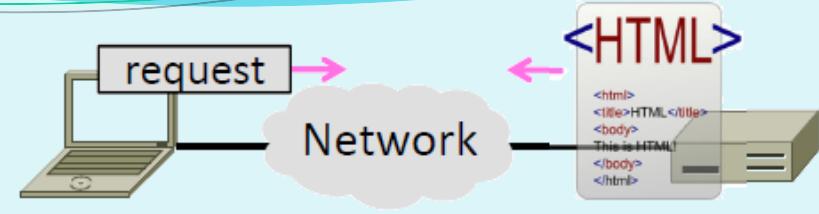


Razvoj spletnih tehnologij 2/2



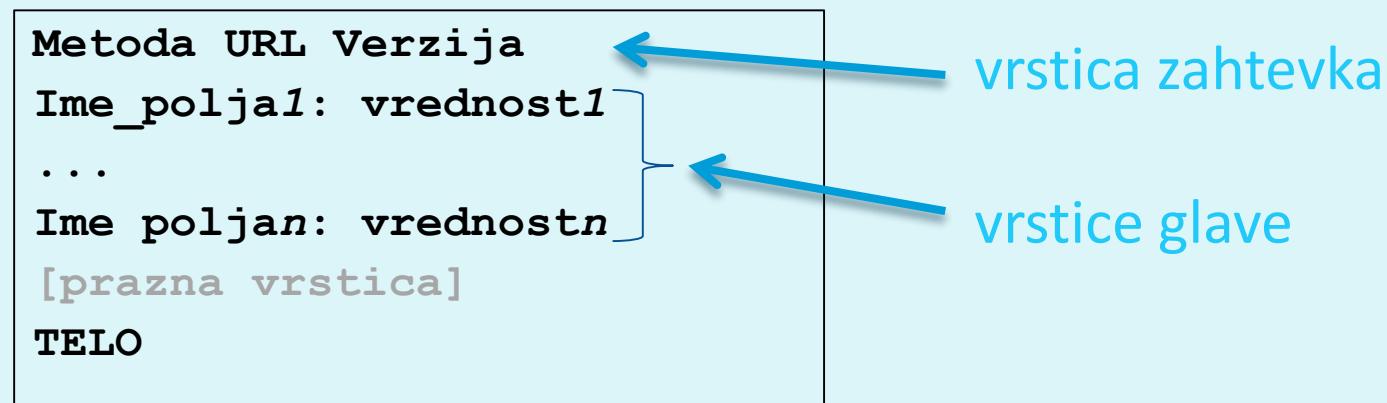
Splet in HTTP

- pojavi se v 1990-ih, revolucionarna aplikacija, omogoča:
 - dostop do vsebin **na zahtevo**
 - vsak lahko oglašuje
 - iskanja, povezave, grafika, vmesniki, multimedija
- specifikaciji : RFC 1945 (HTTP 1.0) in RFC 2616 (HTTP 1.1)
- delovanje:
 - odjemalec naslovi TCP **zahtevo** (*request*) na vrata 80 strežnika
 - strežnik vrne http **odgovor** (*response*)
 - TCP poskrbi za potrditve, ponovitve, vrstni red
 - protokol brez stanj (*stateless*)



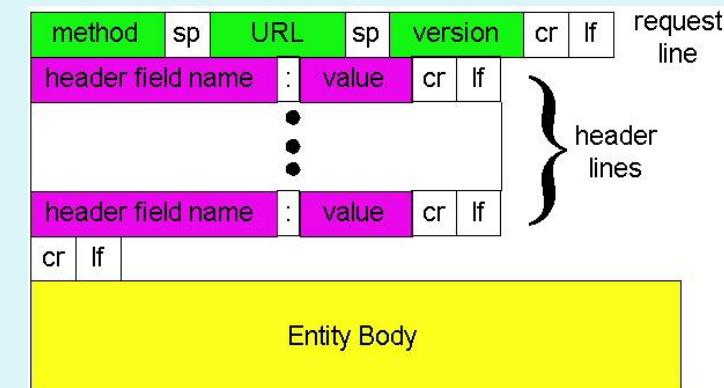
Oblika sporočila HTTP request

- dve vrsti sporočil: REQUEST (zahteva) in RESPONSE (odgovor)



primer

```
GET /sem/ocene.htm HTTP/1.1
Host: marvin.fri.uni-lj.si
Connection: close
...
```

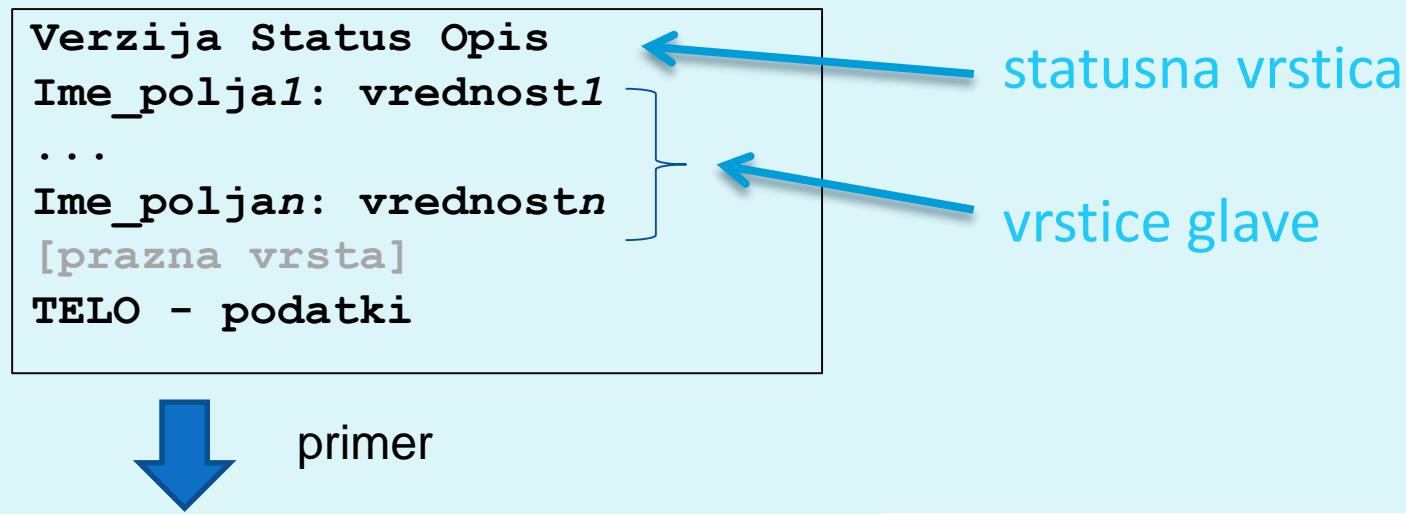


Metode pri HTTP zahtevi

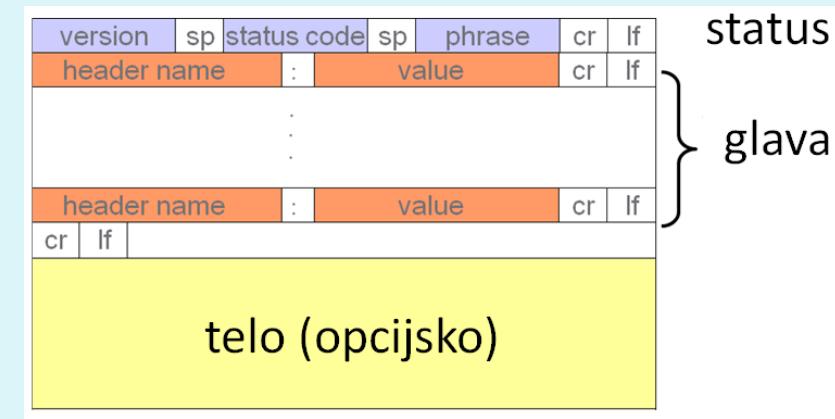
GET	zahteva objekta
POST	zahteva objekta + poslane vrednosti (obrazci) <ul style="list-style-type: none">• obrazec lahko uporabi tudi metodo GET, vrednosti parametrov pa pošilja kot podaljšan naslov (http://...../search/rsr.aspx?lang=slv&id=31318)
HEAD	zahteva, na katero strežnik odgovori z odgovorom brez zahtevanega objekta (uporabno za razhroščevanje)
PUT	(HTTP 1.1) – nalaganje na strežnik (upload)
DELETE	(HTTP 1.1) – brisanje s strežnika
TRACE	razhroščevanje (echo-odmev zahtevka, podobno PING)
CONNECT	povezava preko medstrežnika
OPTIONS	povpraševanje o možnih opcijah pri zahtevku

