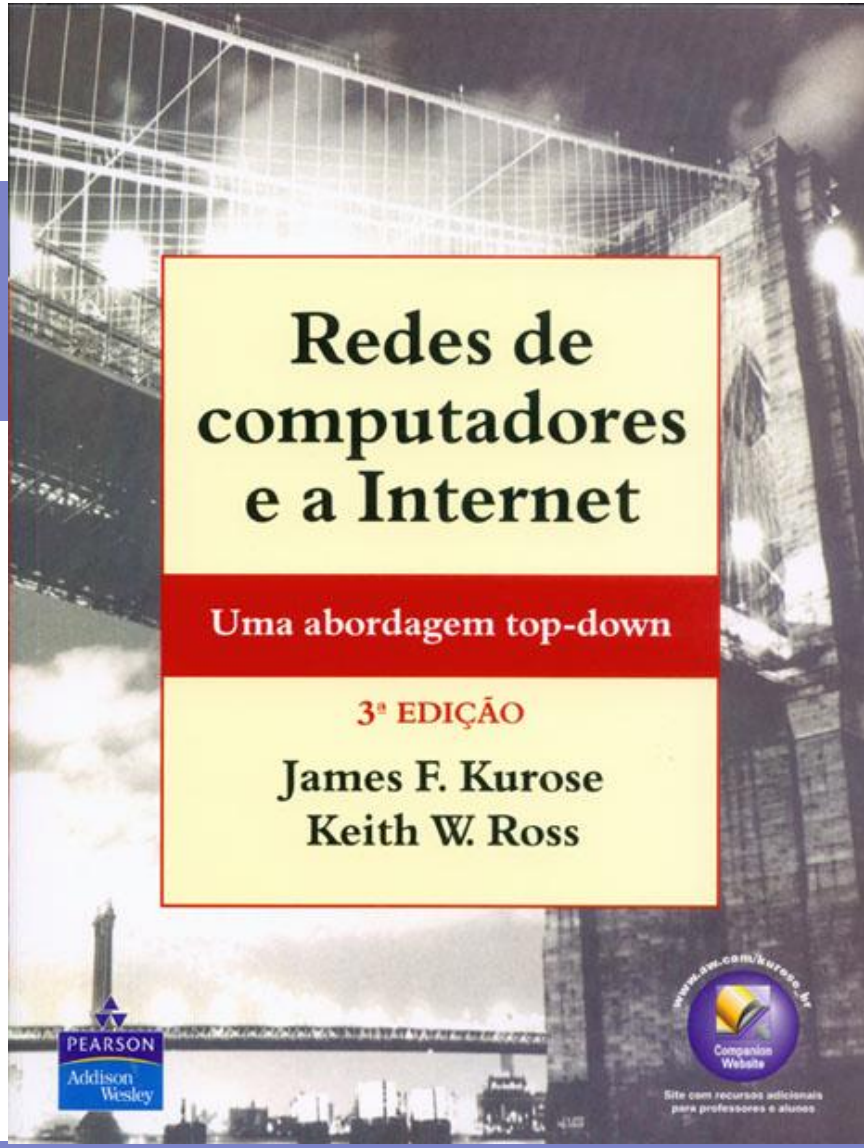
Redes de computadores e a Internet

Capítulo 2

Camada de aplicação

(Alterado por Atslands Rocha)



2 Camada de aplicação

- 2.1 Princípios de aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio electrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Compartilhamento de arquivos P2P

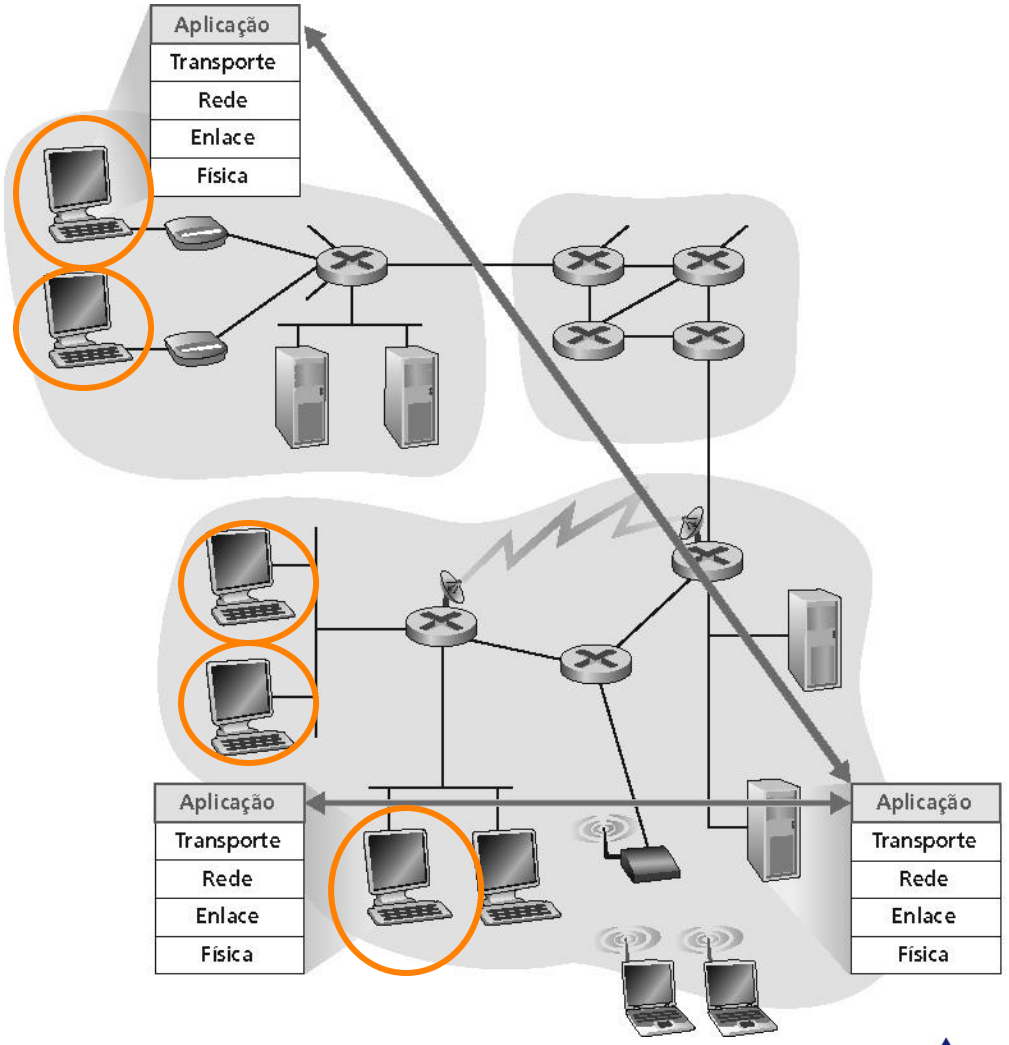
2 Aplicações de rede

Programas que

- Executem sobre diferentes sistemas finais.

Nenhum software é escrito para dispositivos no núcleo da rede

- Dispositivos do núcleo da rede não trabalham na camada de aplicação.



2 Algumas aplicações de rede

- E-mail
- Web
- Mensagem instantânea
- Login remoto
- Compartilhamento de arquivo P2P
- Jogos de rede multi-usuário
- Telefonia via Internet
- Videoconferência em tempo real
- Computação paralela massiva

2 Arquiteturas de aplicação

- Cliente-servidor
- **Peer-to-peer** (P2P)
- Híbrida de cliente-servidor e P2P

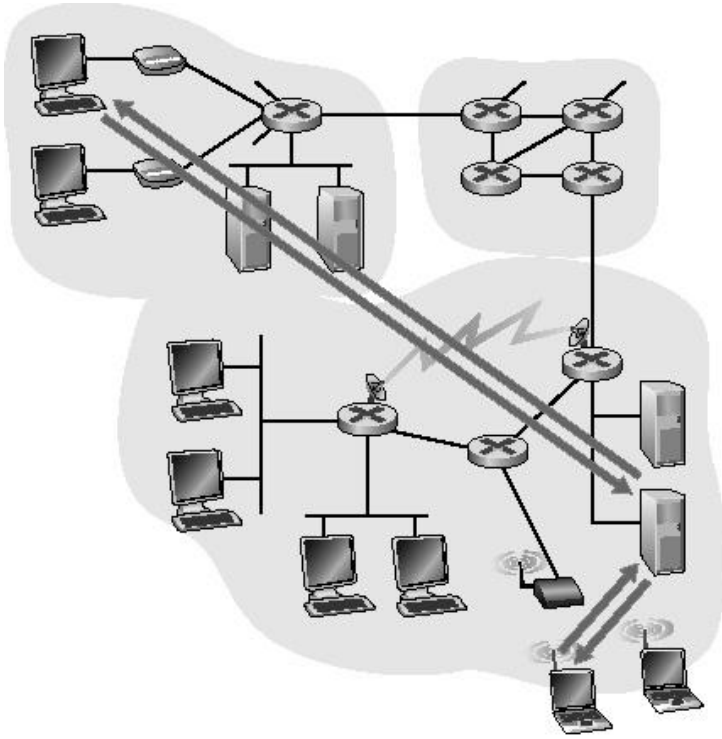
2 Arquitetura cliente-servidor

Servidor:

- Hospedeiro sempre ativo;
- Endereço IP permanente;
- Fornece serviços solicitados pelo cliente;
- Processo servidor que espera para ser contatado.

Clientes:

- Comunicam-se com o servidor;
- Pode ter endereço IP dinâmico;
- Não se comunicam diretamente uns com os outros;
- Processo cliente que inicia a comunicação.



a. Aplicação cliente-servidor

2 De quais requisitos uma aplicação necessita?

Perda de dados

- Algumas aplicações (ex.: áudio, vídeo) podem tolerar alguma perda;
- Outras aplicações (ex.: transferência de arquivos, correio eletrônico) exigem transferência de dados 100% confiável.

Temporização

- Algumas aplicações (ex.: telefonia Internet, jogos interativos,) exigem baixos atrasos para serem “efetivos”.

Banda passante

- Algumas aplicações (ex.: multimídia) exigem uma banda mínima para serem “efetivas”.
- Outras aplicações (“aplicações elásticas”) melhoram quando a banda disponível aumenta.

2 Requisitos de transporte de aplicação comuns

Aplicação	Perdas	Banda	Sensível ao atraso
Transferência arquivos	sem perdas	elástica	não
E-mail	sem perdas	elástica	não
Documentos Web	sem perdas	elástica	não
Áudio/vídeo tempo real	tolerante	aúdio: 5 Kb-1 Mb vídeo:10 Kb-5 Mb	sim, décimos de seg
Áudio/video armazenado	tolerante	igual à anterior	sim, alguns segundos
Jogos interativos	tolerante	kbps	sim, décimos de seg
Mensagem instantânea	sem perda	elástica	sim e não

Serviços dos protocolos de transporte da Internet

Serviço TCP:

- **Orientado à conexão:** conexão requerida entre processos cliente e servidor;
- **Transporte confiável** entre os processor de envio e recepção;
- **Controle de fluxo:** o transmissor não sobrecarrega o receptor;
- **Controle de congestionamento:** protege a rede do excesso de tráfego;

  Não oferece: garantias de temporização e de banda mínima.

