

# 8. Aplikační vrstva

PB156: Počítačové sítě

Eva Hladká

*Slidy připravil: Tomáš Rebok*

Fakulta informatiky Masarykovy univerzity

jaro 2011

# Struktura přednášky

- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# Struktura přednášky

- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# L7. Aplikační vrstva – Přehled

## ISO / OSI

**Aplikační vrstva**  
Síťové aplikace

**Prezentační vrstva**  
Reprezentace dat

**Relační vrstva**  
Relace, meziuzlová komunikace

**Transportní vrstva**  
End-to-end spoje, zajištění spolehlivosti

**Síťová vrstva**  
Výběr cesty a IP (logické adresování)

**Vrstva datového spoje**  
MAC a LLC (fyzické adresování)

**Fyzická vrstva**  
Přenosová média, signály, přenos binárních dat

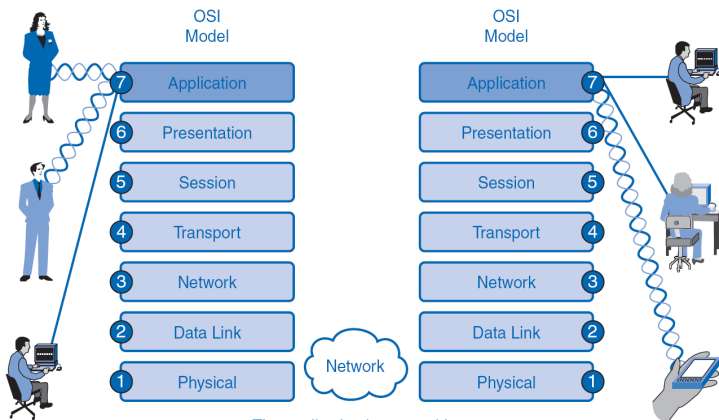
## Proč nestačí L4?

- z pohledu sítě stačí, z pohledu uživatele potřebujeme síťové aplikace

## Co nás nyní čeká. . .

- představení L7
- základní členění aplikací
- pohled programátora
- vybrané síťové aplikace

# L7 z pohledu sítě – kde se pohybujeme?



- aplikační programy – interface pro uživatele

# Struktura přednášky

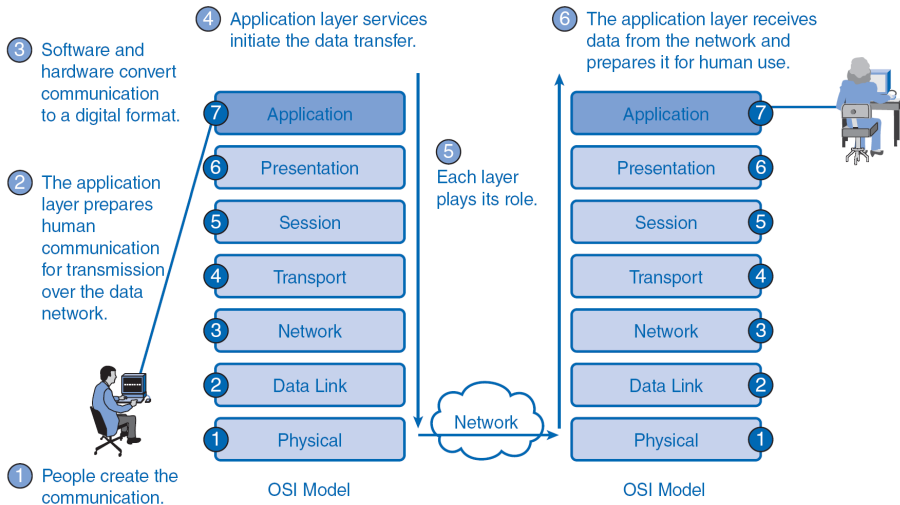
- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# Úvod I.

## aplikační vrstva:

- poskytuje služby pro *uživatele*:
  - aplikační programy (aplikace) specifické pro požadovaný účel
    - např. elektronická pošta, WWW, DNS, atd. atd.
  - aplikace = hlavní smysl existence počítačových sítí
- zahrnuje *síťové aplikace/programy a aplikační protokoly*
  - aplikační protokoly (HTTP, SMTP, atd.) jsou **součástí** síťových aplikací (web, email)
    - nejedná se o aplikace samotné
    - protokoly definují formu komunikace mezi komunikujícími aplikacemi
  - aplikační protokoly definují:
    - typy zpráv, které si aplikace předávají (*request/response*)
    - syntaxi přenášených zpráv
    - sémantiku přenášených zpráv (jednotlivých polí)
    - pravidla, kdy a jak aplikace zprávy vysílají

# Úvod II.





# Struktura přednášky

- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací**
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# Základní členění aplikací

*Dle využitého komunikačního modelu:*

- Client-Server model
- Peer-to-peer model

*Dle přístupu k informacím:*

- pull model
- push model

*Dle nároků na počítačovou síť:*

- aplikace s nízkými nároky na přenosovou síť
- aplikace s vysokými nároky na přenosovou síť

# Komunikační modely – *Client-Server* vs. *Peer-to-peer* I.

## Client-Server

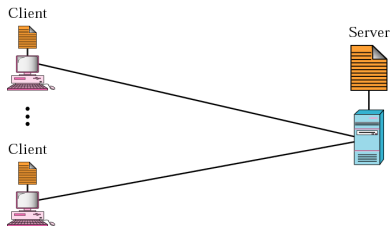
- komunikace iniciována klientem (klient = aplikační program ovládaný uživatelem)
- po ustavení komunikačního kanálu klient zasílá požadavky na server, ten mu odpovídá (mechanismus *request-response*)
- po ukončení komunikace je komunikační kanál uzavřen
- (centralizace zdrojů)
  
- valná většina aplikací v Internetu (WWW, FTP, DNS, SSH, ...)

## Peer-to-peer

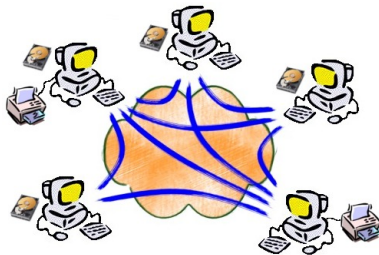
- jednotliví klienti spolu komunikují přímo (uzly jsou si rovnocenné)
- každý uzel poskytuje své zdroje (výpočetní síla, úložná kapacita, atp.) ostatním uzlům
- každý uzel využívá zdrojů poskytovaných ostatními uzly
- (decentralizace zdrojů)
  
- např. sdílení souborů (Gnutella, G2, FastTrack), Skype, VoIP, atp.

# Komunikační modely – *Client-Server* vs. *Peer-to-peer* II.

## Client-Server



## Peer-to-peer



# Komunikační modely – *Client-Server*

## Tenký (*Thin*) vs. Tlustý (*Fat*) klient

### Tenký (**Thin**) klient

- aplikace, u nichž se na straně klienta vykonává minimum aplikační logiky (většina se vykonává na straně serveru)
  - větší hardwarové nároky na stranu serveru a na komunikaci
- + jednodušší, menší nároky na HW (může tak být levnější)
- – menší škálovatelnost (příliš mnoho práce dělá server), většinou vyšší objemy přenášených dat, existence *Single point of failure* (server)
- *příklad*: vzdálené terminály

### Tlustý (**Fat**) klient

- přesný opak tenkého klienta – většina aplikační logiky se vykonává na straně klienta
  - větší hardwarové i softwarové nároky na klienta
- + menší nároky na server ( $\Rightarrow$  dobrá škálovatelnost), většinou nižší objem přenesených dat, možnost práce *offline*
- – komplexní provedení i instalace, značná spotřeba lokálních zdrojů (CPU, paměť, disk)
- *příklad*: Firefox

# Přístup k informacím – *Pull model vs. Push model*

## Pull model

- přenos dat iniciován klientem (forma požadavek-odpověď)
- např. webové prohlížeče
- *vlastnosti:*
  - asymetrický datový tok
  - rozmanité požadavky na propustnost

## Push model

- přenos dat iniciován serverem automaticky na základě znalosti uživatelského profilu (požadavků)
- např. streaming multimedií (IPTV)
- *vlastnosti:*
  - jednosměrný datový tok
  - definované (a stálé) požadavky na propustnost (a zpoždění, jitter, atp.)