Oblika sporočila HTTP odgovor

- dve vrsti sporočil: REQUEST (zahteva) in RESPONSE (odgovor)



HTTP/1.1 200 OK
Connection: close
Date: Mon, 05 Nov 2007
 12:18:23 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: ...
Content-Length: 6534
Content-Type: text/html

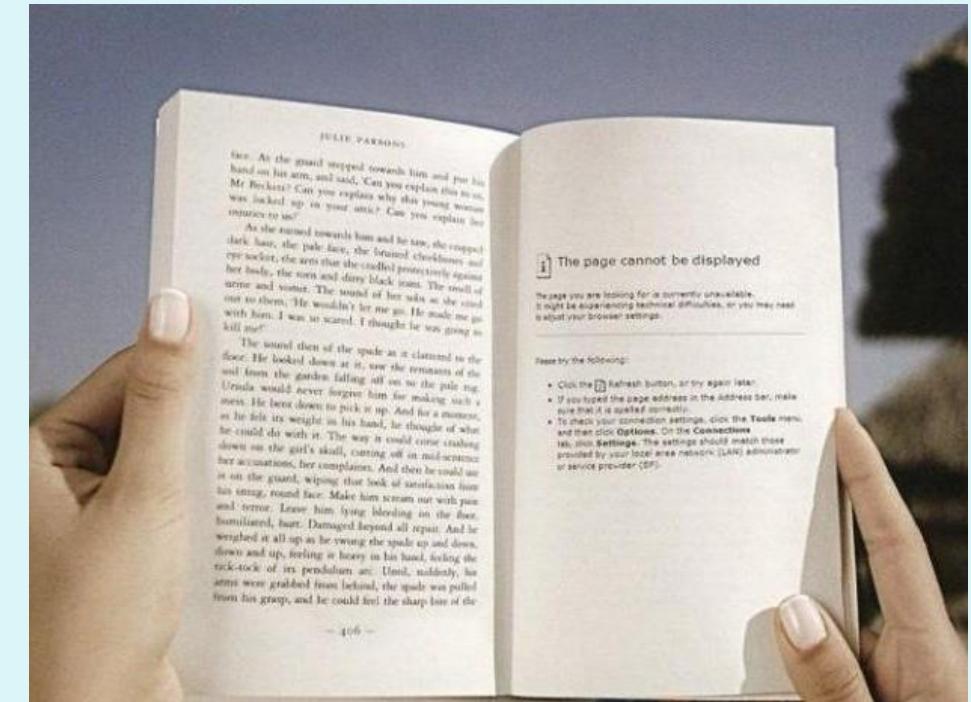


Statusi v HTTP odgovorih

HTTP Status Codes				
For great REST services the correct usage of the correct HTTP status code in a response is vital.				
1xx – Informational	2xx – Successful	3xx – Redirection	4xx – Client Error	5xx – Server Error
This class of status code indicates a provisional response, consisting only of the Status-Line and optional headers, and is terminated by an empty line 100 – Continue 101 – Switching Protocols 102 – Processing	This class of status code indicates that the client's request was successfully received, understood, and accepted. 200 – OK 201 – Created 202 – Accepted 203 – Non-Authoritative Information 204 – No Content 205 – Reset Content 206 – Partial Content 207 – Multi-Status	This class of status code indicates that further action needs to be taken by the user agent in order to fulfill the request. 300 – Multiple Choices 301 – Moved Permanently 302 – Found 303 – See Other 304 – Not Modified 305 – Use Proxy 307 – Temporary Redirect	The 4xx class of status code is intended for cases in which the client seems to have erred. 400 – Bad Request 401 – Unauthorised 402 – Payment Required 403 – Forbidden 404 – Not Found 405 – Method Not Allowed 406 – Not Acceptable 407 – Proxy Authentication Required 408 – Request Timeout 409 – Conflict 410 – Gone 411 – Length Required 412 – Precondition Failed 413 – Request Entity Too Large 414 – Request URI Too Long 415 – Unsupported Media Type 416 – Requested Range Not Satisfiable 417 – Expectation Failed 422 – Unprocessable Entity 423 – Locked 424 – Failed Dependency 425 – Unordered Collection 426 – Upgrade Required	Response status codes beginning with the digit "5" indicate cases in which the server is aware that it has erred or is incapable of performing the request. 500 – Internal Server Error 501 – Not Implemented 502 – Bad Gateway 503 – Service Unavailable 504 – Gateway Timeout 505 – HTTP Version Not Supported 506 – Variant Also Negotiates 507 – Insufficient Storage 510 – Not Extended
Examples of using HTTP Status Codes in REST				
201 – When doing a POST to create a new resource it is best to return 201 and not 200. 204 – When deleting a resources it is best to return 204, which indicates it succeeded but there is no body to return. 301 – If a 301 is returned the client should update any cached URI's to point to the new URI. 302 – This is often used for temporary redirect's, however 303 and 307 are better choices. 409 – This provides a great way to deal with conflicts caused by multiple updates. 501 – This implies that the feature will be implemented in the future.				
Special Cases				
306 – This status code is no longer used. It used to be for switch proxy. 418 – This status code from RFC 2324. However RFC 2324 was submitted as an April Fools' Joke. The message is <i>I am a teapot.</i>				
Key	Description			
Black	HTTP version 1.0			
Blue	HTTP version 1.1			
Aqua	Extension RFC 2295			
Green	Extension RFC 2518			
Yellow	Extension RFC 2774			
Orange	Extension RFC 2817			
Purple	Extension RFC 3648			
Red	Extension RFC 4918			

Statusi v HTTP odgovorih

- **1xx:** informativne kode
- **2xx:** uspešno
 - 200: OK
- **3xx:** preusmeritev
 - 301: Moved Permanently - prestavljen dokument
- **4xx:** napake pri odjemalcu
 - 400: Bad Request – sintaksa
 - 404: Not Found – ni dokumenta
- **5xx:** napake na strežniku
 - 500: Internal Server Error
 - 505: HTTP Version Not Supported