Serviço UDP:

- Transferência de dados não confiável entre transmissor e receptor;
- Não oferece: estabelecimento de conexão, controle de fluxo e de congestionamento, garantia de temporização e de banda mínima.

P.: Por que ambos? Por que existe o UDP?

Aplicação e protocolos de transporte da Internet

Aplicação	Protocolo de aplicação	Protocolo de transporte
E-mail	smtp [RFC 821]	TCP
Acesso de terminais remotos	telnet [RFC 854]	TCP
Web	http [RFC 2068]	TCP
Transferência de arquivos	ftp [RFC 959]	TCP
Multimídia (streaming)	RTP ou proprietário (ex.: RealNetworks)	TCP ou UDP
Servidor de arquivos remoto	NFS	TCP ou UDP
Telefonia Internet	RTP ou proprietário (ex.: Vocaltec)	tipicamente UDP

Web e HTTP

Primeiro alguns jargões

- Página Web consiste de objetos (arquivo HTML, imagem JPEG, Java applet, arquivo de áudio,…);
- A página Web consiste de arquivo-HTML base que inclui vários objetos referenciados;
- Cada objeto é endereçado por uma URL;

- Exemplo de URL:

`www.someschool.edu/someDept/pic.gif`

Nome do hospedeiro | Nome do caminho

Visão geral do HTTP

HTTP: hypertext transfer protocol

- Protocolo da camada de aplicação da Web;
- Modelo cliente/servidor
  - Cliente: browser solicita, recebe e apresenta objetos da Web;
  - Servidor: envia objetos em resposta a pedidos.

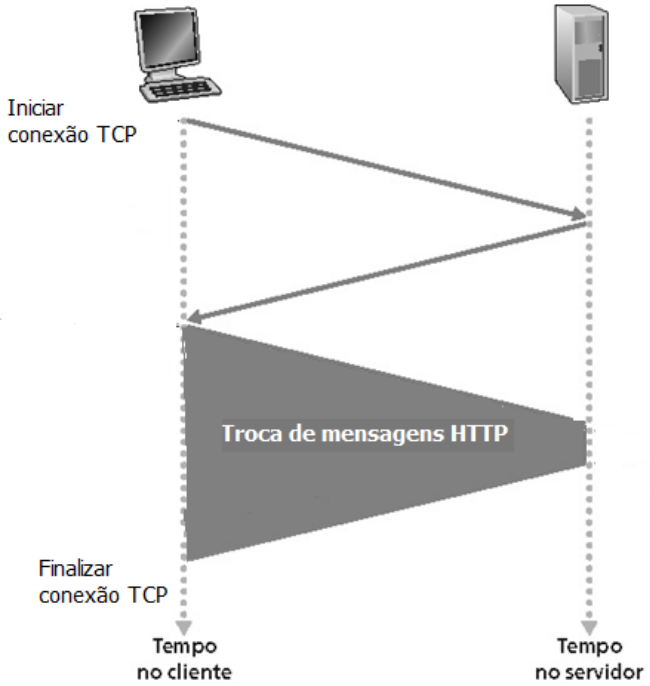
HTTP é um protocolo "sem estado"

- O servidor não mantém informação sobre os pedidos passados pelos clientes.

Visão geral do HTTP

Utiliza TCP:

- Cliente inicia conexão TCP para o servidor na porta 80;
- Servidor aceita uma conexão TCP do cliente;
- mensagens HTTP são trocadas entre o browser (cliente HTTP) e o servidor Web (servidor HTTP);
- A conexão TCP é fechada.



2 Conexões HTTP

HTTP não persistente

- No máximo, um objeto é enviado sobre uma conexão TCP;
- O HTTP/1.0 utiliza HTTP não persistente.

HTTP persistente

- Múltiplos objetos podem ser enviados sobre uma conexão;
- TCP entre o cliente e o servidor;
- O HTTP/1.1 utiliza conexões persistentes em seu modo padrão .

2 HTTP não persistente

Usuário entra com a URL: (contém texto,referências a 10 imagens jpeg)

`www.someSchool.edu/someDepartment/home.index`

1a. Cliente HTTP **inicia conexão TCP** ao servidor em www.exemplo.edu.

1b. Servidor HTTP **"aceita" conexão**, notificando o cliente.

2. Cliente HTTP envia HTTP **request message** (contendo a URL).

3. Servidor HTTP recebe msg de pedido, forma **response message** contendo o objeto solicitado (exemplo/home.index), envia msg para o cliente. 4. Servidor HTTP fecha conexão TCP.

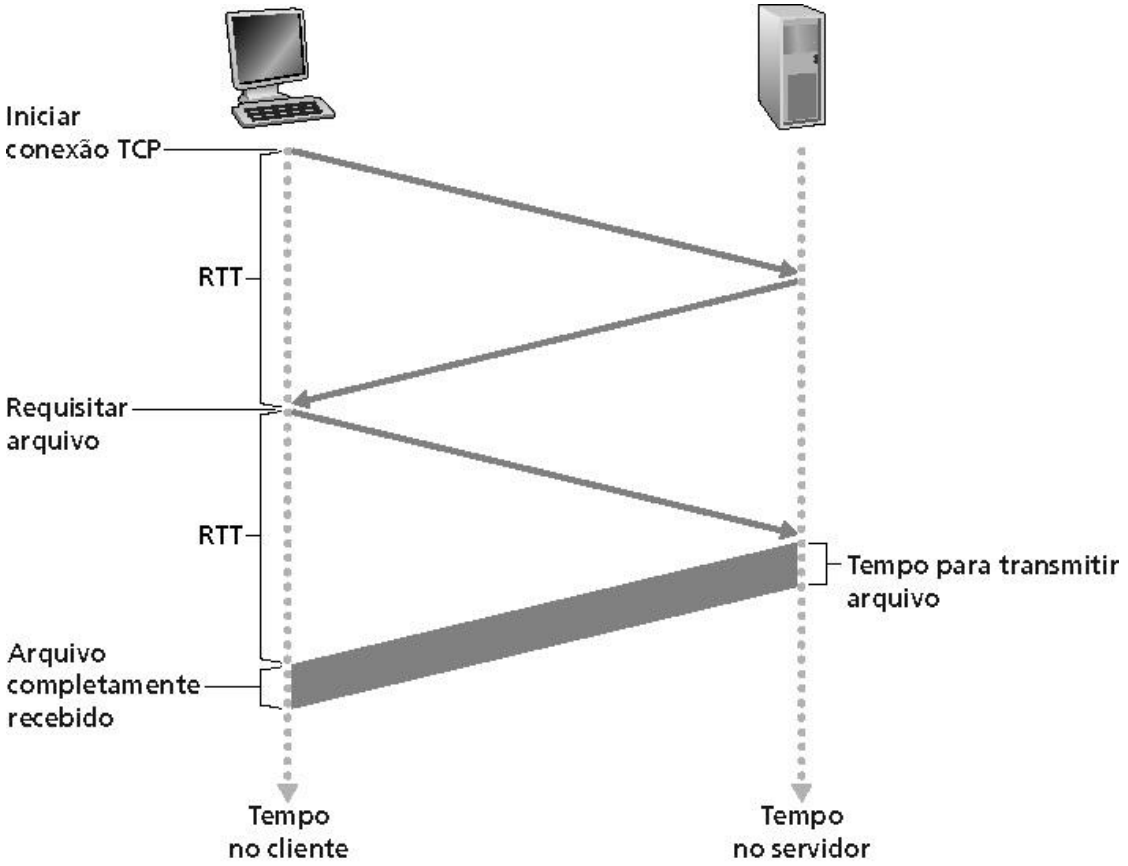
5. Cliente HTTP recebe resposta com o arquivo html. Analisando o arquivo html, encontra 10 objetos jpeg referenciados.

Tempo

2 HTTP não persistente

6. Passos 1-5 (ilustrados abaixo) são repetidos para cada um dos 10 objetos jpeg.

Tempo



HTTP persistente

- Servidor deixa a conexão aberta após enviar uma resposta;
- Mensagens HTTP subseqüentes entre o mesmo cliente/servidor são enviadas pela conexão.

Persistente sem **paralelismo:**

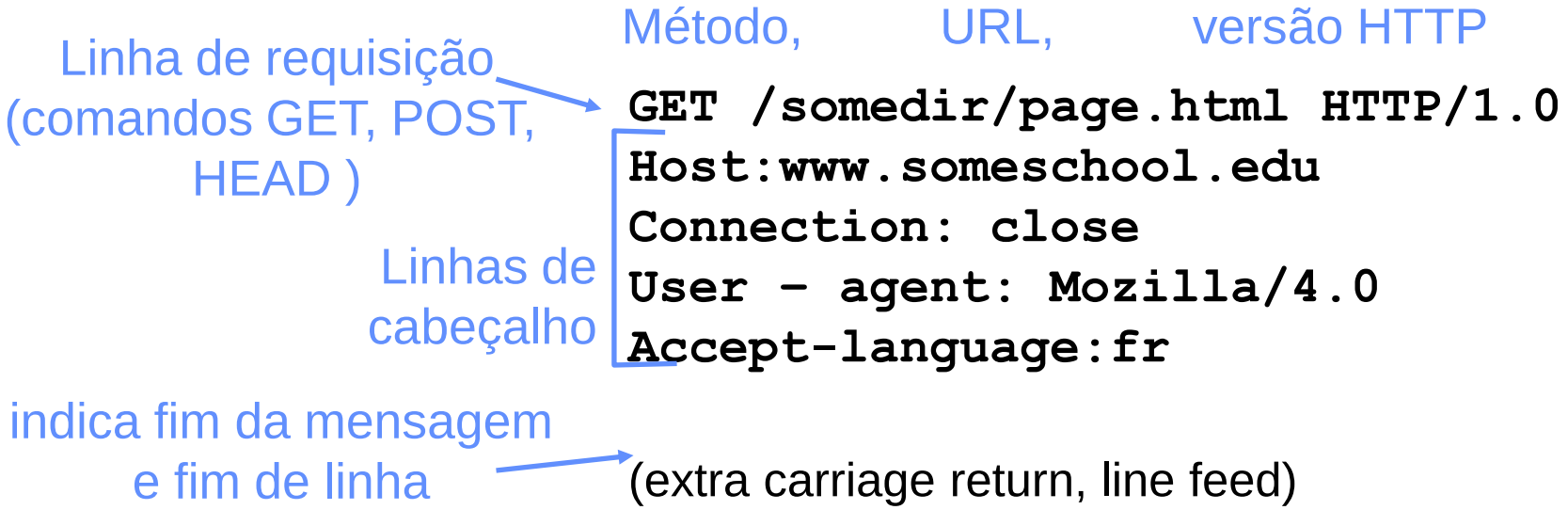
- O cliente emite novas requisições apenas quando a resposta anterior for recebida;
- Conexão ociosa.