## Nároky na počítačovou síť – nízké vs. vysoké

Základní parametry sítě pohledem aplikací:

- *ztrátovost (loss)* – pravděpodobnost ztráty (poškození) přenášených dat
- *propustnost (bandwidth)* – objem přenesených dat za časovou jednotku
- *časová omezení (timing)* – *zpoždění (delay)* (doba nutná pro přenos dat po síti) a *kolísání zpoždění (jitter)*

Application	Data Loss	Bandwidth	Time sensitive?
file transfer	no loss	elastic	no
electronic mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: few Kbps to 1Mbps video: 10's Kbps to 5 Mbps	yes: 100's of msec
stored audio/video	loss-tolerant	same as interactive audio/video	yes: few seconds
interactive games	loss-tolerant	few Kbps to 10's Kbps	yes: 100's msec
financial applications	required	elastic	yes and no

# Struktura přednášky

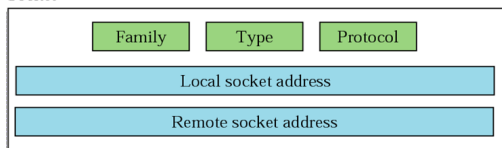
- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora**
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace



# Aplikační vrstva – pohled programátora

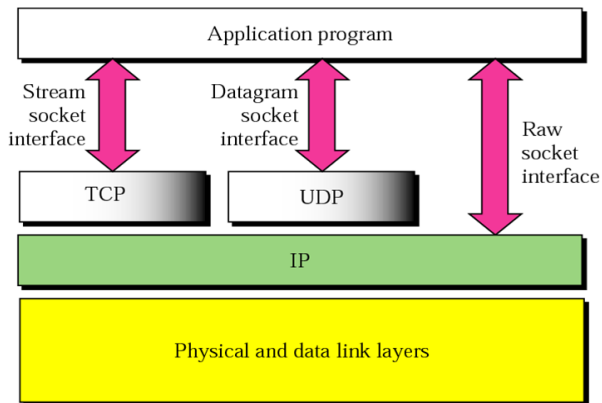
- aplikace spolu komunikují skrze tzv. *sockety*
  - struktura, která jednoznačně popisuje komunikující aplikaci
  - jsou nezbytné na obou komunikujících stranách
- *socket* definuje:
  - **Family** – třída využitého protokolu (IPv4, IPv6, ...)
  - **Type** – proudový (stream, TCP), datagramový (UDP) či základní (raw, ICMP)
  - **Protocol** – obvykle nastaveno na 0 pro TCP a UDP
  - **Lokální adresa socketu** – kombinace lokální IP a čísla portu lokální aplikace
  - **Vzdálená adresa socketu** – kombinace IP a čísla portu vzdálené aplikace (komunikující strany)

Socket



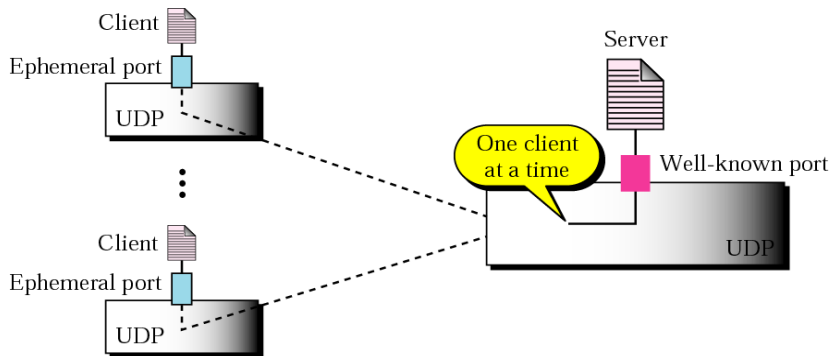
# Aplikační vrstva – pohled programátora

## Typy socketů



# Aplikační vrstva – pohled programátora

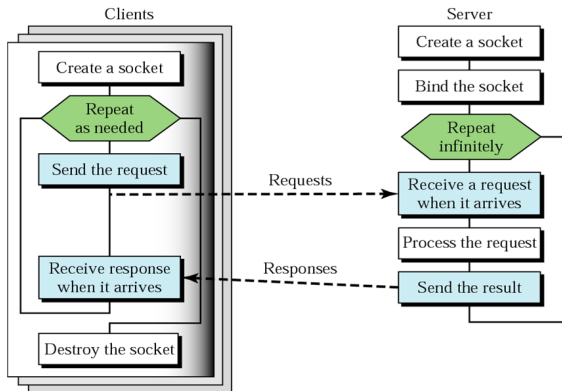
Obsluha socketu pro nespojovanou službu (UDP aplikace) I.



# Aplikační vrstva – pohled programátora

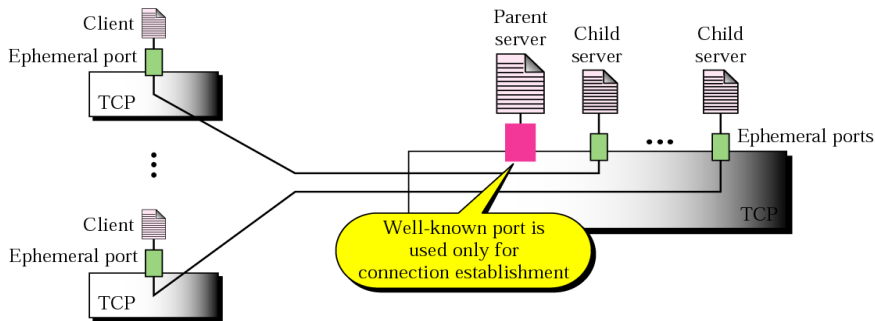
## Obsluha socketu pro nespojovanou službu (UDP aplikace) II.

Each server serves many clients but handles one request at a time.



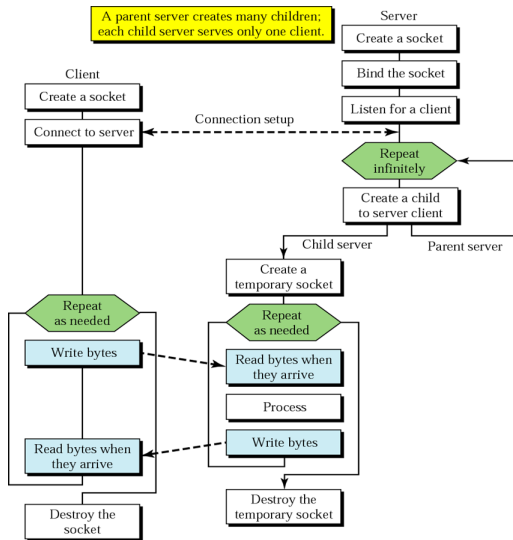
# Aplikační vrstva – pohled programátora

Obsluha socketu pro spojovanou službu (TCP aplikace) I.



# Aplikační vrstva – pohled programátora

## Obsluha socketu pro spojovanou službu (TCP aplikace) II.



# Struktura přednášky

- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace**
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# Vybrané síťové aplikace

- *Jmenná služba – DNS*
- *World Wide Web – HTTP*
- *Elektronická pošta – SMTP*
- *Přenos souborů – FTP*
- *Multimediální přenosy – RTP, RTCP*



# Jmenná služba – DNS

**Domain Name System (DNS)** – služba pro překlad doménových jmen na IP adresy a zpět

- např. `aisa.fi.muni.cz` ↔ `147.251.48.1`

V začátcích Internetu řešeno za pomoci tzv. **host** souborů

- soubory s dvojicemi *doménové jméno*, *IP adresa*
- neškálovatelné řešení
  - s růstem Internetu nemožné mít tyto soubory (obsahující doménová jména celého Internetu) na každém uzlu
  - navíc v nich vyhledávat, aktualizovat, ...

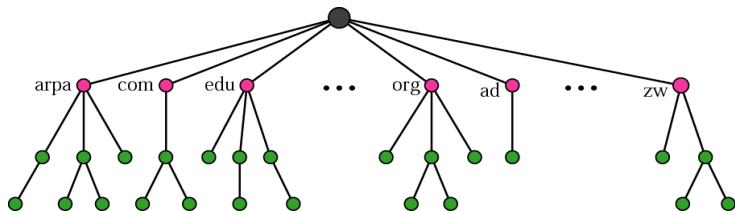
⇒ **Domain Name System (DNS)**

# Jmenný prostor

- *Jmenný prostor*  $\approx$  způsob pojmenování předmětných entit
- 2 základní varianty:
  - *plochý jmenný prostor* – jména bez jakékoliv vnitřní struktury
    - např. `mujRouterDomaVBrne`
    - hlavní nevýhoda: nemožnost využití ve velkém systému (nutnost centrální kontroly pro zamezení duplicit)
  - *hierarchický jmenný prostor* – jména s hierarchickou vnitřní strukturou
    - jména sestávají z několika částí, každá s definovaným významem
    - např. `mujRouter.DomaVBrne.cz`
    - hlavní výhoda: možnost decentralizace správy (přidělování a kontroly) jmen  
(zodpovědnost vždy za určitou (pod)část doménového jména)

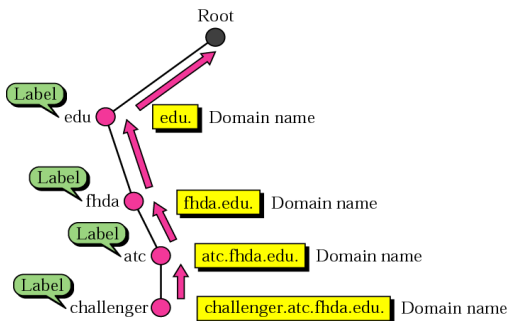
# Jmenný prostor Internetu I.