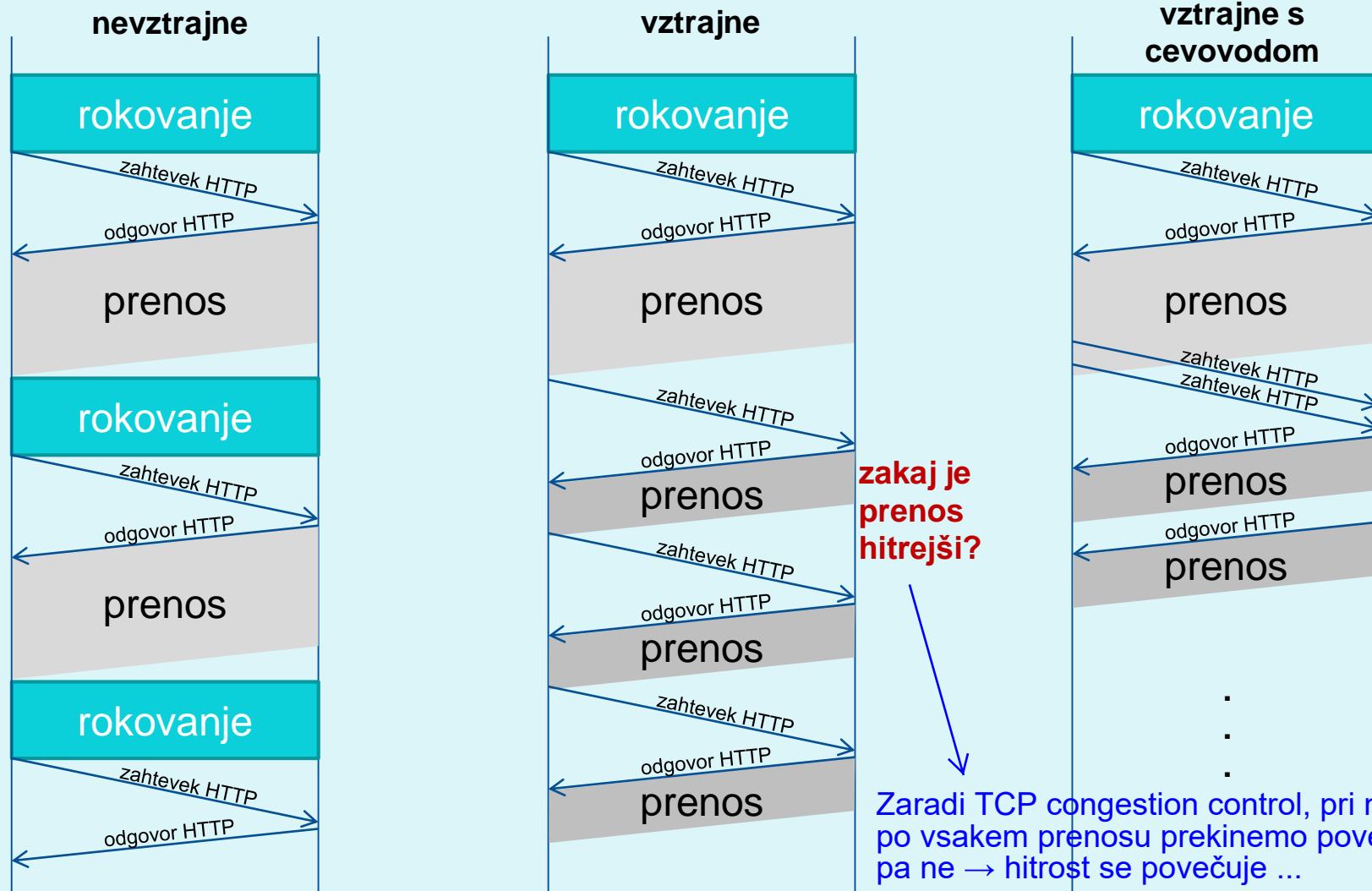


Vrste HTTP povezav

RTT = Round Trip Time

1. **nevztrajne (*nonpersistent*)**: za vsak prenašani objekt (stran, slika) se vzpostavi nova TCP povezava
 - 2 RTT/objekt (rokovanje + prenos)
2. **vztrajne (*persistent*)**: strežnik uporabi isto povezavo za pošiljanje več objektov, strežnik pusti povezavo odprto
 - 1 RTT/objekt (prenos)
3. **vztrajne s cevovodom (*persistent, pipelined*)**: tekoče pošiljanje več zahtev naenkrat, brez čakanja na prejem prejšnjih

Vrste HTTP povezav



Uporaba piškotkov

- HTTP ne pomni stanja povezav (*stateless*), zato med seboj ne razlikuje odjemalcev in ne pomni zgodovine
- nad plastjo HTTP (brez stanj) se ustvari dejna plast (s stanji)
- bogatejša uporabniška izkušnja:
 - avtorizacija
 - nakupovalni vozički
 - priporočila
 - shranjeni podatki o uporabnikih
- problem: varovanje zasebnosti

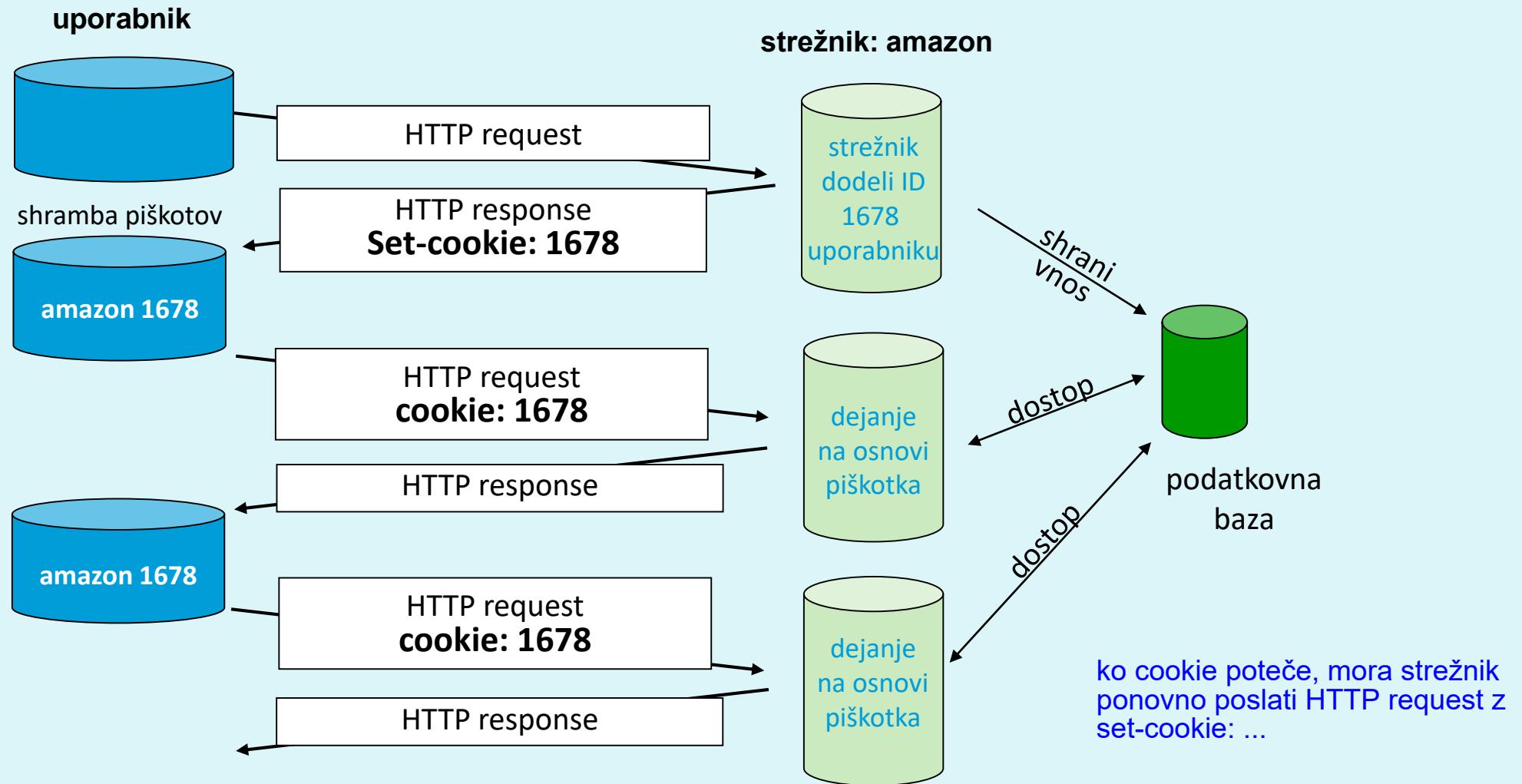


Uporaba piškotkov

- rešitev: uporaba piškotkov (identifikator, ki ga strežnik dodeli uporabniku in ga uporabnik lokalno shrani).
- Elementi:
 - datoteka z ID piškota shranjena na uporabnikovem računalniku
 - strežnik hrani evidenco o uporabnikih izdanih piškotov
 - ID piškota dodamo v glavo zahteve
 - ID piškota dodamo v glavo odgovora