Persistente com **paralelismo** (modo default):

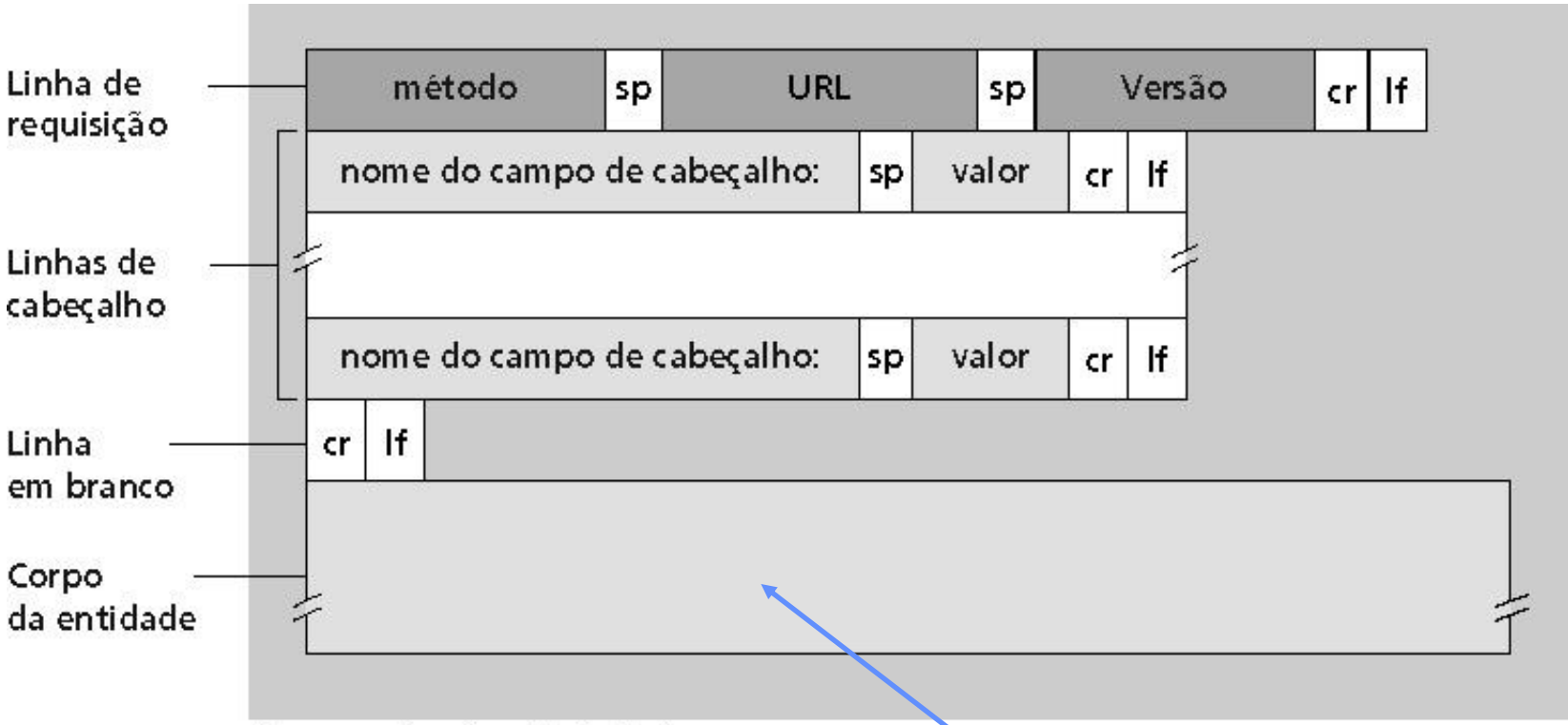
- Padrão no HTTP/1.1;
- O cliente envia requisições assim que encontra um objeto referenciado.

2 Mensagem HTTP request

- Dois tipos de mensagens HTTP: request, response.
- HTTP request message:
  - ASCII (formato legível para humanos)



2 Mensagem HTTP request: formato geral



Obs.: cr = carriage return; lf = line feed

2 Entrada de formulário

Método Post: (+)

- Página Web frequentemente inclui entrada de formulário;
- A entrada é enviada para o servidor no corpo da entidade.

Método URL: (+)

- Utiliza o método GET
- A entrada é enviada no campo de URL da linha de requisição:

www.somesite.com/animalsearch?monkeys&banana

Método HEAD:

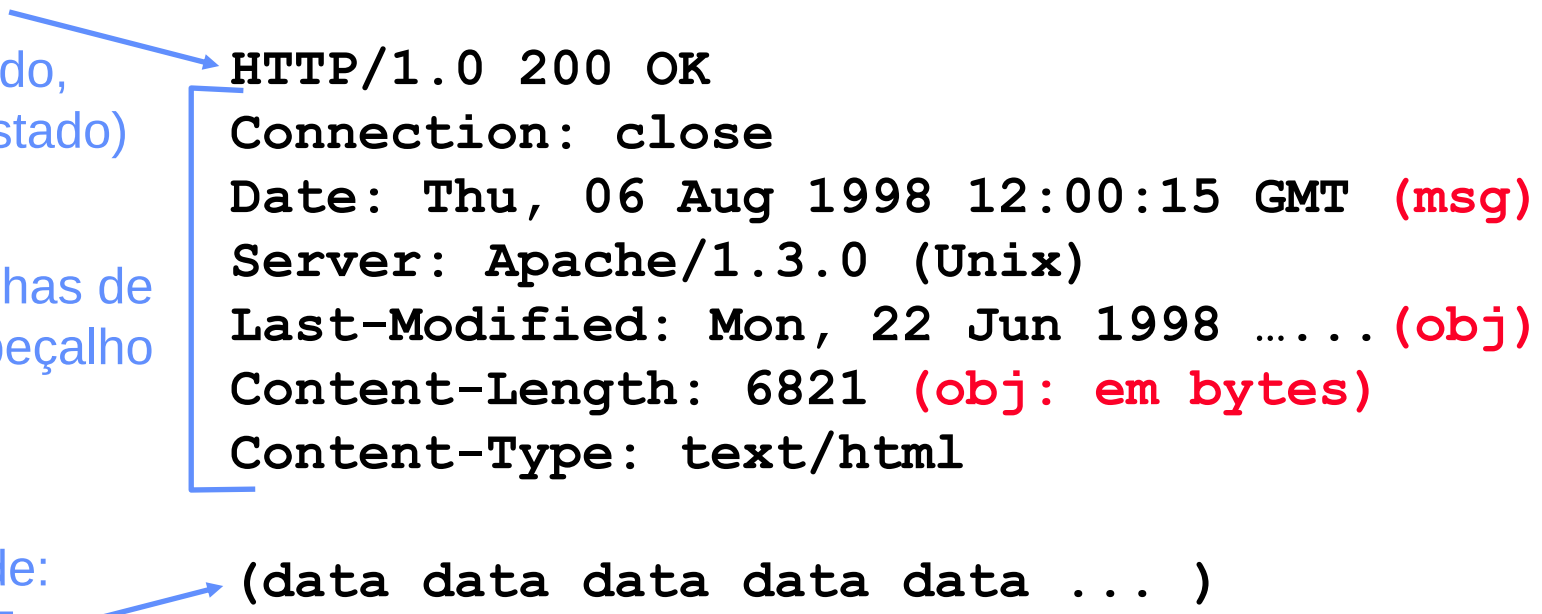
- Semelhante ao método GET;
- Responde com uma mensagem HTTP sem o objeto requisitado;
- Uso em depuração por desenvolvedores.

2 Mensagem HTTP response

Linha de estado (protocolo, código de estado, mensagem de estado)

Linhas de cabeçalho

Corpo da Entidade: Objeto solicitado.Ex.: arquivo html



2 Códigos de *status* das respostas

Na primeira linha da mensagem de resposta servidor → cliente.
Alguns exemplos de códigos:

200 OK
- Requisição bem-sucedida, objeto requisitado a seguir nesta mensagem

301 Moved permanently
- Objeto requisitado foi movido, nova localização especificada a seguir nesta mensagem (Location:)

400 Bad request
- Mensagem de requisição não compreendida pelo servidor

404 Not Found
- Documento requisitado não encontrado neste servidor

505 HTTP version not supported
- Versão HTTP não suportada pelo servidor

2 Cookies

O que os cookies podem trazer:

- Autorização
- Cartões de compra
- Recomendações

Cookies e privacidade:

- Cookies permitem que sites saibam muito sobre você;
- Você pode fornecer nome e e-mail para os sites;
- Mecanismos de busca usam redirecionamento e cookies para saberem mais sobre você;
- Companhias de propaganda obtêm informações por meio dos sites.

P: Facilidade *versus* Segurança?

2 Estado usuário-servidor: cookies

A maioria dos grandes Web sites utilizam cookies (monitora usuários!)