- *Jmenný prostor Internetu* – **doménový jmenný prostor (Domain Name Space)**
  - varianta hierarchického uspořádání
  - struktura „invertovaného stromu“
  - maximální počet úrovní = 128



## Jmenný prostor Internetu II.

- každému uzlu přidělena tzv. *jmenovka (label)* a *doménové jméno (domain name)*
  - label* – řetězec (max. 63 znaků) popisující daný uzel
    - label* kořenového uzlu je prázdný řetězec
  - domain name* – sekvence jmenovek (oddělená znakem „.“) od daného uzlu ke kořenovému
    - plné doménové jméno vždy končí znakem „.“



## Jmenný prostor Internetu III.

- **Fully Qualified Domain Name (FQDN)** – plné doménové jméno obsahující všechny značky až ke kořenovému uzlu, končí znakem „.“
  - např. `aisa.fi.muni.cz.`
  - koncový znak „.“ se běžně neuvádí, interně (v rámci DNS) se však používá (pro rozlišení FQDN od PQDN)
- **Partially Qualified Domain Name (PQDN)** – doménové jméno, které začíná na daném uzlu, ale neobsahuje všechny značky až ke kořenovému uzlu
  - k PQDN se pak přidává *suffix*, který slouží pro získání FQDN
  - např. PQDN `aisa.fi` v kontextu FQDN `muni.cz.` (= suffix) označuje uzel s FQDN `aisa.fi.muni.cz.`

FQDN

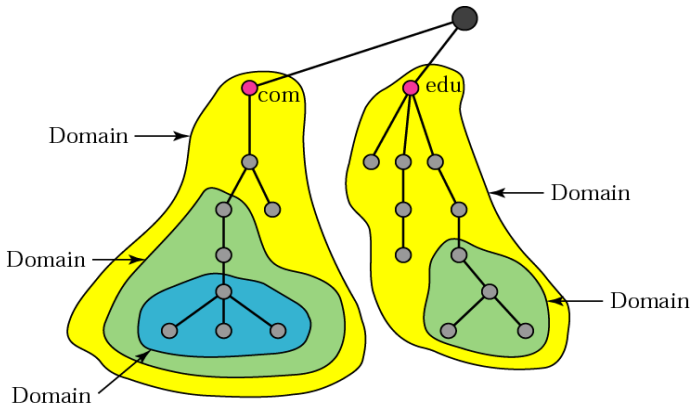
```
challenger.atc.fhda.edu.  
cs.hmme.com.  
www.funny.int.
```

PQDN

```
challenger.atc.fhda.edu  
cs.hmme  
www
```

# Jmenný prostor Internetu IV.

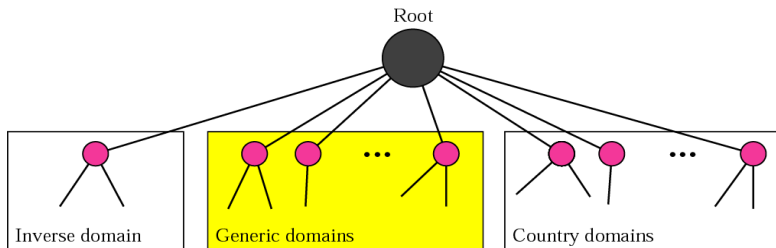
- **Doména** – podstrom doménového jmenného prostoru
  - jménem domény je doménové jméno jejího kořenového uzlu



# Domény v Internetu

Jmenný prostor Internetu rozdělen na 3 základní typy domén:

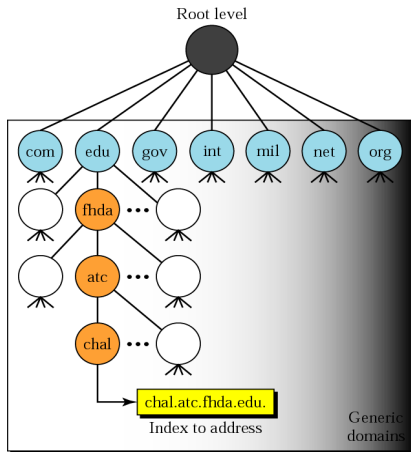
- *základní (generic) domény*
- *národní (country) domény*
- *reverzní (inverse) domény*



# Domény v Internetu

## Základní domény

Základní domény (*Generic Domains*) definují uzly podle jejich povahy (chování)





# Domény v Internetu

## Základní domény – Tabulka I.

Label	Description
<b>com</b>	Commercial organizations
<b>edu</b>	Educational institutions
<b>gov</b>	Government institutions
<b>int</b>	International organizations
<b>mil</b>	Military groups
<b>net</b>	Network support centers
<b>org</b>	Nonprofit organizations

# Domény v Internetu

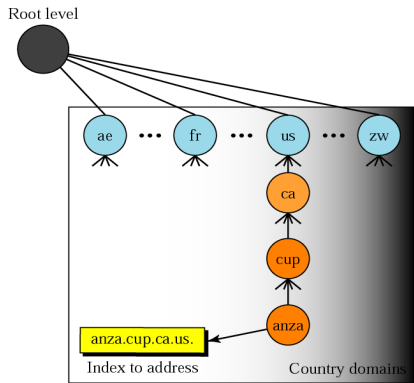
## Základní domény – Tabulka II.

Label	Description
<b>aero</b>	Airlines and aerospace companies
<b>biz</b>	Businesses or firms (similar to com)
<b>coop</b>	Cooperative business organizations
<b>info</b>	Information service providers
<b>museum</b>	Museums and other nonprofit organizations
<b>name</b>	Personal names (individuals)
<b>pro</b>	Professional individual organizations

# Domény v Internetu

## Národní domény

Národní domény (*Country Domains*) definují uzly podle jejich příslušnosti ke státu. Na první úrovni jsou využity dvoupísmenné zkratky státu (cz, sk, ca, us, ...)

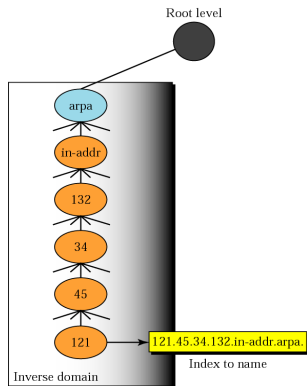


# Domény v Internetu

## Reverzní/Inverzní domény

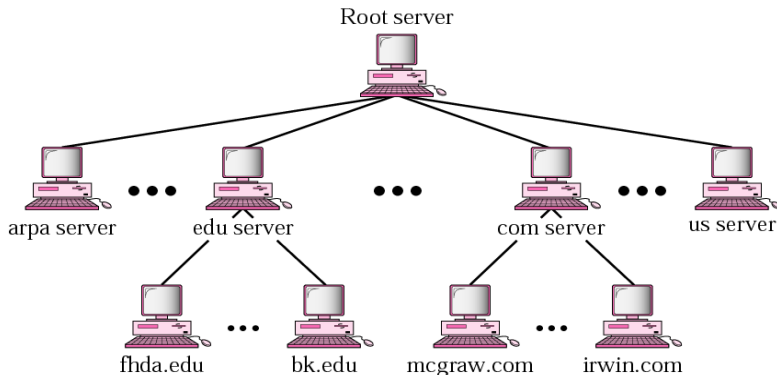
Reverzní/inverzní domény (*Inverse Domains*) slouží pro mapování IP adres na doménová jména

- definovaná struktura: převrácená IP adresa + identifikátor *in-addr* (*inverse address*) (IPv4) nebo *ip6* (IPv6) + identifikátor *arpa* (z historických důvodů)



# Hierarchie jmenných serverů I.

- „*Jak spravovat obrovské množství doménových jmen?*“
  - distribucí přes více *DNS serverů*
  - každý server pak je zodpovědný za určenou doménu/subdoménu