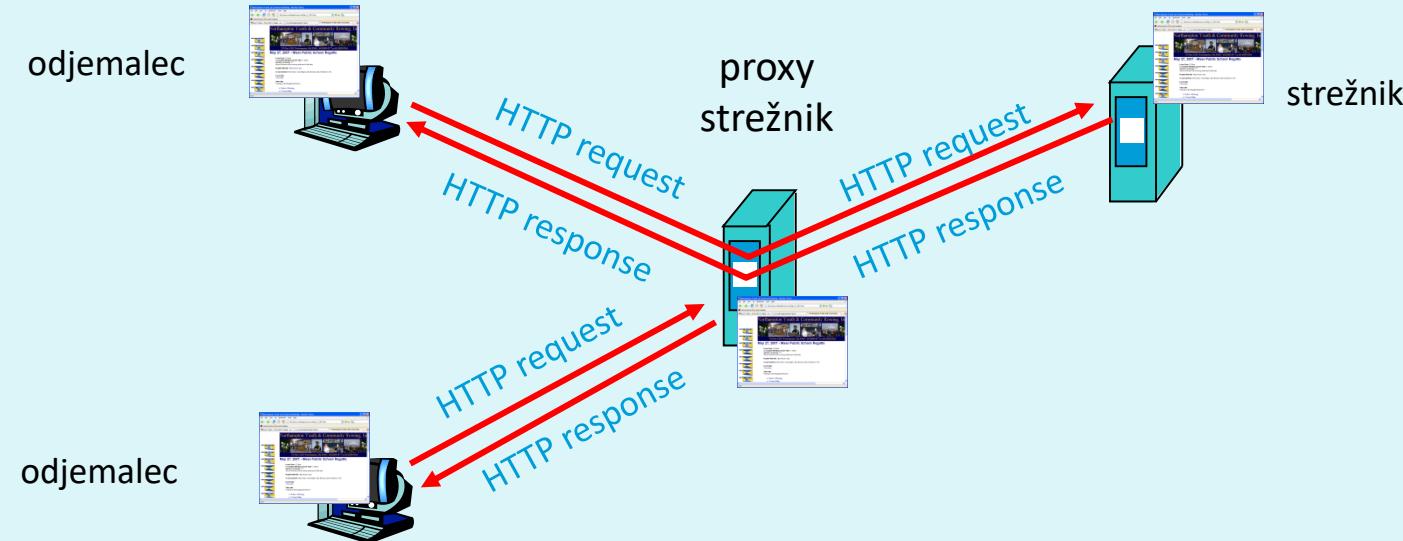


Uporaba piškotov



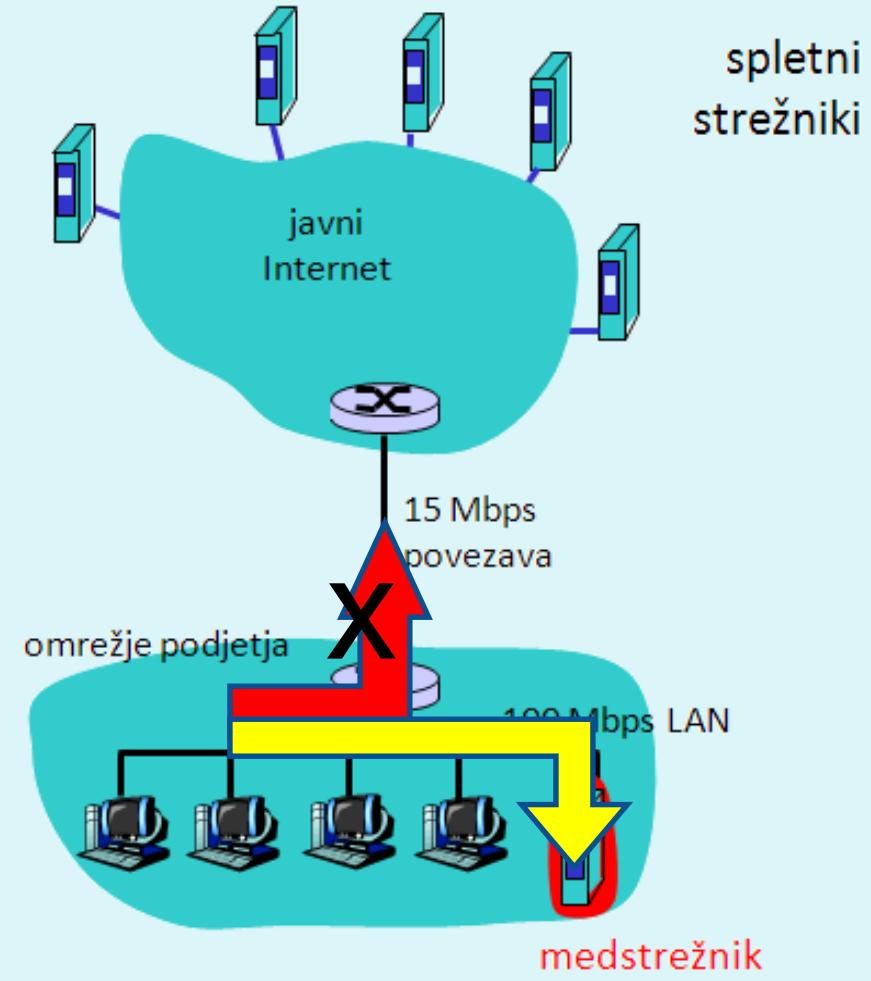
Medstrežnik (*proxy strežnik*)

- *Web cache, proxy strežnik* (navadno pri ISP-ju)
- odgovarja na zahteve namesto strežnikov, ima svoje kopije spletnih strani (če je nima, jo zahteva od pravega strežnika)
- odjemalec mora biti ustrezno konfiguriran



Zakaj posredniki?