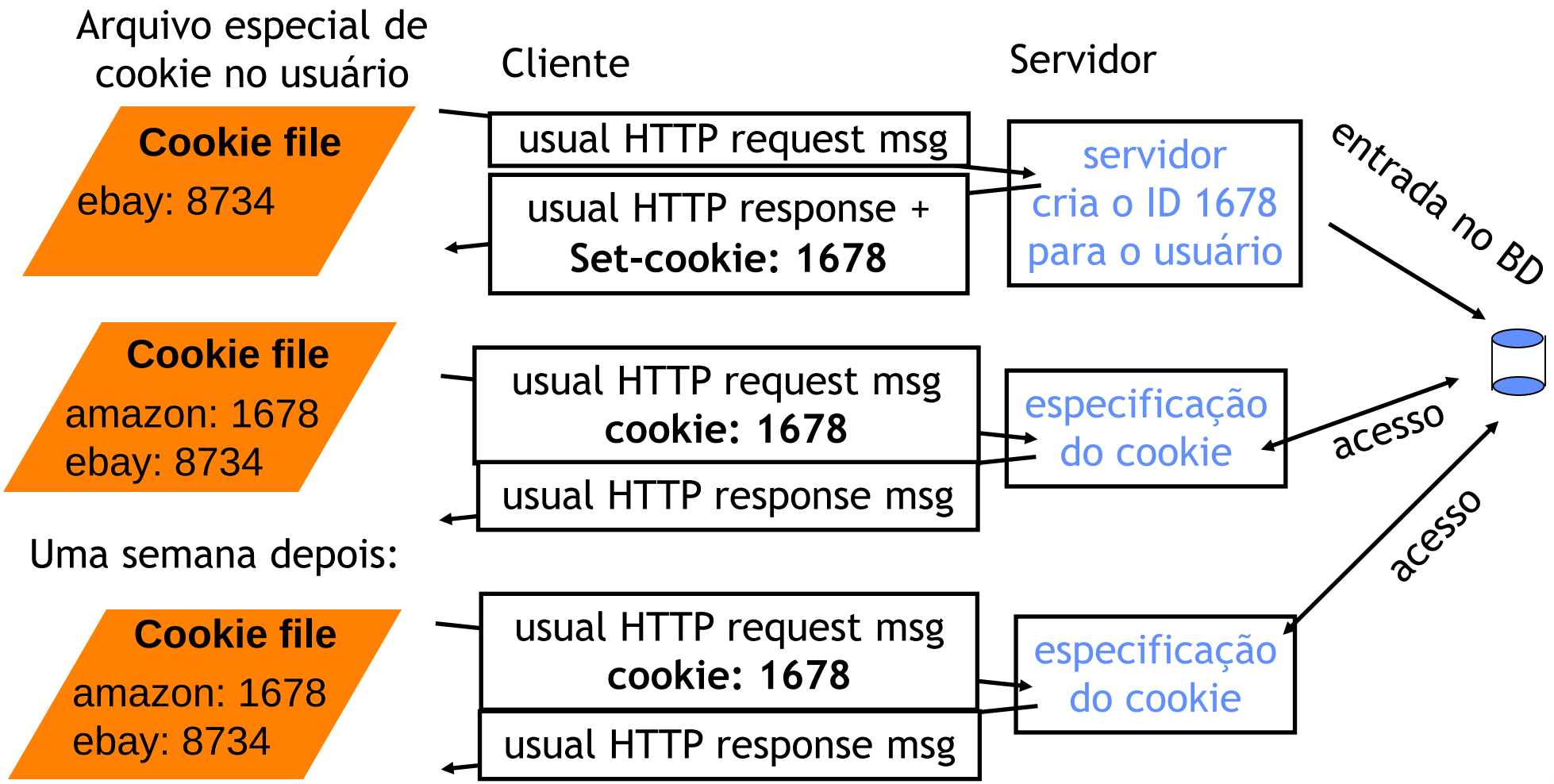
Quatro componentes:

1) Linha de cabeçalho do cookie na mensagem HTTP response
2) Linha de cabeçalho de cookie na mensagem HTTP request
3) Arquivo de cookie mantido no hospedeiro do usuário e manipulado pelo browser do usuário
4) Banco de dados de apoio no site Web

Exemplo: (+)

- Susan acessa a Internet sempre do mesmo PC
- Ela visita um site específico de e-commerce pela primeira vez
- Quando a requisição HTTP inicial chega ao site, este cria um ID único e uma entrada no banco de dados para este ID

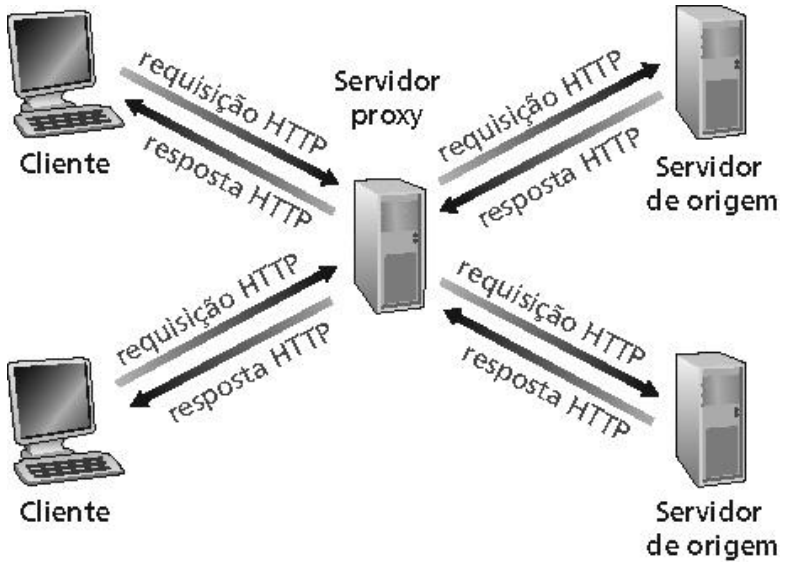
2 Cookies: mantendo “estado”



2 Web caches (Servidor proxy)

Conceito: **Entidade da rede que atende resquisições HTTP em nome de um servidor WEB de origem.**

- Usuário configura o browser: para acessar Web via proxy;
- Cliente envia todos os pedidos HTTP para o proxy;
  - Se o objeto existe no proxy, ele retorna o objeto;
  - Senão, solicita objeto do servidor original e envia o objeto ao cliente armazenando uma cópia local.

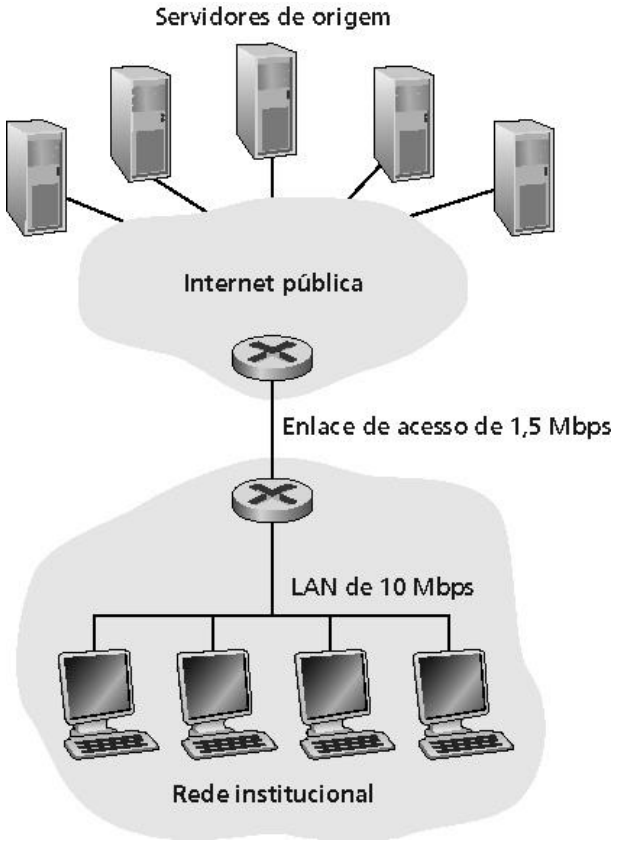


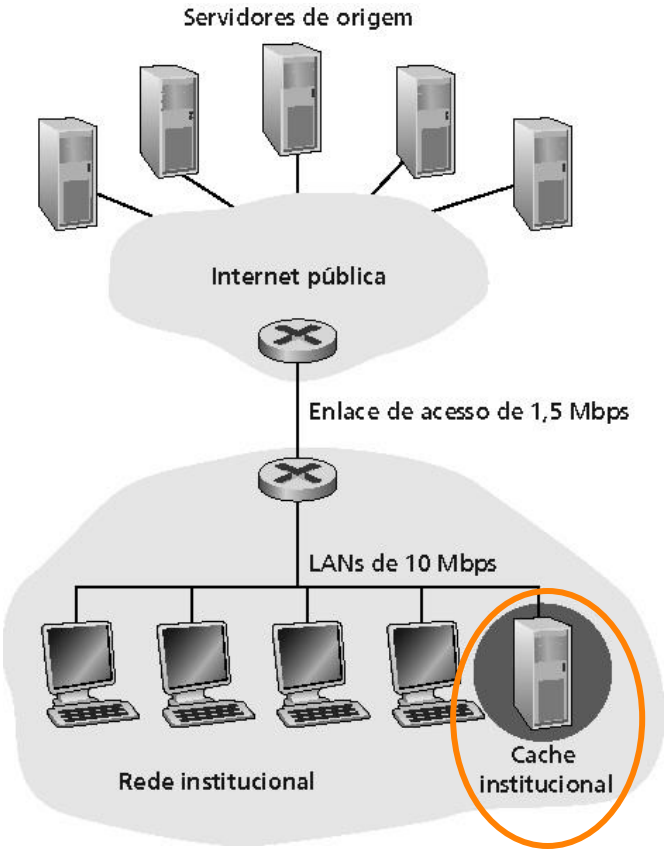
Proxy é servidor e cliente! Por quê?

2 Web caches (Servidor proxy)

Objetivos:

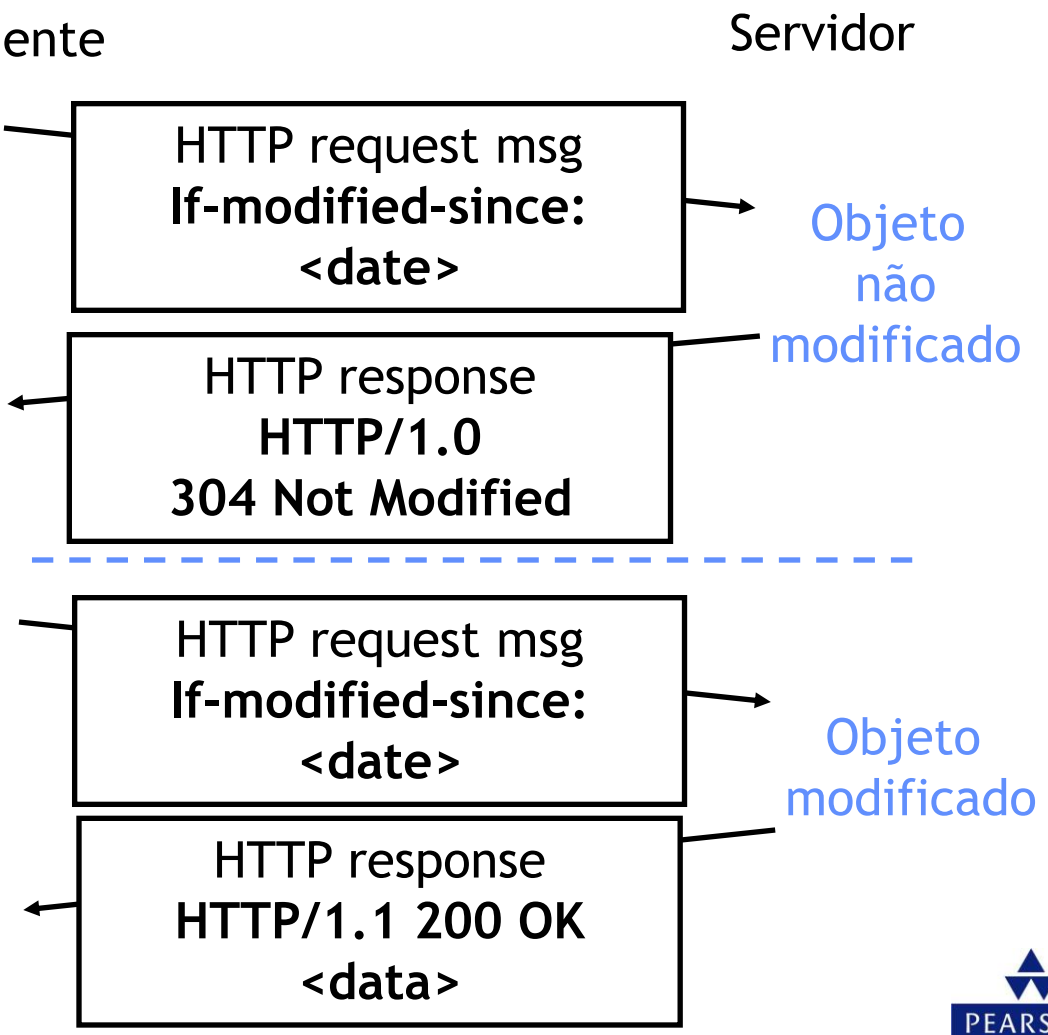
- Reduz o tempo de resposta para o cliente;
- Reduz o tráfego na instituição.