## Hierarchie jmenných serverů II.

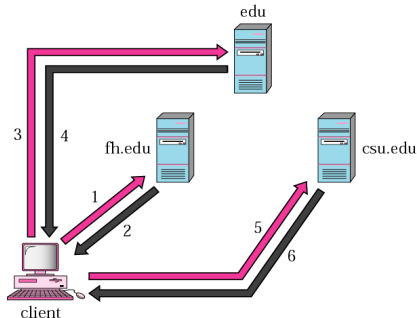
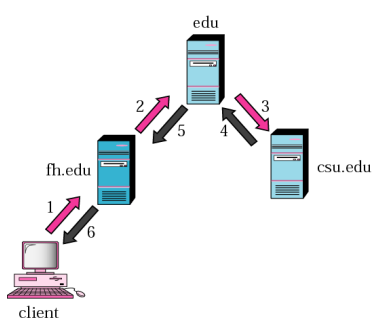
Rozeznáváme:

- *kořenové DNS servery* – obsahují informace o tzv. *top-level doménách* (základní, národní a reverzní domény)
  - aktuálně 13 serverů rozmístěných po světě
- *primární DNS servery* – servery spravující informace o určité doméně či její části
  - spravované domény/části domén se nazývají *zóny*
- *sekundární DNS servery* – redundantní servery získávající informace o zónách od primárních serverů
  - úkolem zvýšení škálovatelnosti a redundance při výpadku
- *cacheovací DNS servery* – servery sloužící pro zkrácení doby odpovědi na opakující se dotazy
  - ukládají si informace o posledních provedených dotazech/odpovědích
  - *Pozn.:* mechanismus **cache serverů** využit i pro jiné aplikace (např. WWW)

# Překlad adres/doménových jmen

Proces překladu doménových jmen na IP adresy (či zpět) se nazývá *name-address resolution*

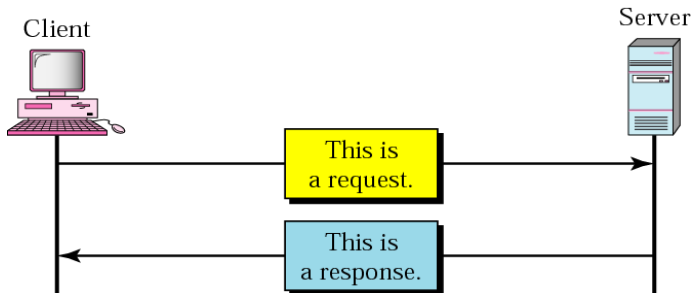
- klient zasílá požadavek na překlad nejbližšímu DNS serveru
  - UDP nebo TCP protokolem, v obou případech na port 53
- ten mu buď rovnou odpoví nebo:
  - 1 se zeptá dále – tzv. *rekurzivní chování*
  - 2 klienta přeměruje – tzv. *iterativní chování*



# World Wide Web – HTTP

**HyperText Transfer Protocol (HTTP)** – protokol pro přístup k datům na World Wide Webu (WWW)

- přenášená data mohou být ve formě textu, hypertextu, audia, videa, atp.
- základní idea: klient vysílá požadavek, WWW server zasílá odpověď
  - komunikace TCP protokolem na portu 80





# World Wide Web – HTTP

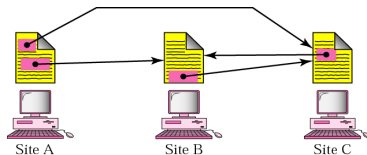
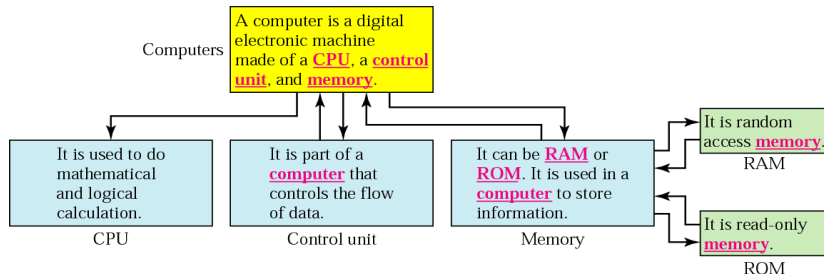
## Hypertext

*Co je to „hypertext“?*

# World Wide Web – HTTP

## Hypertext

Co je to „hypertext“?



# World Wide Web – HTTP

## Uniform Resource Locator (URL)

Součástí požadavku je tzv. *Uniform Resource Locator (URL)*

- standardní mechanismus pro specifikaci „čehokoliv“ na Internetu
- definuje zdroj, který chce klient získat
- součástí URL je:
  - *method* – metoda (protokol), který má být využit pro přístup k odkazovanému zdroji
  - *host* – uzel, kde se odkazovaná informace nachází (kde má být vyhledána)
  - *port* – volitelná součást, pokud je využit jiný než standardní port
  - *path* – cesta, kde se odkazovaná informace nachází (+ případně další informace (parametry))

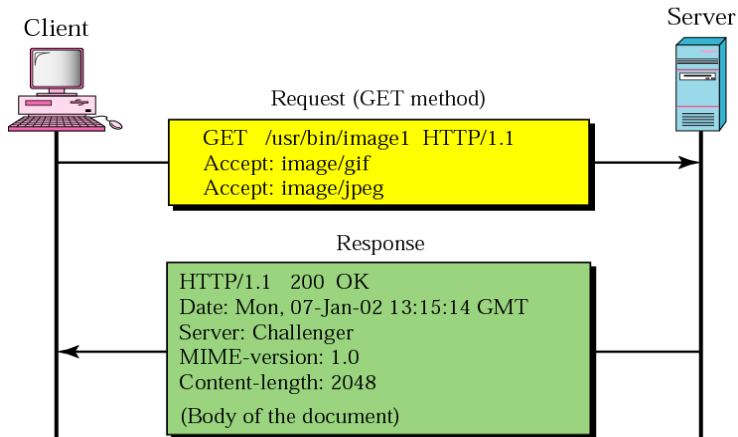
URL

Uniform resource locator



# World Wide Web – HTTP

## Příklad



# World Wide Web – HTTP

## Persistentní vs. Nepersistentní spojení

### *Nepersistentní spojení (Nonpersistent Connection)*

- samostatné TCP spojení uzavřeno pro každý požadavek/odpověď
  - 1 klient otevírá TCP spojení a zasílá požadavek
  - 2 server zasílá odpověď a uzavírá spojení
  - 3 klient zpracovává přijatá data a potvrzuje uzavření spojení
- pro získání  $N$  souborů (stránek) je zapotřebí ustavit/uzavřít  $N$  spojení
- standardní mechanismus protokolu HTTP verze 1.0

### *Persistentní spojení (Persistent Connection)*

- TCP spojení přetrvává delší dobu
  - typicky do vypršení definovaného timeoutu či uzavření spojení
  - jedno spojení je tak možno využít pro získání více souborů (stránek)
- standardní mechanismus protokolu HTTP verze 1.1

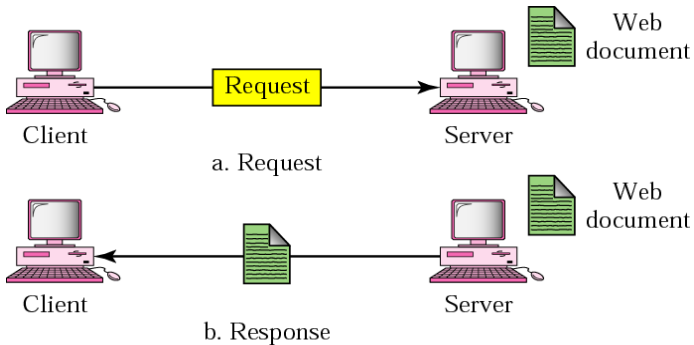
# World Wide Web – Dokumenty

Základní kategorie WWW dokumentů:

- *statické* – na serveru uložené dokumenty s pevným obsahem
  - např. HTML dokumenty
- *dynamické* – neexistují v předem definovaném formátu; jsou tvořeny webovým serverem dle požadavků klienta
  - např. CGI skripty
- *aktivní* – serverem poskytnuté programy spouštěné na straně klienta
  - např. JAVA aplikace