- medstrežnik igra vlogo odjemalca in strežnika
- omogoči hitrejši odgovor odjemalcu
- manj prometa na dostopni povezavi do javnega omrežja
- pogojna zahteva (*conditional GET*)
 - vrstica glave:
`If-modified-since: Wed, 31 Oct 2007 09:32:22`
 - strežnik pošlje novo stran ali
`HTTP/1.1 304 Not Modified` (prazno telo)



Možni scenariji

1



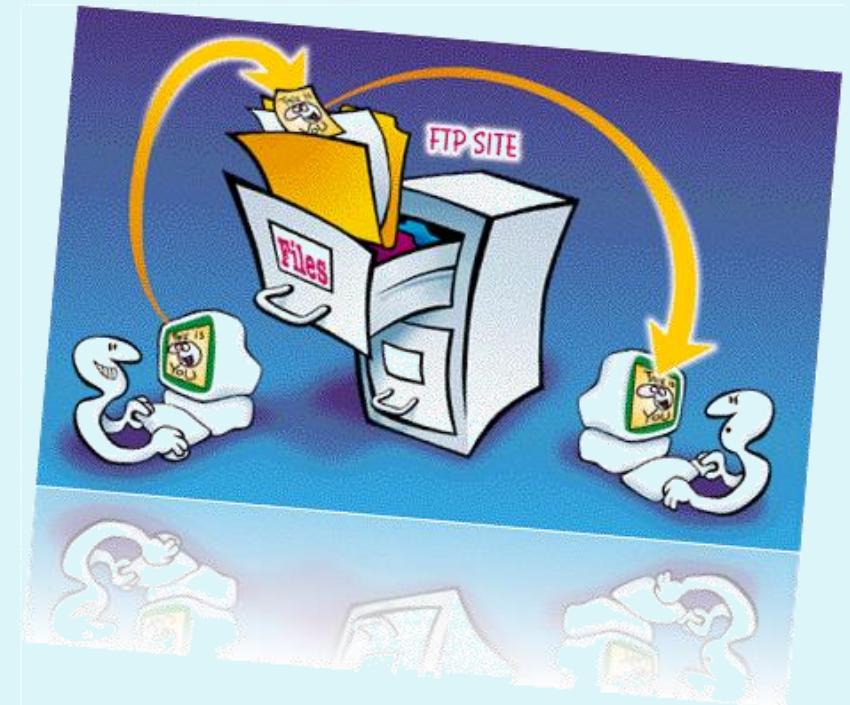
2



3

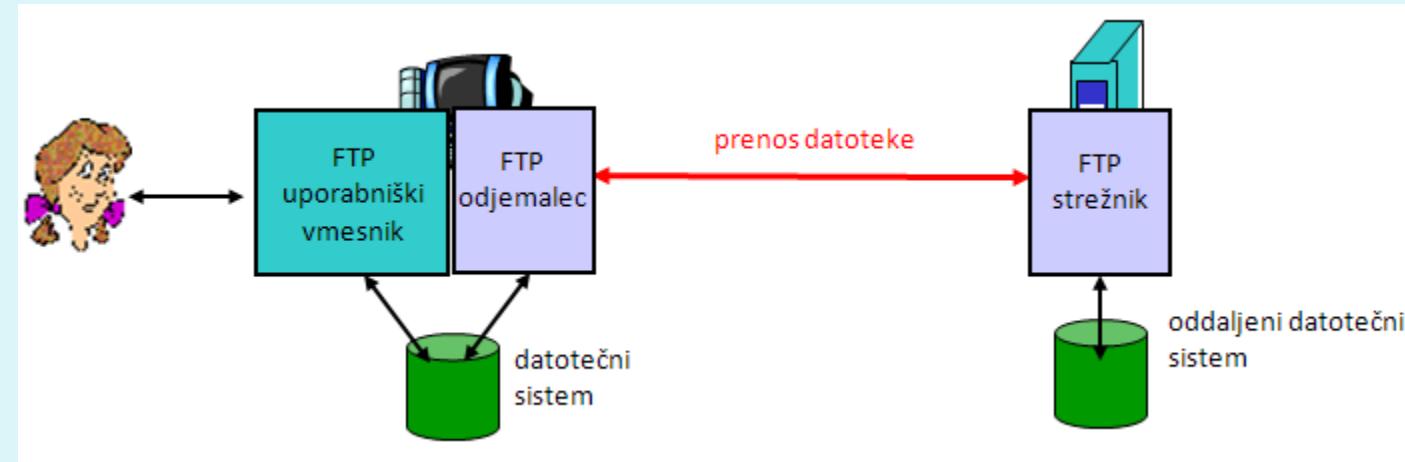


PROTOKOL FTP



Prenos datotek (File Transfer Protocol)

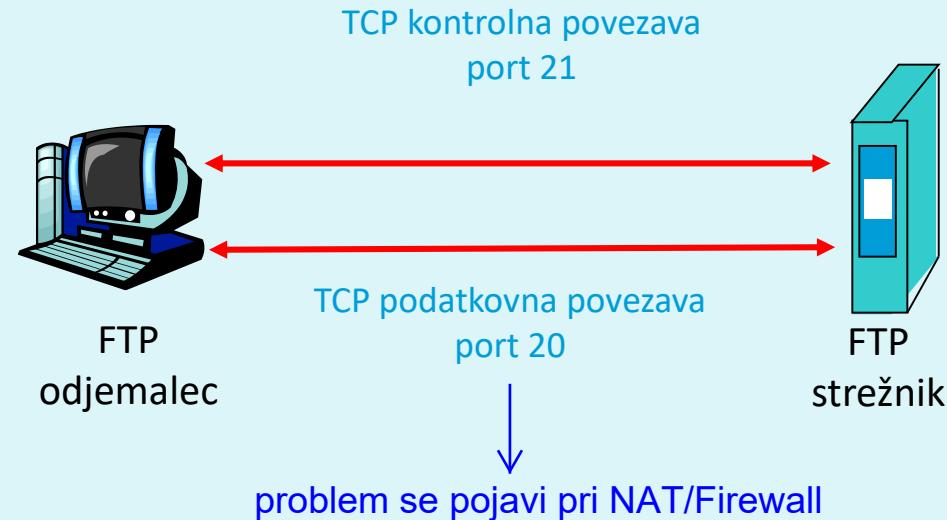
- prenos datotek z oddaljenega računalnika k uporabniku in obratno



- protokol, ki hrani stanje povezave: strežnik ve, kdo je odjemalec (avtentikacija), spremišča, kateri imenik pregleduje itd.

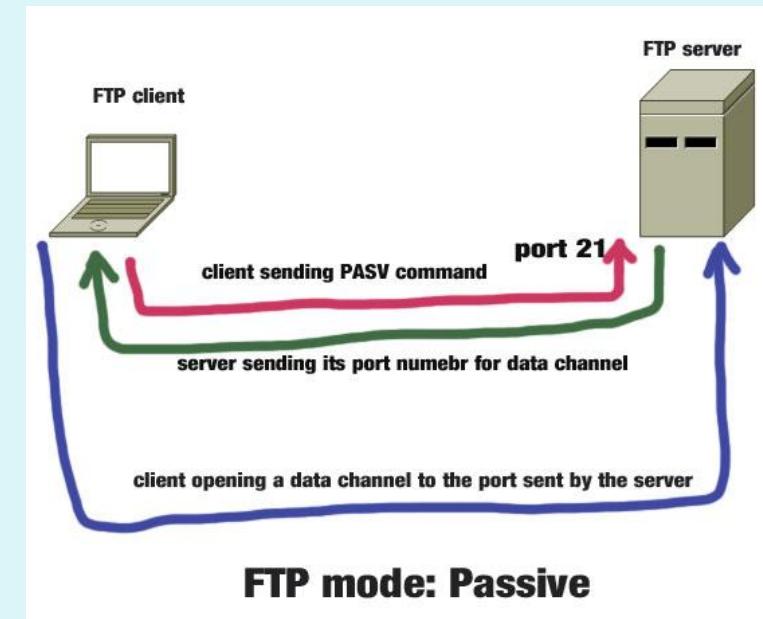
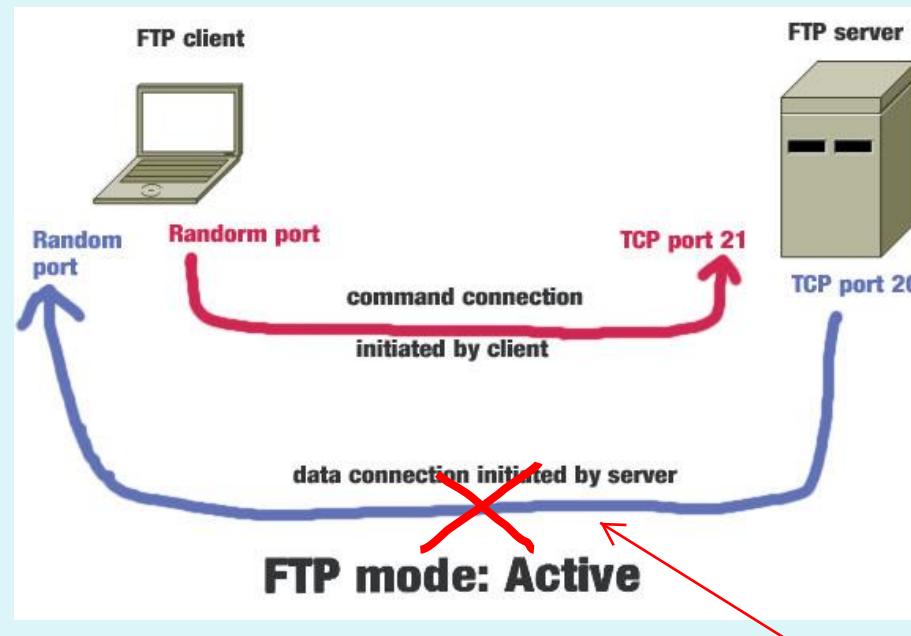
Prenos datotek (File Transfer Protocol)