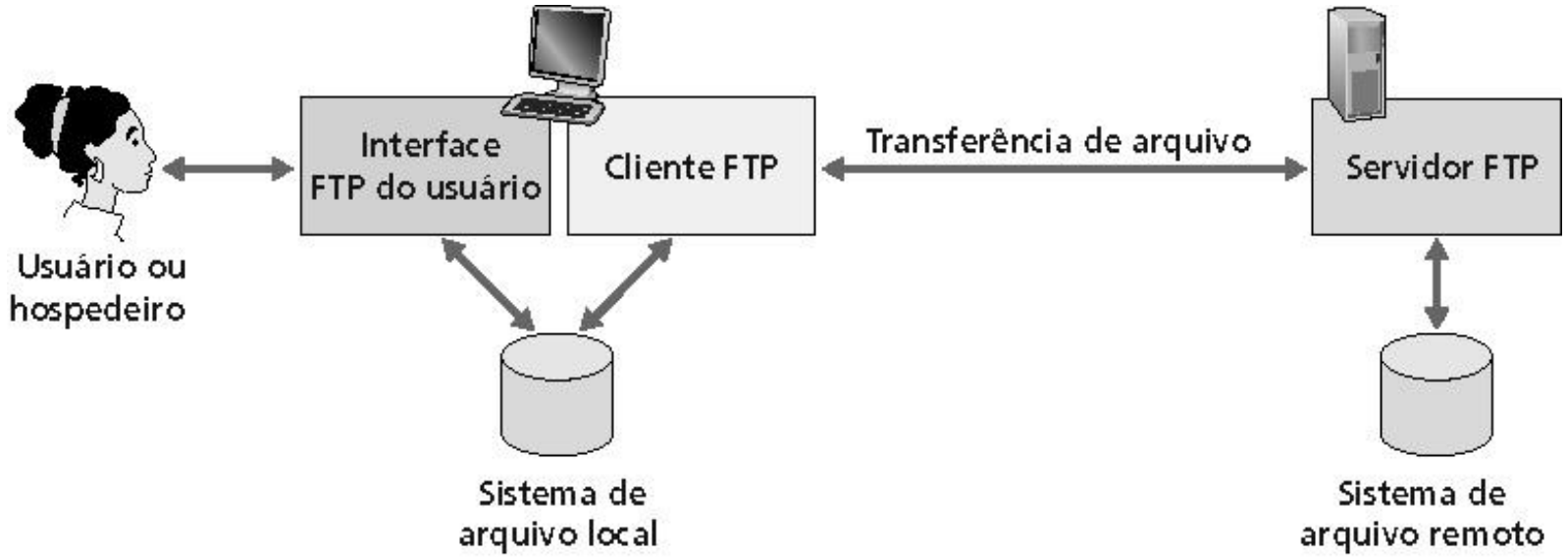


2 GET condicional

- Possíveis problemas Web Cache (proxy): objetos desatualizados!!
- Razão: não enviar objeto se a versão que o cliente já possui está atualizada.
- Cliente: especifica data da versão armazenada no pedido HTTP
  - **If-modified-since: <date>**
- <date> = <Last-Modified> HTTP response anterior
- Servidor: resposta não contém objeto se a cópia é atualizada:
  HTTP/1.0 304 Not Modified



2 FTP: o protocolo de transferência de arquivos

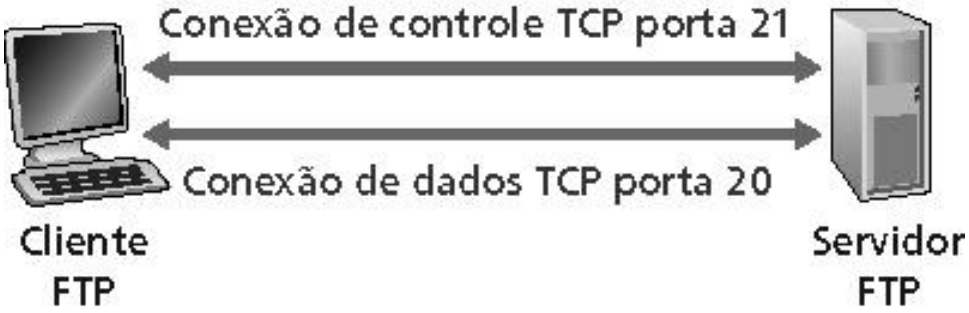


- Transferência de arquivos de/para o computador remoto;
- Modelo cliente servidor.

- Teste no Windows Explorer: ftp://ftp.pucmg.br (Usuário: anonymous Sem senha) (Diretório: pub/tmp/tcpip)
  Nota: Vários programas podem ser usados!

2 FTP: controle separado, conexões de dados

- Cliente FTP contata o servidor FTP na porta 21 (com TCP);
- Cliente obtém autorização pela conexão de controle;
- Quando o servidor recebe um comando para uma transferência de arquivo, ele abre uma conexão de dados TCP para o cliente;
- Após a transferência de um arquivo, o servidor fecha a conexão;
- Servidor abre uma segunda conexão de dados TCP para transferir outro arquivo;
- Conexão de controle: "fora da banda" (HTTP controle "na banda". Por quê?)
- Servidor FTP mantém "estado": diretório atual, autenticação anterior



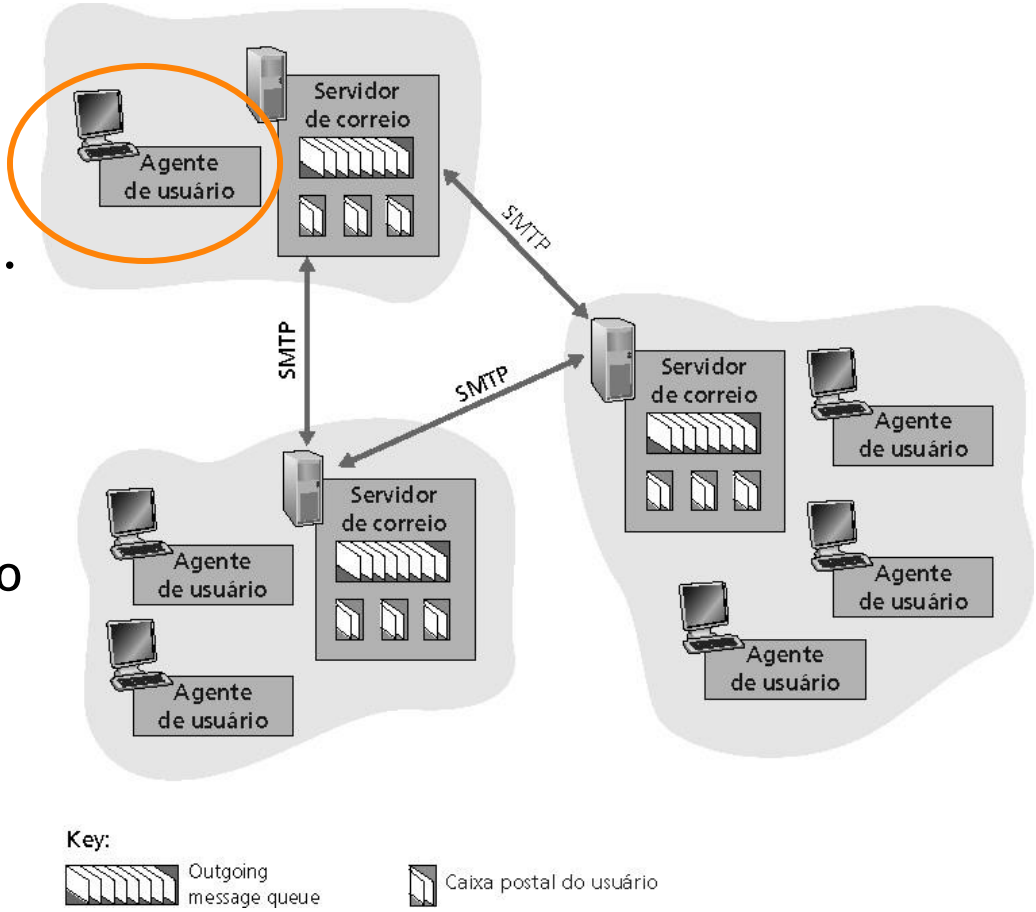
2 Correio eletrônico

Três componentes principais:

- Agentes de usuário;
- Servidores de correio;
- Simple mail transfer protocol: SMTP.

Agente de usuário

- Ex.: Eudora, Outlook, Netscape Messenger;
- Mensagens de entrada e de saída são armazenadas no servidor.