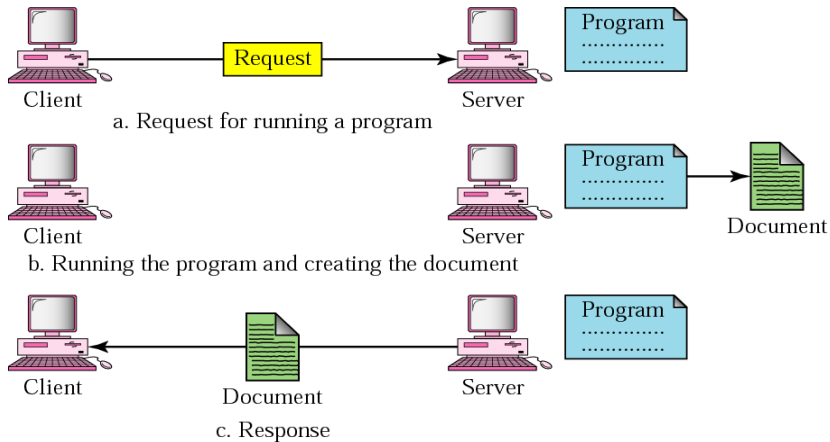
# World Wide Web – Dokumenty

## Statické dokumenty – ilustrace



# World Wide Web – Dokumenty

## Dynamické dokumenty – ilustrace



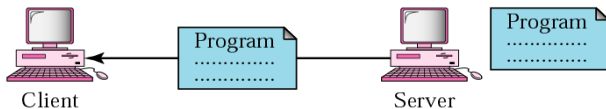


# World Wide Web – Dokumenty

## Aktivní dokumenty – ilustrace



a. Request for a copy of a program



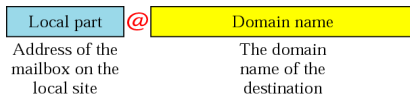
b. Sending a copy of the program



c. Running the program and creating the document

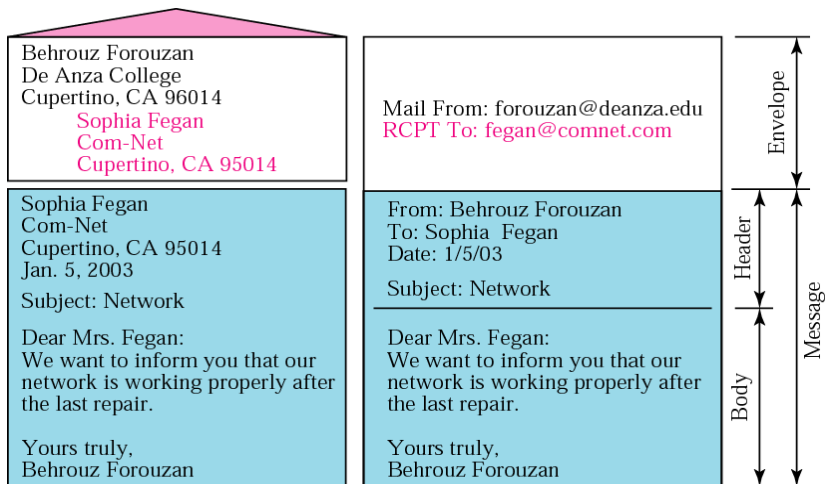
# Elektronická pošta – SMTP

- **Simple Mail Transfer Protocol (SMTP)** – standardní mechanismus pro posílání elektronické pošty (*electronic mail, email*) v Internetu
- struktura SMTP emailové zprávy:
  - *obálka (envelope)* – obsahuje adresu odesílatele, adresu příjemce a další případné informace
  - *vlastní zpráva (message)* – dělí se na hlavičky a tělo zprávy
    - hlavičky – definují odesílatele, příjemce, předmět zprávy, ...
    - tělo zprávy – vlastní přenášená zpráva
- *emailové adresy*:
  - skládají se z tzv. *lokální části* a *doménového jména*
    - lokální část definuje jméno souboru, kam je doručována pošta předmětného uživatele (tzv. *mailbox*)
    - doménové jméno dané organizace
  - doručení emailu probíhá na základě emailových adres uvedených v **obálce zprávy**



# Elektronická pošta – SMTP

## Příklad emailové zprávy



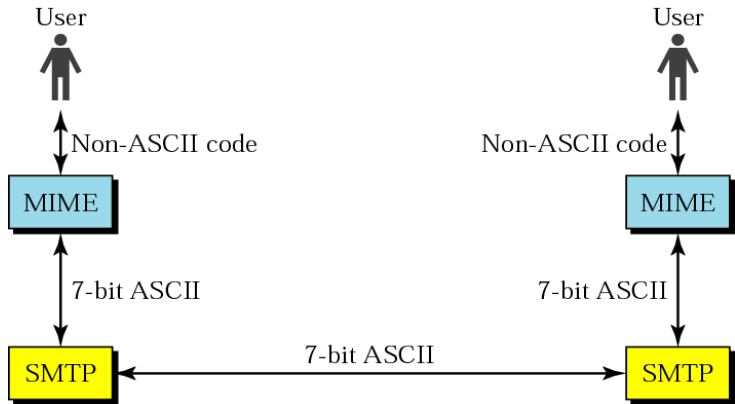
# Elektronická pošta – SMTP

## *Multipurpose Internet Mail Extensions (MIME) I.*

- SMTP je velmi jednoduchým protokolem
  - standardně dokáže přenášet jen textové zprávy v 7-bitovém ASCII kódování
    - ⇒ nedokáže přenášet zprávy se specifickými znaky/diakritikou ani binární soubory (obrázky, videa, atp.)
    - pro přenos non-ASCII dat SMTP protokolem navržen doplňkový protokol **Multipurpose Internet Mail Extensions (MIME)**
- *Multipurpose Internet Mail Extensions (MIME)*
  - MIME převádí non-ASCII data na odesílací straně do ASCII dat, které jsou skrze SMTP přeneseny příjemci; tam jsou ASCII data MIME protokolem opět převedeny do non-ASCII dat
  - nejedná se o emailový protokol (nenahrazuje SMTP, jen jej rozšiřuje)

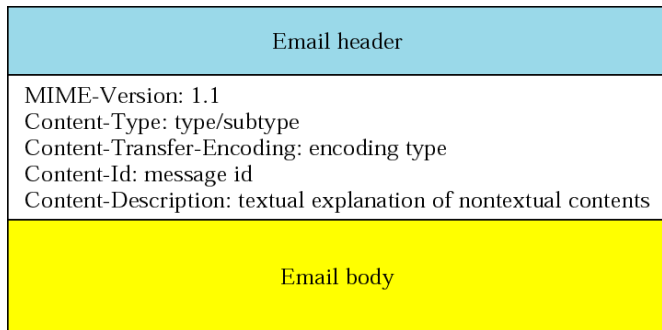
# Elektronická pošta – SMTP

## *Multipurpose Internet Mail Extensions (MIME) II.*



# Elektronická pošta – SMTP

*Multipurpose Internet Mail Extensions (MIME) – příklad hlaviček*



**MIME header**

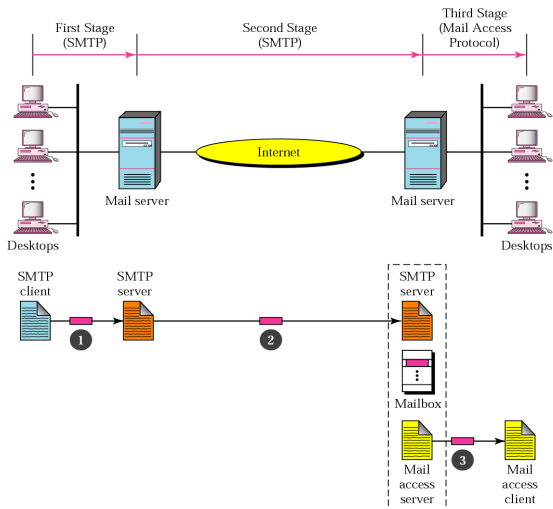
# Elektronická pošta – Doručení elektronických zpráv

Doručení elektronické zprávy se skládá ze 3 fází:

- 1 *doručení emailu lokálnímu poštovnímu serveru (mailserveru)*
  - poštovní klient (*Mail Transfer Agent, MTA*) ustaví TCP spojení (port 25) s poštovním serverem
  - po předání zprávy (s využitím SMTP protokolu) spojení uzavře
- 2 *předání emailu cílovému poštovnímu serveru*
  - lokální mailserver předá emailovou zprávu cílovému mailserveru (SMTP protokolem)
- 3 *předání/čtení emailu cílovým poštovním klientem*
  - iniciováno cílovým uživatelem (poštovním klientem) s využitím protokolu *POP3* či *IMAP4*