- 2 ločeni TCP povezavi na FTP strežnik:
 - vrata TCP 21 (kontrolna povezava): ukazi za prenos datotek, uporabniško ime/geslo, menjava map,
...
 - vrata TCP 20 (prenos podatkov) na zahtevo odjemalca strežnik odpre povezavo, po kateri prenese podatke



Aktivni in pasivni način

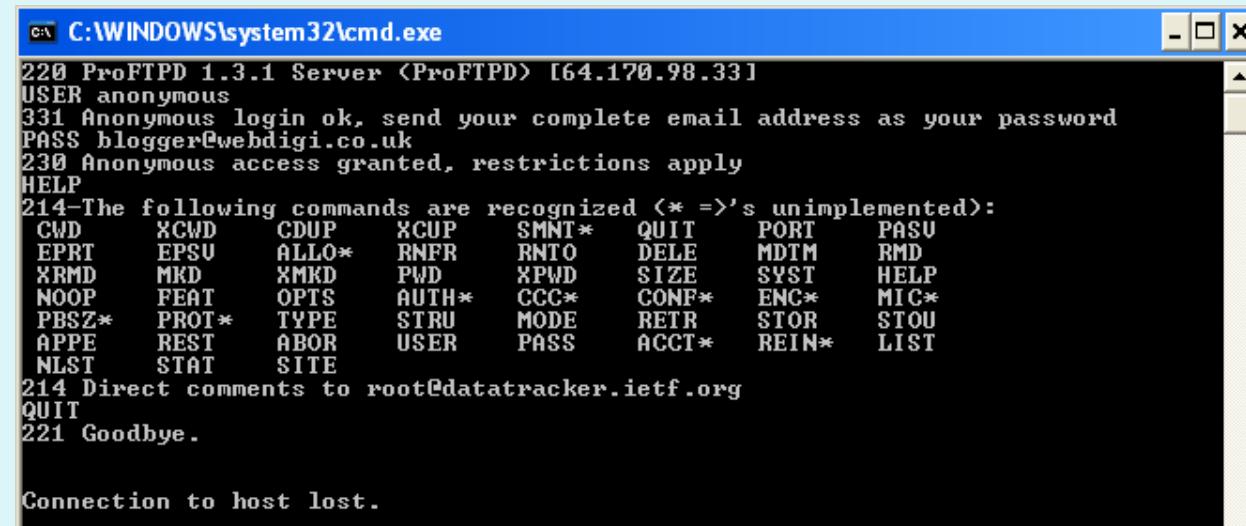
- težave s komunikacijo strežnik->odjemalec preko podatkovne povezave (port 20), če je med njima požarni zid ali NAT,
- rešitev: vzpostavitev podatkovne povezave od odjemalca k strežniku (**pasivni način**)



če imamo NAT/Firewall, ne deluje

Sporočila FTP protokola

- nadzorna povezava uporablja 7-bitne ASCII ukaze
- Ukazi in odgovori
 - USER ime
 - PASS geslo
 - LIST
 - RETR ime_datoteke
 - STOR ime_datoteke
- Odgovori strežnika
 - 331 Username OK, password required
 - 125 Data connection open, transfer starting
 - 452 Error writing file
 - 425 Can't open data connection



The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\system32\cmd.exe'. The session starts with a 220 response from a ProFTPD 1.3.1 Server. The user logs in anonymously. The server prompts for a password, which is provided as 'blogger@webdigi.co.uk'. An anonymous access grant is confirmed. A 'HELP' command is issued, listing recognized commands. The server also provides direct comments to 'root@datatracker.ietf.org'. Finally, a 221 Goodbye message is sent, followed by a 'Connection to host lost.' message.

```
220 ProFTPD 1.3.1 Server <ProFTPD> [64.170.98.33]
USER anonymous
331 Anonymous login ok, send your complete email address as your password
PASS blogger@webdigi.co.uk
230 Anonymous access granted, restrictions apply
HELP
214-The following commands are recognized (*=>'s unimplemented):
CWD XCWD CDUP XCUP SMNT* QUIT PORT PASV
EPRT EPSU ALLO* RNFR RNTO DELE MDTM RMD
XMDK MKD XMKD PWD XPWD SIZE SYST HELP
NOOP FEAT OPTS AUTH* CCC* CONF* ENC* MIC*
PBSZ* PROT* TYPE STRU MODE RETR STOR STOU
APPE REST ABOR USER PASS ACCT* REIN* LIST
NLST STAT SITE
214 Direct comments to root@datatracker.ietf.org
QUIT
221 Goodbye.

Connection to host lost.
```

<ftp://speedtest.tele2.net>

Elektronska pošta

SMTP, POP3, IMAP



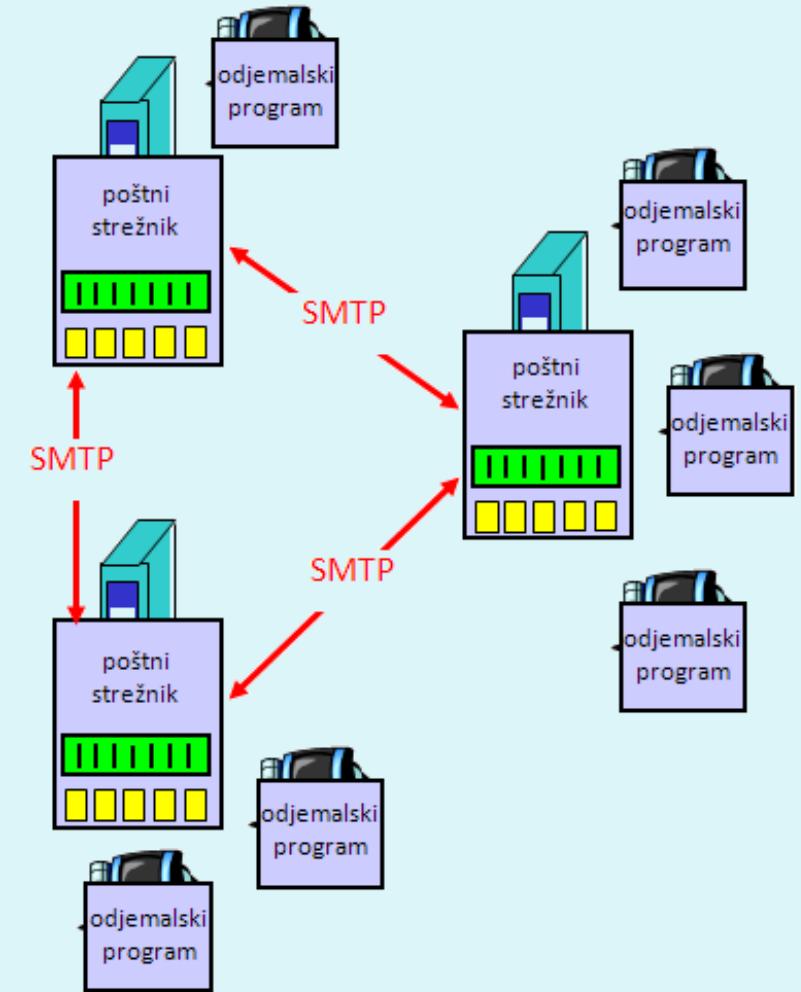
Elektronska pošta kot aplikacija

Prednosti:

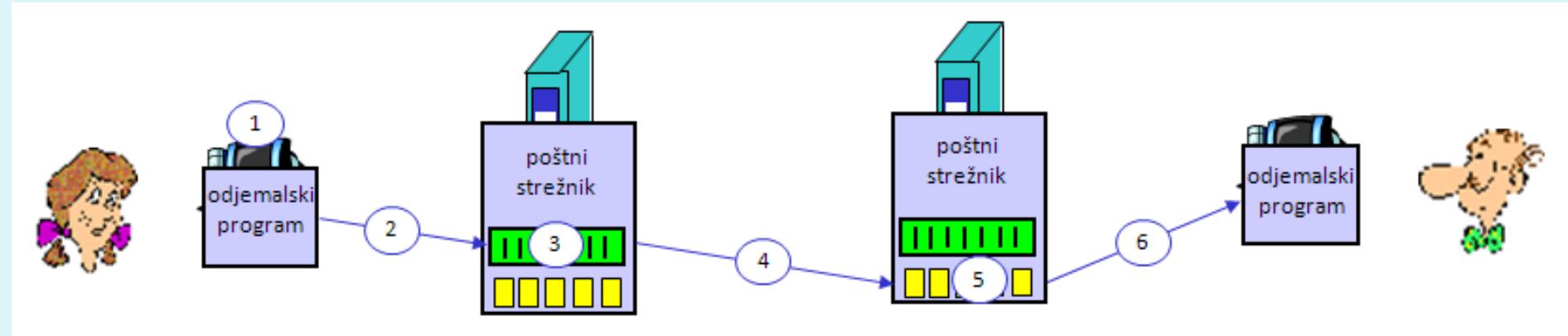
- hitra, poceni, enostavna dostava
- priponke, povezave, HTML

Arhitektura e-poštne aplikacije:

- **poštni strežniki**, ki hranijo:
 - poštne predale
 - imajo izhodno vrsto sporočil
- **odjemalski programi (user agent)**: tekstovni, grafični
- **protokol za prenos sporočil (SMTP)**



Pot elektronskega sporočila



1. uporabnik uporabi odjemalca, da sestavi sporočilo
2. odjemalec pošlje sporočilo strežniku
3. strežnik shrani sporočilo v izhodno vrsto
4. strežnik vzpostavi TCP povezavo s poštnim strežnikom prejemnika in prenese sporočilo
5. prejemnikov poštni strežnik shrani sporočilo v ustrezni poštni predal
6. prejemnik uporabi svoj odjemalski program za prenos in branje sporočila

Protokol SMTP

- *Simple Mail Transfer Protocol (SMTP)*, star več kot 30 let
- uporablja **TCP vrata 25**
- ukazi in telo sporočila morata biti kodirana z **7-bitnim ASCII!** Binarne priponke je potrebno prekodirati v ASCII in na prejemni strani nazaj v binarno obliko
- primer kodirane priponke

```
Delivered-To: ssmith@aspalliance.com
Received: by 10.220.195.140 with SMTP id ec12cs15030vcb;
          Thu, 10 Nov 2011 05:52:18 -0800 (PST)
Received: by 10.182.152.37 with SMTP id uv5mr1852779obb.53.1320933
          Thu, 10 Nov 2011 05:52:17 -0800 (PST)
Return-Path: <mailout@maillist.codeproject.com>
Received: from mail.maillist.codeproject.com (mail.maillist.codepr
          by mx.google.com with ESMTP id n2si1487012yhm.88.2011.11.1
          Thu, 10 Nov 2011 05:52:17 -0800 (PST)
Received-SPF: pass (google.com: domain of mailout@maillist.codepro
Message-Id: <4ebbd711.02b6ec0a.3783.6bd8SMTPIN ADDED@mx.google.com
Received: from jobs1 (jobs1.codeproject.com [192.168.5.180])
          by mail.maillist.codeproject.com (Postfix) with ESMTP id E
          for <ssmith@aspalliance.com>; Thu, 10 Nov 2011 08:51:45 -0
MIME-Version: 1.0
From: "The Code Project"
<mailout@maillist.codeproject.com>
To: "Steve Smith"
<ssmith@aspalliance.com>
Date: 10 Nov 2011 08:52:16 -0500
Subject: CodeProject | Daily News - Clean up after Visual Studio
Content-Type: text/html; charset=utf-8
Content-Transfer-Encoding: base64

PCFETONUWVBFIEhUTUwgUFVCTE1DICItLy9XM0MvL0RURCBIVE1MIDQuMDEgVHJh
bnNpdGlvbmFsLy9FTiIgImh0dHA6Ly93d3cudzMub3JnL1RSL2h0bWw0L2xvb3N1
LmR0ZCI+IA0KPGh0bWw+DQo8aGVhZD48dG10bGU+RGFpbHkgTmV3cyAgLSBDbGVh
biB1cCBhZnRlcibWaXN1YWwgU3R1ZG1vPC90aXRsZT4NCjxtZXRhIGH0dHAt2XF1
aXY9IkNvbnRlbnQtVHlwZSIgY29udGVudD0idGV4dC9odG1sOyBjaGFyc2V0PWlz
by04ODU5LTEiIC8+DQo8L2h1YWQ+DQo8IS0tMTgyMzMxMS0tPg0KPGJvZHk+DQo8
ZG12IHN0eWx1PSJkaXNwbGF50m5vbmU7Y29sb3I6d2hpGU1P11vdXIgbW9ybmlu
ZyB1cGRhdGU6IERhaWx5IE51d3MgIC0gQ2x1YW4gdXAgYWZ0ZXIgVm1zdWFsIFN0
dWRpbzvZG12Pg0KPHN0eWx1IHR5cGU9InR1eHQvY3NzIj4NCjwhLS0NCmJvZHks
THB1LCBwLGaxLGauLGa2LGazLGa0LGxvLGJab2NrcYVvrdGU1ca2ZvbnOr7mFr+Mx5
```

Primer SMTP komunikacije

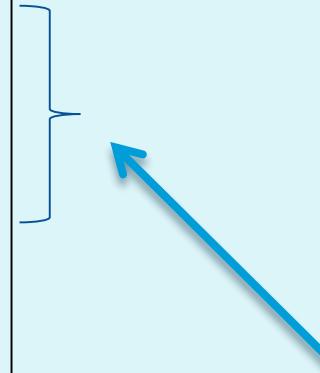
```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Oblika SMTP sporočila

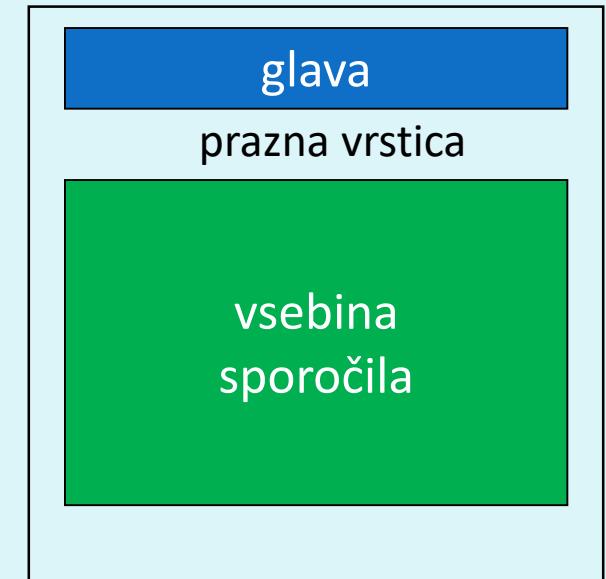
- sporočilo je sestavljeno iz glave, prazne vrstice in telesa (vsebine) sporočila,
- glava sporočila hrani podatke o sporočilu (razlikovanje od SMTP ukazov!)