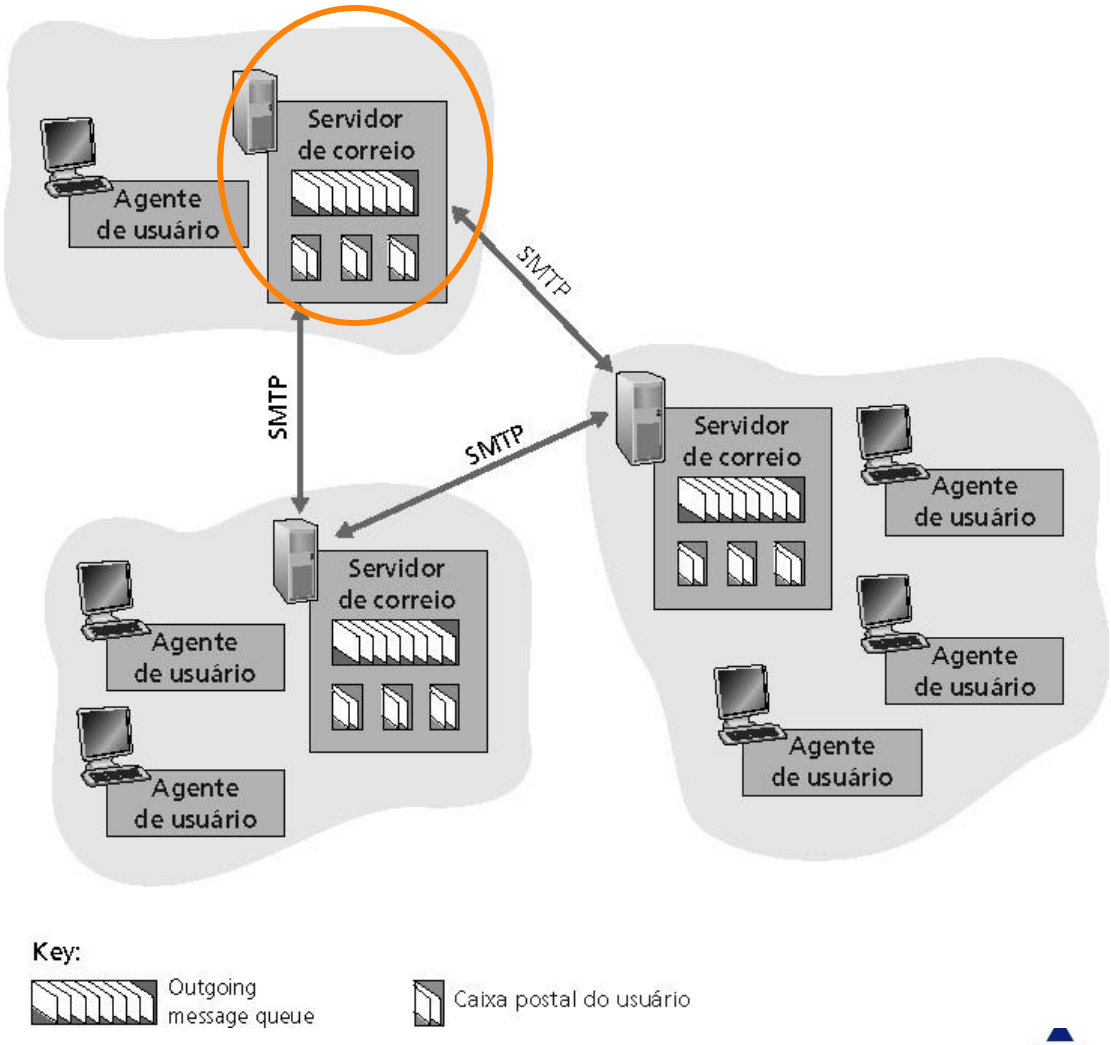
Correio eletrônico

Servidores de correio

- Caixa postal contém emails que chegaram para o usuário;
- Fila de mensagens contém os emails a serem enviados.

Protocolo SMTP permite aos servidores de correio trocarem mensagens entre si:

- Cliente: servidor de correio que envia;
- "servidor": servidor de correio que recebe;
- Problemas de entrega: Novas tentativas a cada 30min até timeout.
- Em caso timeout: msg de erro!



2 Correio eletrônico: SMTP

- Usa TCP para transferência confiável de e-mails, porta 25 (Mesma conexão TCP para todos os e-mails de Alice para Bob);
- Transferência direta: servidor que envia ("cliente") para o servidor que recebe ("servidor");

- Três fases de transferência
  - Handshaking (apresentação);
  - Transferência de mensagens;
  - Fechamento.

Legenda:

S: servidor

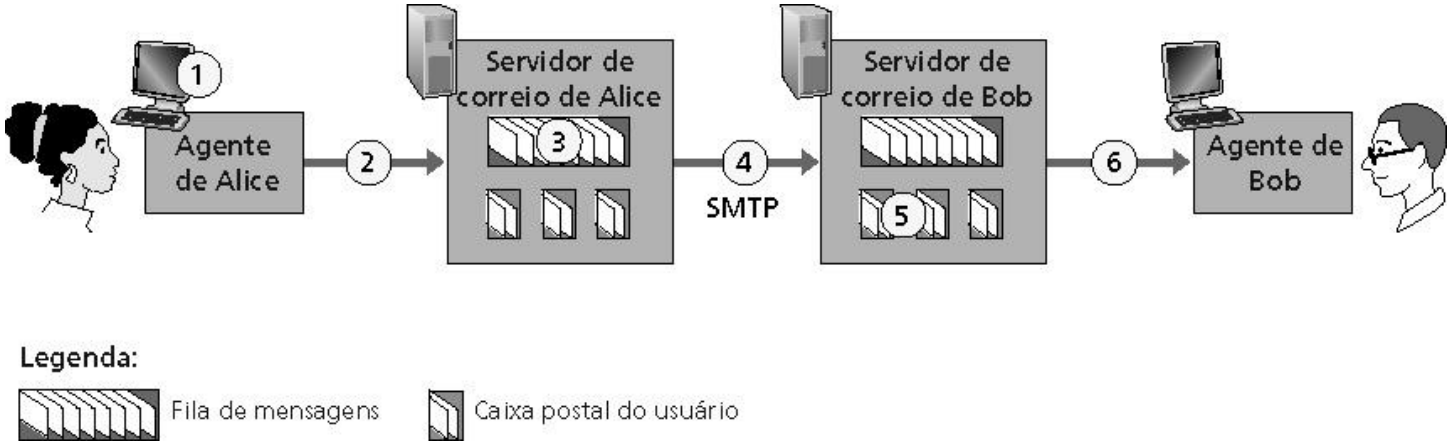
C: cliente

```
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
```

Texto do E-mail

2 Cenário: Alice envia mensagem para Bob

1) Alice usa o agente de usuário para compor a msg “para” bob@someschool.edu.
2) O agente de usuário dela envia a msg para o seu servidor de correio; a msg é colocada na fila de msgs.
3) Lado cliente SMTP abre uma conexão TCP DIRETA com o servidor de correio do Bob.
4) O cliente SMTP envia a msg de Alice pela conexão TCP.
5) O servidor de correio de Bob coloca a msg na caixa de correio de Bob.
6) Bob invoca seu agente de usuário para ler a msg.



Formato das mensagens: extensões multimídia

- MIME: Multipurpose Internet Mail Extensions, RFC 2045, 2056
- Linhas adicionais no cabeçalho declaram o tipo de conteúdo MIME

Versão da MIME

Método usado para codificar dados

Dados multimídia tipo, subtipo, declaração de parâmetro

Dados codificados

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg (ação)

base64 encoded data .....
.........................
......base64 encoded data
```

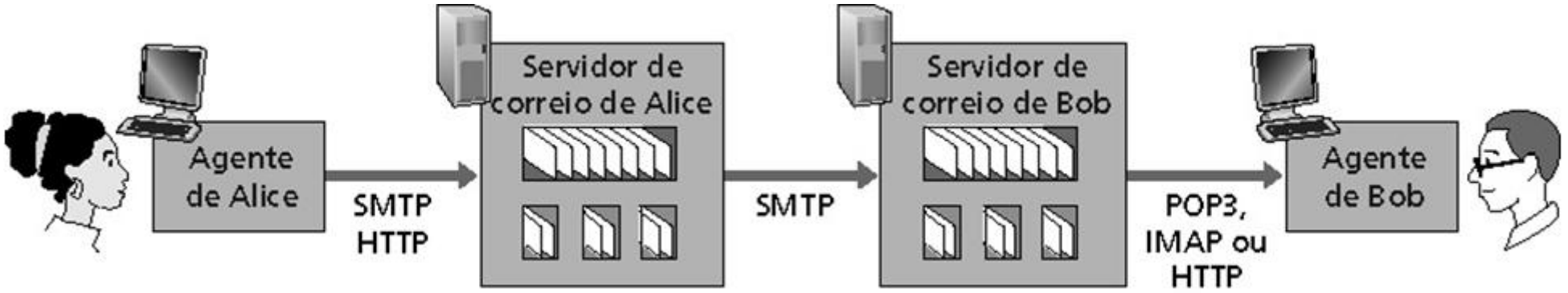
2 Formato das mensagens: Received

- Linhas inseridas pelo servidor SMTP destinatário
- Visão da mensagem pelo usuário destinatário

```
Received:from crepes.fr by hamburger.edu; 12 Oct 98 15:27:39 GMT
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data
```

2 Protocolos de acesso ao correio



- Protocolos de acesso:
  - POP3: Post Office Protocol version 3
  - IMAP: Internet Mail Access Protocol
  - HTTP: Gmail, Hotmail , Yahoo! Mail etc.
    - Remetente -> Servidor Remetente (HTTP)
    - Servidor Destino -> Destino (HTTP)
    - Servidor Remetente -> Servidor Destino (SMTP)

P: Por que não usa SMTP? SMTP é protocolo de envio e não de recuperação!

2 Protocolos de acesso ao correio

- POP3: Post Office Protocol version 3 (Porta 110)
  - Autorização: (usuário e senha);
  - Transação: recupera msgs, marca msgs para deleção, estatísticas de correio;
  - Atualização: (após comando *quit*) Deleção das msgs.
  - Protocolo “sem estado” através das sessões.
  - Não permite criar pastas remotas no servidor.

2 Protocolo POP3

Fase de autorização
- comandos do cliente:
  - **user:** declara nome do usuário
  - **pass:** password

respostas do servidor
  - **+OK**
  - **-ERR**

Fase de transação, cliente:
- **list:** lista mensagens e tamanhos
- **retr:** recupera msg pelo número
- **dele:** apaga
- **Quit**
- **Nota: Modo ler-e-apagar**
- **Nota: Após o *quit*, apaga 2msgs**

```
S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK user successfully
logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

2 POP3 (mais) e IMAP

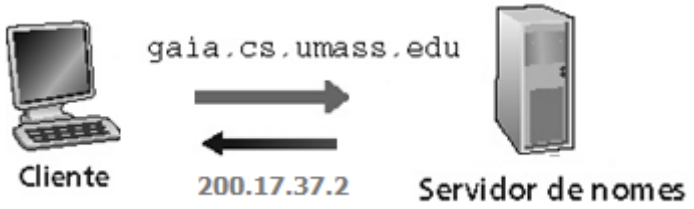
IMAP

- Mais complexo.
- Mensagens associadas à pastas (Ex: Inbox).
- Permite que o usuário organize as msgs em pastas.
- IMAP mantém o estado do usuário através das sessões:
  - Nomes das pastas e mapeamentos entre msgs e pastas.