# Elektronická pošta – Doručení elektronických zpráv

## Ilustrace doručení emailové zprávy





# Elektronická pošta – Doručení elektronických zpráv

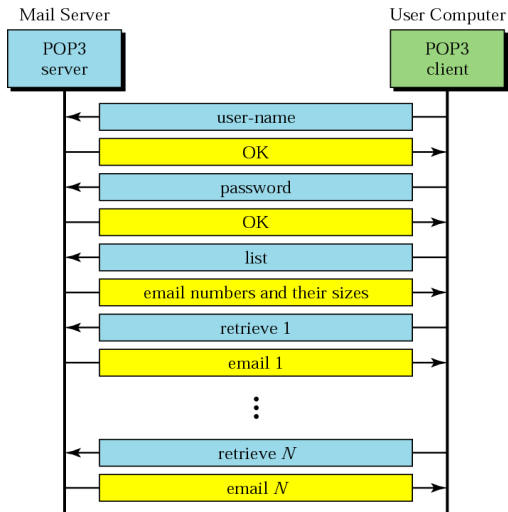
## *Post Office Protocol version 3 (POP3) I.*

### *Post Office Protocol version 3 (POP3)*

- jednoduchý protokol pro přístup k emailovým zprávám na poštovním serveru
- poštovní klient (POP3 klient) uzavírá spojení s poštovním serverem (POP3 server)
  - s využitím protokolu TCP, port 110
  - po autentizaci klienta server předává přijaté emailové zprávy
  - zprávy následně smazány či ponechány v mailboxu

# Elektronická pošta – Doručení elektronických zpráv

## Post Office Protocol version 3 (POP3) II.



# Elektronická pošta – Doručení elektronických zpráv

## *Internet Mail Access Protocol version 4 (IMAP4)*

### Nevýhody POP3:

- předpokládá, že při každém připojení klienta k mailserveru dojde k vyprázdnění celého mailboxu
  - nepohodlná práce při přístupu k emailu z více poštovních klientů
- neumožňuje organizaci emailu na serveru
- neumožňuje nahlédnutí do emailu před jeho vlastním stažením

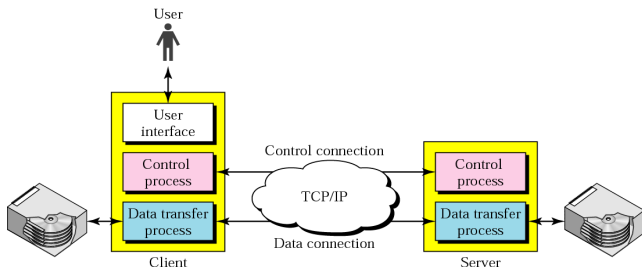
### *Internet Mail Access Protocol version 4 (IMAP4)*

- protokol podobný POP3 s mnoha vylepšeními:
  - nahlédnutí do emailu před jeho stažením
  - podpora částečných stažení emailů
  - podpora práce s mailboxy na poštovním serveru
  - atd.

# Přenos souborů – FTP

**File Transfer Protocol (FTP)** – standardní mechanismus Internetu určený pro přenos souborů mezi uzly

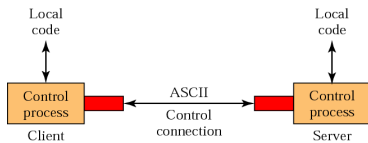
- oproti jiným *klient-server* aplikacím FTP klient s FTP serverem ustavuje **dvě samostatná TCP spojení**
  - řídicí zprávy zasílány tzv. *out-of-band*
- ① *řídicí spojení* (TCP, port 21)
  - udržováno po celou dobu ustavené relace
- ② *datové spojení* (TCP, port 20)
  - otevíráno/zavíráno pro každý přenášený soubor



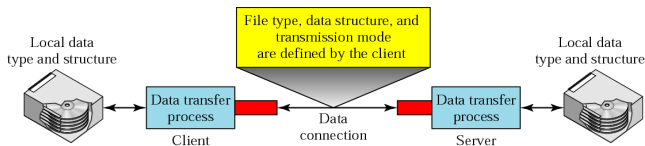
# Přenos souborů – FTP

## Řídící vs. datová komunikace

- *řídící komunikace* – přenos požadavků klienta a odpovědí serveru
  - domluva na parametrech spojení
    - typ souboru (textový vs. binární), vnitřní struktura souboru (obvykle bez struktury) a přenosový mód (proudový, blokový, komprimovaný)
    - nezbytné pro překonání heterogenity komunikujících stran

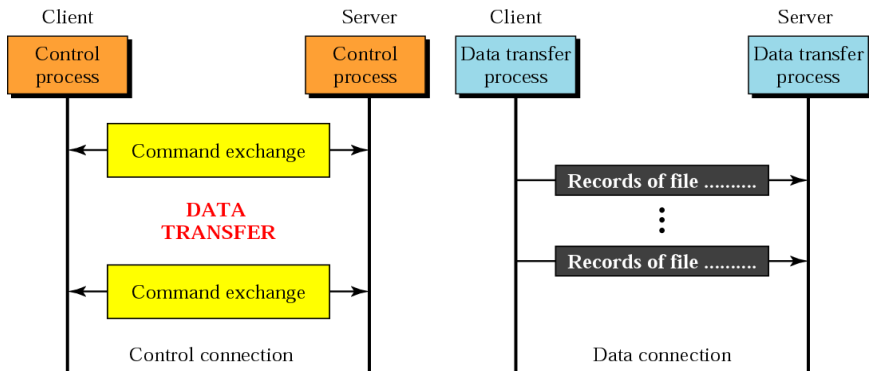


- *datová komunikace*



# Přenos souborů – FTP

## Příklad – uložení souboru na server



# Multimediální přenosy – Úvod

## Multimédia a datové sítě

- posun od využití sítě pro přenos statických dat (emaily, dokumenty, obrázky, ...) k přenosu dynamických dat (přenosy audia&videa)
- vyžadují relativně velké objemy přenášených dat
- specifické nároky na přenos (chybovost, latence, jitter, atp.)
  - požadavky na přenos zásadně ovlivňují možnosti zpracování

### *Aplikace multimediálních přenosů:*

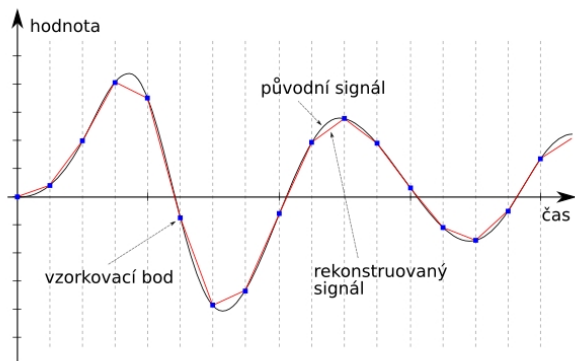
- Streaming uloženého audia/videoa
- *Streaming live audia/videoa*
  - doručování multimediálního obsahu, který vzniká živě během streamování
- *Videokonference, Internetová telefonie*
  - aplikace požadující zcela konkrétní vlastnosti přenosu (např. minimální end-to-end zpoždění)
  - jednoznačný požadavek na interaktivitu

# Zpracování zvuku

- *zvuk* – podélné mechanické vlnění v látkovém prostředí (vzduch), které je schopno vyvolat v lidském uchu sluchový vjem
  - akvizice mikrofonom  $\Rightarrow$  analogový signál spojitý v čase
- zpracování zvuku:
  - vzorkování a kvantování – převod analogového signálu do digitálního
  - zpracování digitálních dat – použití filtrů (ekvalizace, odstranění šumu/echa, atp.)
  - komprese – snížení datového objemu
    - pro audio data není nezbytná (objem audio dat je relativně malý)
    - MP3 (MPEG audio layer 3), OGG, WMA (Windows Media Audio), RA (Real Audio), ...

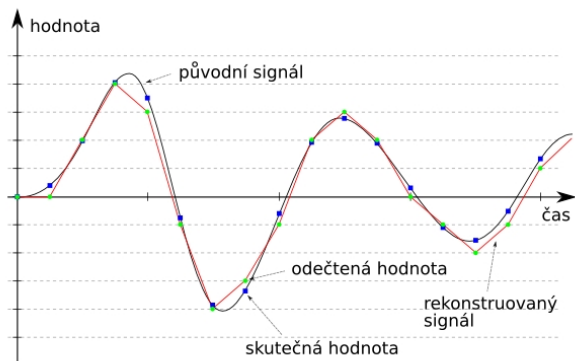


# Zpracování zvuku – vzorkování



- *vzorkování* = **odebírání vzorku signálu** v definovaných časových intervalech (*vzorkovací frekvence*)
  - převádí spojité časové průběh signálu na diskrétní reprezentaci

# Zpracování zvuku – kvantování



- **kvantování** = diskrétní reprezentace **hodnoty intenzity zvuku** v okamžiku odebrání vzorků
  - rozdělení svislé osy zvukové křivky na diskrétní hodnoty

# Zpracování obrazu

- *obraz* – elektromagnetické vlnění s velmi úzkou šířkou spektra (viditelné světlo) odražené od objektů v okolí a dopadající na světlocitlivě buňky sítnice oka
  - akvizice videokamerou  $\Rightarrow$  sekvence diskrétních obrázků
- zpracování obrazu:
  - vzorkování a kvantování – převod analogového elektromagnetického vlnění do digitálního signálu
    - při akvizici obraz rozdělén na diskétní vzorky (typicky  $768 \times 576$  bodů,  $1920 \times 1080$  bodů, atp.)
    - úkolem kvantování je ohodnotit barvu/jas/intenzitu jednotlivých bodů
    - *framerate* = počet obrazových snímků za sekundu pro zachování iluze pohybu (typicky 25 fps)
  - zpracování digitálních dat – úpravy jasu, vyvážení bílé, atp.
  - komprese – snížení datového objemu
    - u video dat nezbytná (velké objemy oproti audio datům)
    - pro účely minimalizace end-to-end latence však může být výhodnější využít nekomprimované video (např. uncompressed HD)
    - MJPEG, MPEG, DV, HD, ...