```
From: darko@ucilnica.fri.uni-lj.si
To: jan@ucilnica.fri.uni-lj.si
Subject: Pomembna modrost

Zdravo, Jan!
Pomemben in pameten rek pravi,
da je potrpežljivost mati modrosti.
Zapomni si to :)
```



vrstice glave



Celotna glava poštnega sporočila

Received: from ns.fri.uni-lj.si ([212.235.188.18]) by fri-postar1.fri1.uni-lj.si with Microsoft SMTPSVC(6.0.3790.4675);
Sun, 13 Feb 2011 18:27:58 +0100

Received: from AG (unknown [212.235.188.19])
by ns.fri.uni-lj.si (Postfix) with ESMTP id 17586833A
for <zoran.bosnic@fri.uni-lj.si>; Sun, 13 Feb 2011 18:27:58 +0100 (CET)

Received: from localhost ([212.235.188.18]) by AG with Microsoft SMTPSVC(6.0.3790.4675);
Sun, 13 Feb 2011 18:26:55 +0100

X-Virus-Scanned: amavisd-new at fri.uni-lj.si

Received: from ns.fri.uni-lj.si ([127.0.0.1])
by localhost (ns.fri.uni-lj.si [127.0.0.1]) (amavisd-new, port 10024)
with ESMTP id CSPiVHX2mvZ8 for <zoran.bosnic@fri.uni-lj.si>;
Sun, 13 Feb 2011 18:26:52 +0100 (CET)

Received: from web4 ([198.115.93.97]) by web4.editorialmanager.com with Microsoft SMTPSVC(6.0.3790.4675);
Sun, 13 Feb 2011 12:26:39 -0500

MIME-Version: 1.0

From: "Knowledge and Information Systems" <srilakshmi.patrudu@springer.com>

Sender: em.kais.0.210cb1.358f57e1@editorialmanager.com

To: "Zoran Bosni?" <zoran.bosnic@fri.uni-lj.si>

Date: 13 Feb 2011 12:26:39 -0500

Subject: Decision on your Manuscript #KAIS-2487

Content-Type: text/plain; charset=iso-8859-1

Content-Transfer-Encoding: 7bit

Message-ID: <WEB40gIxRhrLjANJQ0w00000fef@web4.editorialmanager.com>

X-OriginalArrivalTime: 13 Feb 2011 17:26:39.0849 (UTC) FILETIME=[2C987D90:01CBCBA3]

Return-Path: em.kais.0.210cb1.358f57e1@editorialmanager.com

X-AntiVirus: checked (incoming) by AntiVir MailGuard (Version: 10.0.1.38; AVE: 8.2.4.166; VDF: 7.11.3.52)

oznake
prejemnih
strežnikov

podatki o
sporočilu

Primerjava SMTP in HTTP

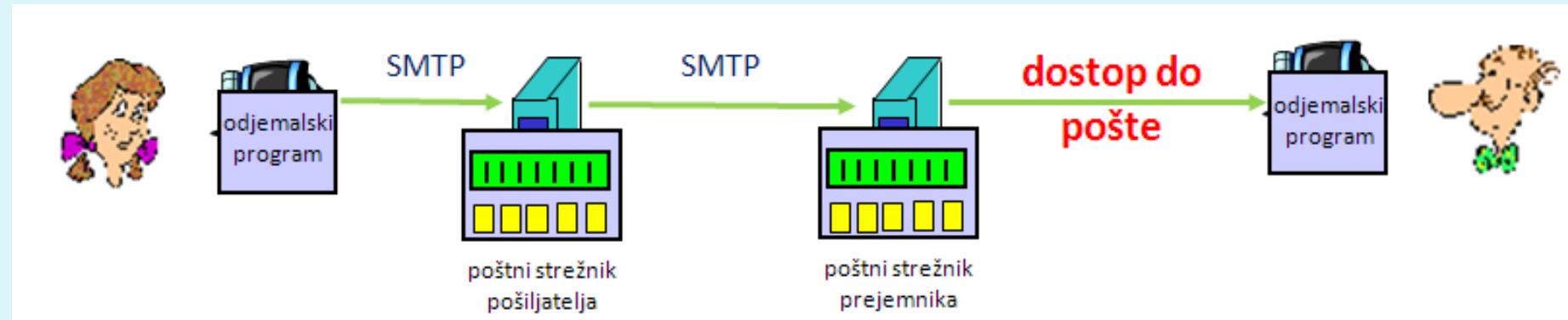
SMTP

- PUSH princip
- zahteva 7-bitni ASCII nabor znakov
- več objektov lahko poslanih v enem sporočilu
- uporablja vztrajne povezave

HTTP

- PULL princip
- lahko uporablja binarno kodiranje
- vsak objekt se pošilja v svojem sporočilu
- uporablja lahko vztrajne ali nevztrajne povezave

Dostop do poštnega predala



- SMTP: dostava pošte do prejemnikovega strežnika
- POP3, IMAP, HTTP: protokoli za prejem pošte s poštnega strežnika
 - POP: Post Office Protocol [RFC 1939]: avtentikacija in prenos
 - IMAP: Internet Mail Access Protocol [RFC 1730]: avtentikacija, prenos, organizacija po mapah, naprednejše storitve
 - HTTP: gmail, Hotmail, Yahoo! Mail, etc.

Primerjava protokolov

- **POP**
 - preprost, omejena funkcionalnost, TCP vrata 110
 - 3 faze: avtorizacija, prenos sporočil in oznak, posodabljanje (QUIT)
 - slabosti: lokalno urejanje pošte, ne hrani stanja med sejami
- **IMAP**
 - kompleksen, zahtevnejši, več funkcionalnosti
 - ✓ uporabnik lahko določi mape na strežniku
 - ✓ možen prenos le delov sporočil
 - Večja obremenitev strežnika
- **HTTP**
 - ✓ brskalnik, dostop od koderkoli, brezplačni ponudniki
 - mape kot pri IMAP
 - skripte na HTTP strežniku komunicirajo s poštnim strežnikom

POP3 ukazi - primer

```
S: +OK POP3 server ready
O: user zoranb                               uporabnik se predstavi
S: +OK
O: pass tralala                             nezaščiteno!
S: +OK user successfully logged on

O: list
S: 1 678                                     1. sporočilo je veliko 678B
S: .
O: retr 1                                    ni več sporočil
S: <vsebina sporočila>
S: .
O: dele 1                                    prenesi sporočilo 1
O: quit
S: +OK POP3 server signing off               briši sporočilo 1
```

Naslednjič gremo naprej!

- aplikacijska plast: DNS, P2P
- kriptografija