2 DNS: Domain Name System

Internet hospedeiros, roteadores:

- Endereços IP (ex: 200.17.37.2)- usados para endereçar pacotes.
- “Nome” (ex.: gaia.cs.umass.edu) - usados por humanos.



Domain Name System (Porta 53) é:

- (1) Base de dados distribuída implementada numa hierarquia de muitos servidores de nomes (servidores DNS).
- (2) Protocolo de camada de aplicação para tradução nome/endereço.
  - Função interna da Internet, implementada como protocolo da camada de aplicação (Usado por HTTP, SMTP, FTP...)

DNS

DNS serviços

- Nome do hospedeiro para tradução de endereço IP.

- Apelidos de hospedeiros (Converte nomes “complicados” para apelidos ou vice-versa)
  Ex: Nome: relay1.west-coast.enterprise.com (canônico)
  Apelidos: enterprise.com, www.enterprise.com

- Apelidos para servidores de email (idem anterior).

- Distribuição de carga
  - Servidores Web replicados: DNS retorna a lista de endereços IP para um nome canônico de forma aleatória.
  - Cliente envia mensagem para o primeiro IP da lista (balanceando a carga).

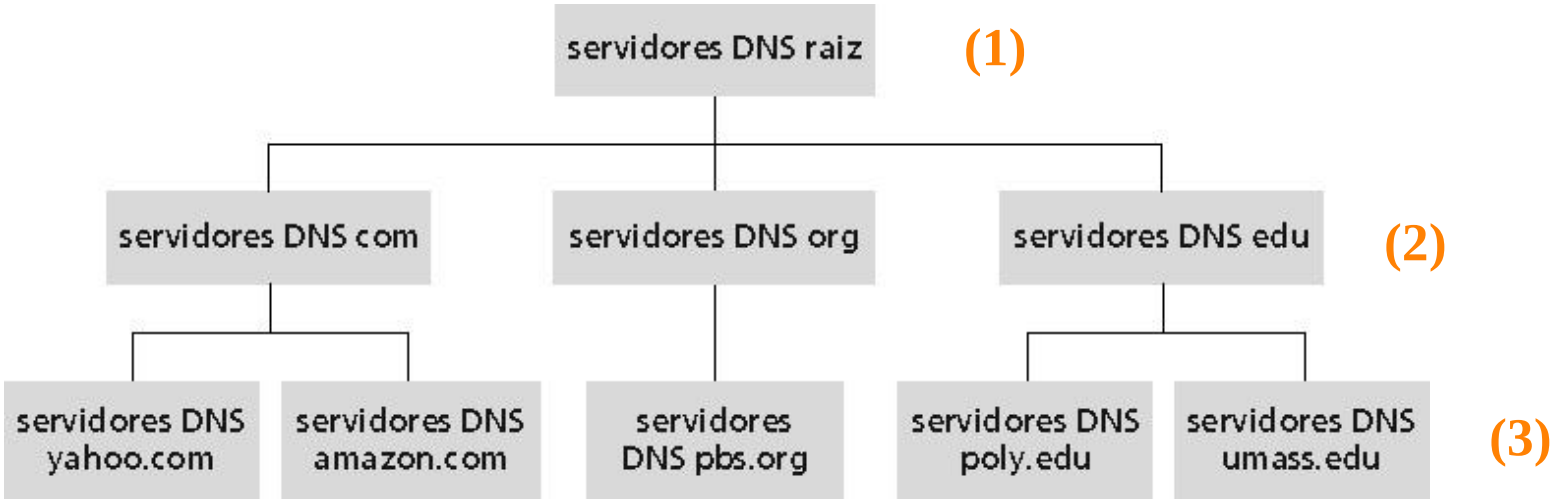
DNS

Por que não centralizar o DNS?

- Ponto único de falha (Internet inteira parada?!?!)
- Volume de tráfego (Quantas requisições por segundo no mundo?)
- Base centralizada de dados distante (Como ter um BD perto do Brasil e da Austrália?!?!)
- Manutenção (Volume de Atualização?!?!)

Enfim, porque não seria escalável!

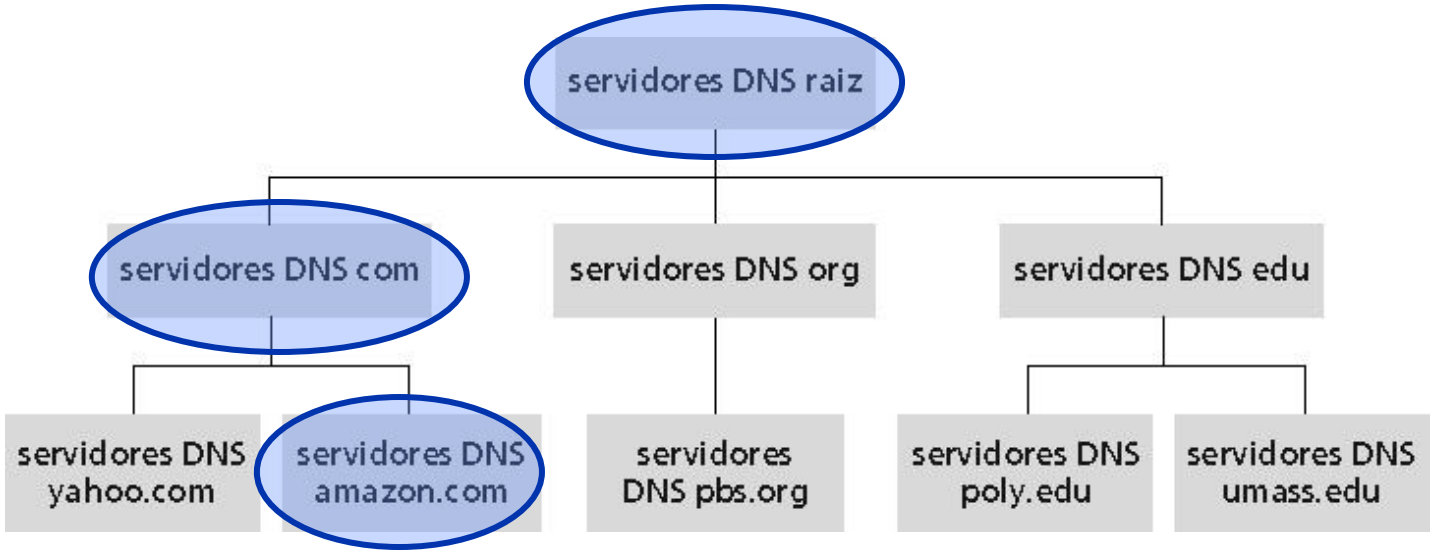
2 Base de dados distribuída, hierárquica



Classes:

- (1) Servidores de nomes raiz
- (2) Servidores de nomes de Domínio de Alto Nível (TLD)
- (3) Servidores de nomes com autoridade

2 Base de dados distribuída, hierárquica



Cliente quer o IP para **www.amazon.com**; 1ª aprox.:

- Cliente consulta:
  - Um servidor DNS raiz para encontrar o servidor DNS com.
  - O servidor DNS com para obter o servidor DNS amazon.com.
  - O servidor DNS amazon.com para obter o endereço IP para www.amazon.com.

2 DNS: servidores de nomes raiz

- Contatados pelos servidores de nomes locais que não podem resolver um nome.
- Buscam servidores de nomes autorizados se o mapeamento do nome não for conhecido.

Existem 13 servidores de nomes raiz no mundo (2010)

(Na verdade, cada um é um conjunto de servidores replicados!)

2 Servidores TLD e autoritários

Servidores top-level domain (TLD): responsáveis pelos domínios com, org, net, edu etc e todos os domínios **top-level** nacionais uk, fr, ca, jp.

- Network Solutions mantém servidores para o TLD “com” (2004)

Servidores DNS autorizados: servidores DNS de organizações que têm hosts que podem ser acessados pela Internet.

- Podem ser mantidos na organização ou ISP

Servidores de nomes locais: Não pertence estritamente a uma hierarquia.

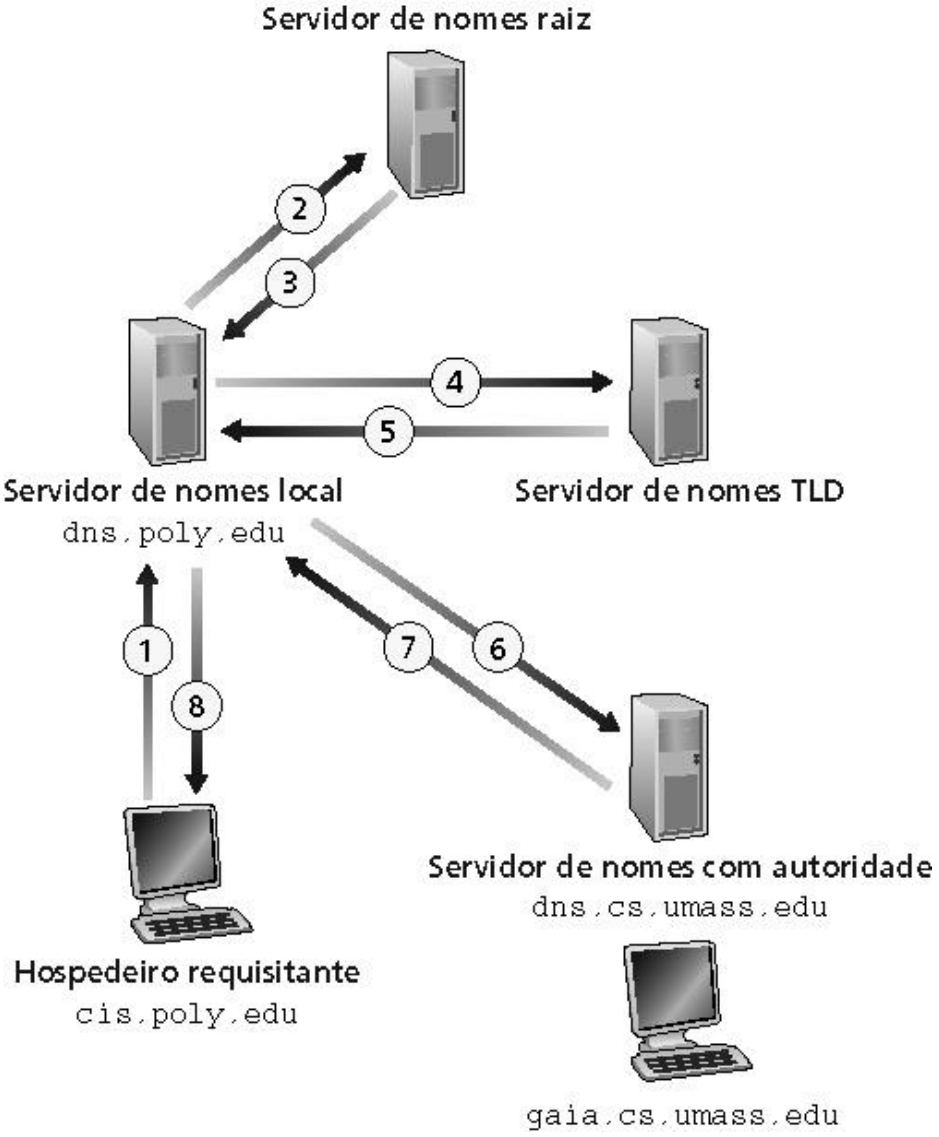
- Cada ISP possui um.
- Host solicita serviço DNS para seu servidor DNS local. Age como proxy, encaminhando as perguntas para dentro da hierarquia.