# Přenosy multimediálních dat – transportní protokoly

## TCP

- zajištění bezchybnosti přenosu je na úkor zvýšení end-to-end latence
- zajištění férovosti nedovoluje dostatečnou šířku pásma na vytížených linkách

## UDP

- nemá režii spojenou s ověřováním (a zajišťováním) bezchybnosti přenosu
- minimalistický, efektivnější, rychlejší
  - minimálně navyšuje latenci přenosu
- ⇒ vhodný pro přenos multimediálních dat
  - výhodný zejména pro interaktivní přenosy
  - využíván v drtivé většině případů (až na speciální výjimky)

# Přenosy multimediálních dat – přenosová síť

- Internet poskytuje best-effort službu
- vzhledem k využití UDP protokolu je zapotřebí vyrovnat se s chybovostí přenosu
  - avšak i s dalšími „neduhy“ – latence, jitter

# Přenosy multimediálních dat – přenosová síť

## Chybovost přenosu

### Chybovost přenosu:

- malá chybovost není problém
  - lidské ucho/oko jsou nedokonalé
    - ucho je však citlivější, proto je chybovost kritičtější pro přenos zvuku
  - „malá“ = v závislosti na kódování 1 – 20%
- lze je eliminovat s využitím techniky *dopředné korekce chyb* (*Forward Error Correction, FEC*)
  - založeno na vkládání redundantních informací do přenášených paketů
  - základní techniky:
    - XORování skupin přenášených paketů
    - posílání druhého proudu (v nižší kvalitě)
    - prokládání (*Interleaving*)

# Přenosy multimediálních dat – přenosová síť

## Chybovost přenosu – XORování skupin přenášených paketů

Oprava chyb s využitím *XORování skupin přenášených paketů*:

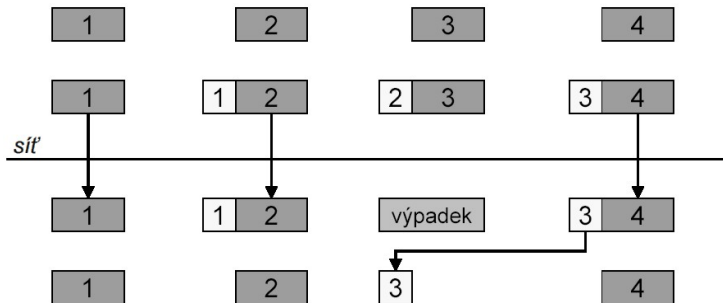
- přenášené multimediální pakety rozděleny do skupin po  $n$  paketech (pevné délky)
- po každém odeslání  $n$  paketů následuje XORovací paket
  - obsahuje XOR relevantních bitů všech odeslaných paketů dané skupiny
- vlastnosti:
  - ztrátu 1 paketu dokáže příjemce zcela opravit
  - ztrátu více paketů opravit nelze
- problém velikosti  $n$ :
  - malá hodnota  $n \Rightarrow$ 
    - velká schopnost opravy na straně příjemce
    - podstatné navýšení objemu přenášených dat
  - velká hodnota  $n \Rightarrow$ 
    - malá schopnost opravy na straně příjemce
    - malé navýšení objemu přenášených dat
    - nutnost větších bufferů  $\Rightarrow$  zvýšení end-to-end latence

# Přenosy multimediálních dat – přenosová síť

## Chybovost přenosu – Posílání druhého proudu (v nižší kvalitě)

Oprava chyb s využitím *posílání druhého proudu (v nižší kvalitě)*:

- do přenášených paketů jsou vkládána bezprostředně dříve odeslaná data
  - eventuelně v nižší kvalitě
- v případě ztráty paketu příjemce využije data přenášená v následujícím paketu



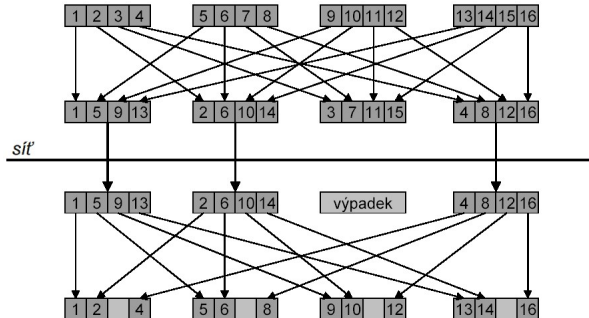


# Přenosy multimediálních dat – přenosová síť

## Chybovost přenosu – Prokládání (*Interleaving*)

Oprava chyb s využitím *prokládání* (*Interleaving*):

- změna původní sekvence dat za účelem minimalizace důsledku výpadku paketu
- vlastnosti:
  - + velmi malý overhead
  - – navýšení end-to-end zpoždění



# Přenosy multimediálních dat – přenosová síť

## Latence přenosu

### Latence přenosu:

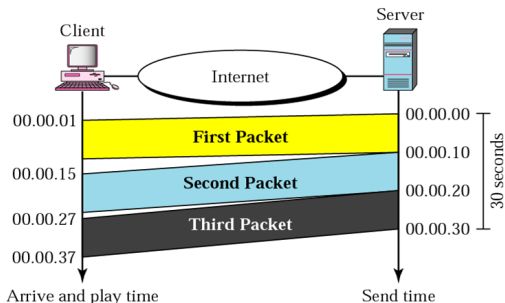
- akumulace času nutného pro:
  - vlastní přenos po síťových linkách
  - + zpracování a bufferování na síťových prvcích
  - + čas nutný pro zpracování dat na koncových systémech
- kritická pro interaktivní přenosy
  - např. pro interaktivní audio přenosy max. 150 ms (ideálně 100 ms)
  - např. pro haptické přenosy max. 1 ms
  - později příchozí pakety mohou být zahozeny

# Přenosy multimediálních dat – přenosová síť

## Jitter

### Jitter:

- = fluktuace zpoždění příchodu paketů



- techniky pro eliminaci důsledků jitteru:
  - pevně opožděné přehrávání
  - adaptivní opožděné přehrávání

# Přenosy multimediálních dat – přenosová síť

## Jitter – Opožděné přehrávání I.

Eliminace jitteru s využitím *pevně opožděného přehrávání*  
(*Fixed playout delay*):

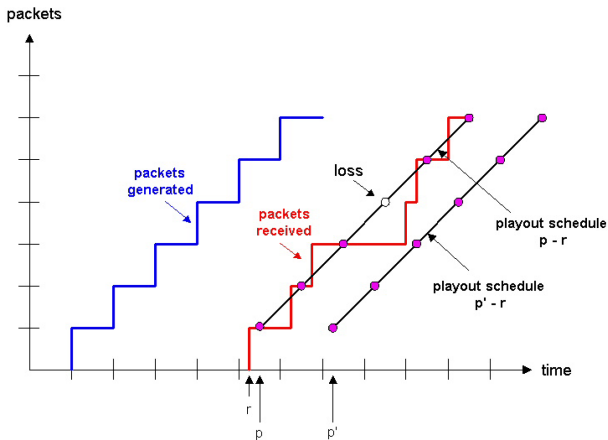
- přijatá data přehrávána s definovaným opožděním
  - v mezičase bufferována
  - předpokládá se, že data dorazí před plánovaným přehráváním
- problém určení opoždění přehrávání:
  - malé opoždění  $\Rightarrow$  více zpožděné pakety nemusí stihnout dorazit
  - velké opoždění  $\Rightarrow$  navýšení end-to-end latence přenosu

Eliminace jitteru s využitím *adaptivního opožděného přehrávání*  
(*Adaptive playout delay*):

- snaha o průběžné přizpůsobování konstanty určující opoždění přehrávání

# Přenosy multimediálních dat – přenosová síť

## Jitter – Opožděné přehrávání II.



Obrázek: Ilustrace důsledků nedostatečné ( $p$ ) a dostatečné ( $p'$ ) konstanty pro pevné opožděné přehrávání.

# Přenosy multimediálních dat – přenosová síť

## Videokonference vs. Streaming

### Videokonference:

- při přenosu nelze používat buffery ani na straně odesílajícího ani na straně příjemce – vyžadujeme interaktivitu a tedy nízké latence
  - ⇒ potřeba využívat kodeky s nízkou latencí
- latence a jitter jsou při přenosu sítí také velmi problematické

### Streaming

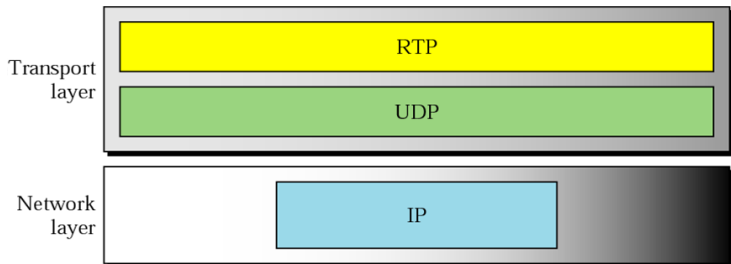
- díky jednosměrnosti provozu můžeme data bufferovat
- latence při přenosu vznikající při kompresi videa není problém
- latence vznikající přenosem v síti a její rozptyl také není podstatná – lze řešit bufferem

# Přenosy multimediálních dat – přenosové protokoly

- multimediální přenosy převážně realizovány nad protokolem UDP
  - UDP však nemá žádnou podporu pro multimediální aplikace
    - časové značky, informace o pořadí paketu (sekvenční čísla), atp.
  - ⇒ návrh protokolu **RTP (Real-time Transport Protocol)**

## RTP (Real-time Transport Protocol):

- transportní protokol postavený nad protokolem UDP
  - obohacuje UDP o vlastnosti vhodné pro přenos multimediálních dat



# Přenosy multimediálních dat – přenosové protokoly

## RTP (Real-time Transport Protocol)

### **RTP (Real-time Transport Protocol):**

- klíčové vlastnosti:
  - identifikace obsahu
  - sekvenční číslování paketů
  - časové značky pro jednotlivé pakety
- protokol sám od sebe nezaručuje kvalitu přenosu, pouze aplikacím poskytuje prostředky pro možné zaručení kvality

### **RTCP (RTP Control Protocol):**

- doplňuje protokol RTP
- poskytuje out-of-band informace pro řízení proudu dat přenášeného pomocí RTP
  - RTCP poskytuje aplikaci zpětnou vazbu na kvalitu přenosu pomocí protokolu RTP



# Přenosy multimediálních dat – přenosové protokoly

## SIP (Session Initiation Protocol) a H.323

### SIP (Session Initiation Protocol):

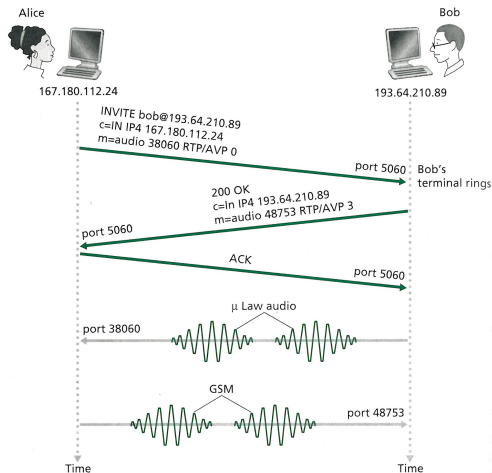
- protokol aplikační vrstvy určený pro ustavení, správu a ukončení multimediálních relací (hovorů)
  - přenášené zprávy jsou textové
  - vlastní přenos multimediálních dat realizován např. prostřednictvím RTP/RTCP protokolu
- využíván např. ve *Voice over IP (VoIP)* telefonii

### H.323:

- organizací ITU definovaná alternativa k protokolu SIP
- vlastní přenos dat realizován prostřednictvím RTP/RTCP protokolu
- pro ustavení a správu hovoru využívá mnoho dalších protokolů
  - G.71, G.723.1, H.245, Q.931, H.225, atp.

# Přenosy multimediálních dat – přenosové protokoly

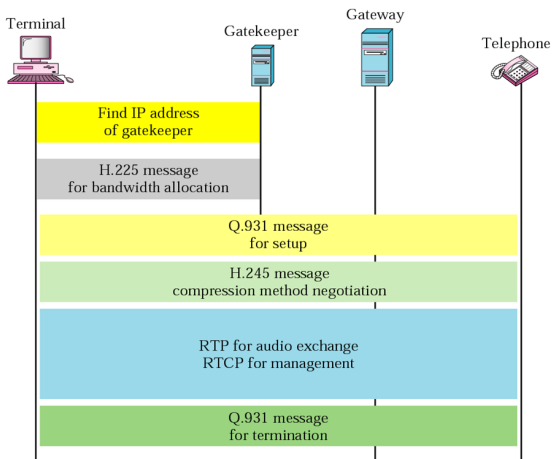
## SIP (Session Initiation Protocol)



Obrázek: Příklad ustavení VoIP hovoru s využitím protokolu SIP.

# Přenosy multimediálních dat – přenosové protokoly

## H.323



Obrázek: Příklad ustavení VoIP hovoru s využitím protokolu H.323.

# Přenosy multimediálních dat – přenosové protokoly

## RTSP (Real-Time Streaming Protocol) a MMS (Microsoft Media Services)

Protokoly pro streaming (uložených) multimediálních dat:

- **RTSP (Real-Time Streaming Protocol):**

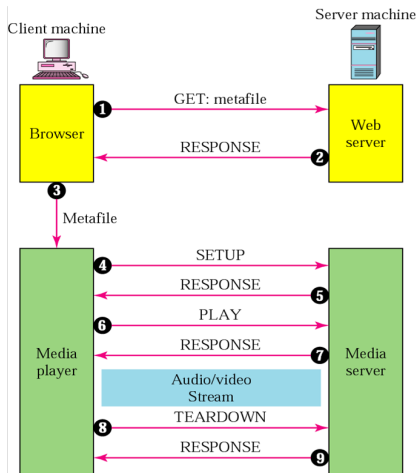
- stavový protokol založený na HTTP požadavcích (GET apod.)
- ovládání streaming serveru (VCR příkazy jako Play, Pause a Stop) a přístup k souborům podle času
- pro přenos dat se používá protokol RTP + RTCP

- **MMS (Microsoft Media Services):**

- nebo také Netshow services
- proprietární protokol
- pro přenos dat se používají protokoly UDP nebo i TCP pokud se nezdaří vyjednat spojení na protokolu UDP
- jako poslední z možností je „streaming“ pomocí upraveného protokolu HTTP (tedy opět nad protokolem TCP)

# Přenosy multimediálních dat – přenosové protokoly

## RTSP (Real-Time Streaming Protocol)



Obrázek: Příklad streamování uloženého videa s využitím protokolu RTSP.

# Struktura přednášky

- 1 Přehled
- 2 Úvod
- 3 Základní členění aplikací
  - Komunikační modely – Client-Server vs. Peer-to-peer
  - Přístup k informacím – Pull model vs. Push model
  - Nároky na počítačovou síť – nízké vs. vysoké
- 4 Pohled programátora
- 5 Vybrané síťové aplikace
  - Jmenná služba – DNS
  - World Wide Web – HTTP
  - Elektronická pošta – SMTP
  - Přenos souborů – FTP
  - Multimediální přenosy v datových sítích
- 6 Rekapitulace

# Rekapitulace – Aplikační vrstva

- poskytuje služby pro uživatele
  - rozhraní mezi uživatelem a počítačovou sítí
- aplikace lze členit dle nejrůznějších hledisek
  - klient/server vs. peer-to-peer, pull vs. push model, nároky na počítačovou síť, atp.
- příklady stěžejních aplikací a aplikačních protokolů Internetu:
  - jmenná služba (DNS)
  - World-Wide-Web (HTTP)
  - elektronická pošta (SMTP)
  - přenos souborů (FTP)
  - multimediální přenosy (RTP/RTCP)
- *další informace:*
  - PA159: Počítačové sítě a jejich aplikace I. (doc. Hladká)
  - PA160: Počítačové sítě a jejich aplikace II. (prof. Matyska)
  - PV188: Principy zpracování a přenosu multimédií (doc. Hladká, dr. Liška, Ing. Šiler)
  - atd.