2 Consulta DNS: Exemplo

- O host em cis.poly.edu quer o endereço IP para gaia.cs.umass.edu

- Transfere a tarefa para o servidor de nomes consultado

- Resposta = “Eu não sei isto, mas pergunte a este servidor”



2 DNS: armazenando e atualizando registros

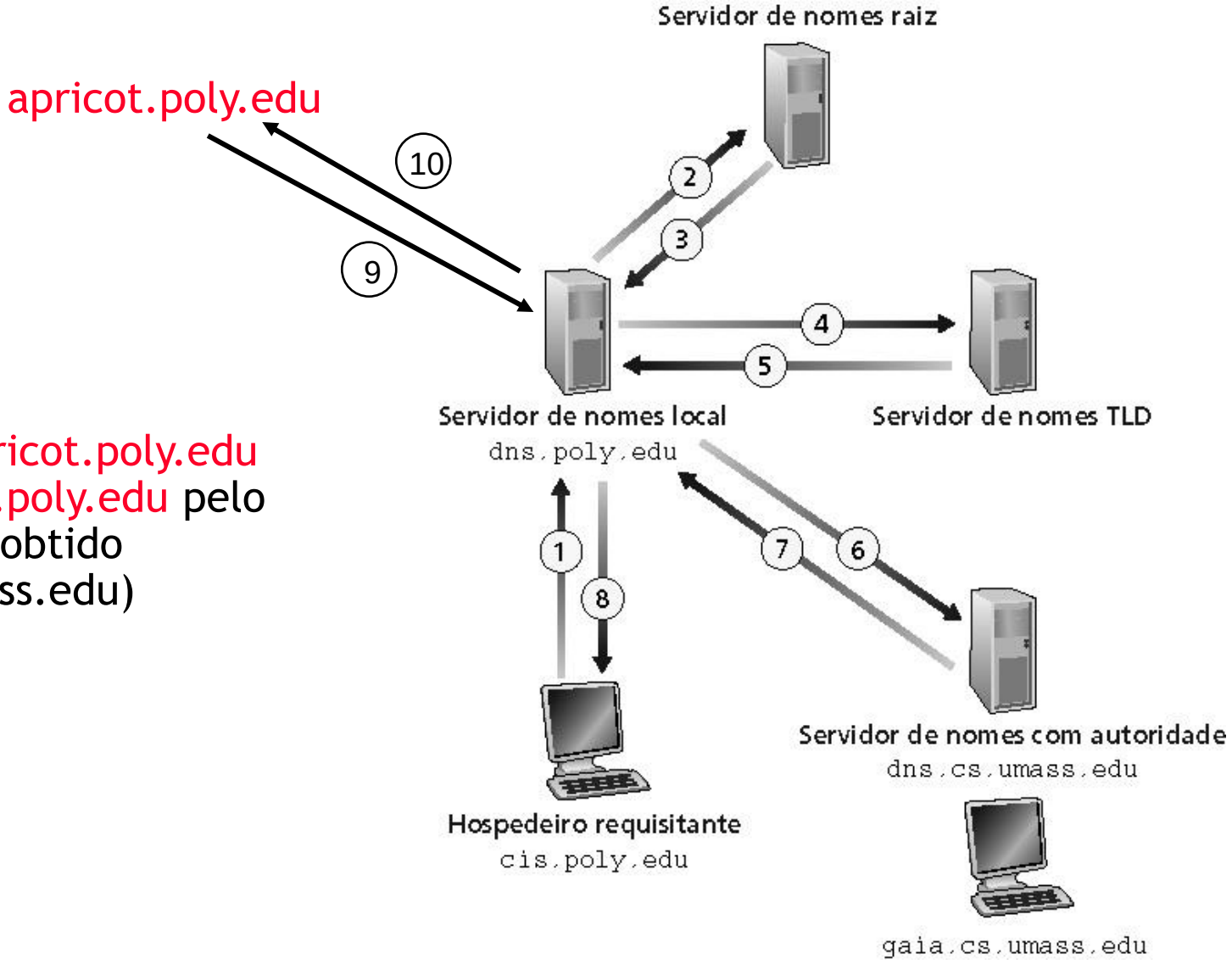
Até agora, ignoramos que:

Sempre que um servidor DNS aprende um mapeamento, armazena em cache. **(Mesmo que não seja de autoridade daquele domínio!)**

- Registro do cache desaparecem depois de um certo tempo (2 dias).

2 Exemplo de Cache

- Situação: apricot.poly.edu consulta dns.poly.edu pelo mesmo IP já obtido (gaia.cs.umass.edu)



2 Registros do DNS

DNS: base de dados distribuída que armazena registros de recursos (RR)

formato dos RR: **(name, value, type, TTL)**

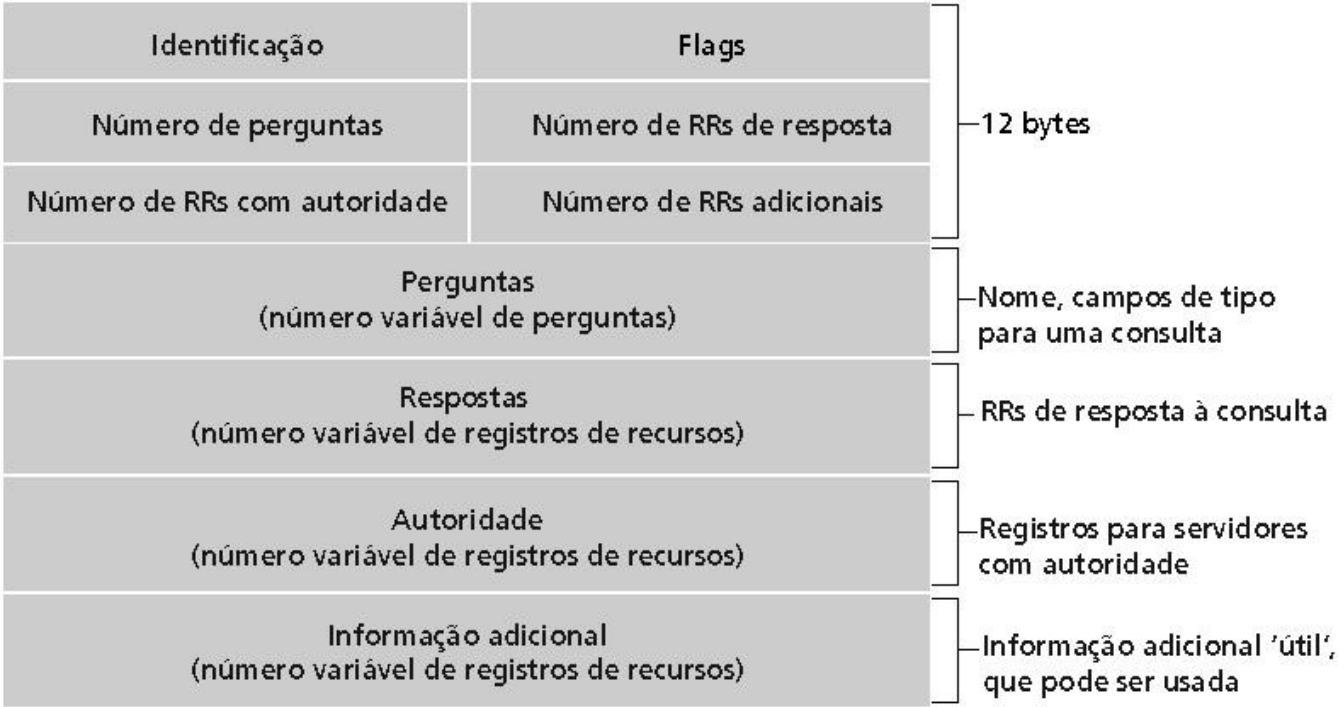
- Type = A
  - **name** é o nome do computador
  - **value** é o endereço IP
- Type = NS
  - **name** é um domínio (foo.com)
  - **value** é o IP do servidor DNS autorizado para este domínio
- TTL: tempo de vida para um recurso ser removido do cache
- Type = CNAME
  - **name** é um "apelido" para algum nome "canônico"
  - **value** é o nome canônico
  - Ex: (www.ibm.com, servereast.backup2.ibm.com, CNAME)
- Type = MX
  - **value** é o nome canônico do servidor de correio associado com **name.** Ex: (foo.com, mail.bar.foo.com, MX)

2 DNS: protocolo e mensagem

Protocolo DNS: mensagem de **consulta** e resposta, ambas com o mesmo **formato de mensagem**

Cabeçalho da msg

- ID: 16 bits para consulta (resposta usa o mesmo número)
- Flags:
  - Consulta(0)/ resposta(1)
  - Recursão desejada
  - Recursão disponível
  - Resposta é autorizada



2 DNS: Inserindo registros no DNS

- Exemplo: empresa recém-criada “Network Utopia”
- Registrar o nome networkuptopia.com num “registrar” (ex.: Network Solutions) (Lista das entidades registradoras http://www.internic.net)
  - É necessário fornecer ao registrar os nomes e endereços IP do seu servidor nomes autorizados (primário e secundário)
  - Registrar insere dois RRs no servidor TLD do domínio com:

```
(networkutopia.com, dns1.networkutopia.com, NS)
(dns1.networkutopia.com, 212.212.212.1, A)
```

- No servidor autorizado, inserir um registro Tipo A para www.networkuptopia.com e um registro Tipo MX para networkutopia.com

2 Arquiteturas de aplicação

- Cliente-servidor
- **Peer-to-peer** (P2P)
- Híbrida de cliente-servidor e P2P

2 Híbrida de cliente-servidor e P2P

Napster

- Transferência de arquivo P2P
- Busca centralizada de arquivos:
  - Conteúdo de registro dos pares no servidor central
  - Consulta de pares no mesmo servidor central para localizar o conteúdo

Instant messaging

- Bate-papo entre dois usuários é P2P
- Detecção/localização centralizada de presença:
  - Usuário registra seu endereço IP com o servidor central quando fica on-line
  - Usuário contata o servidor central para encontrar endereços IP dos vizinhos

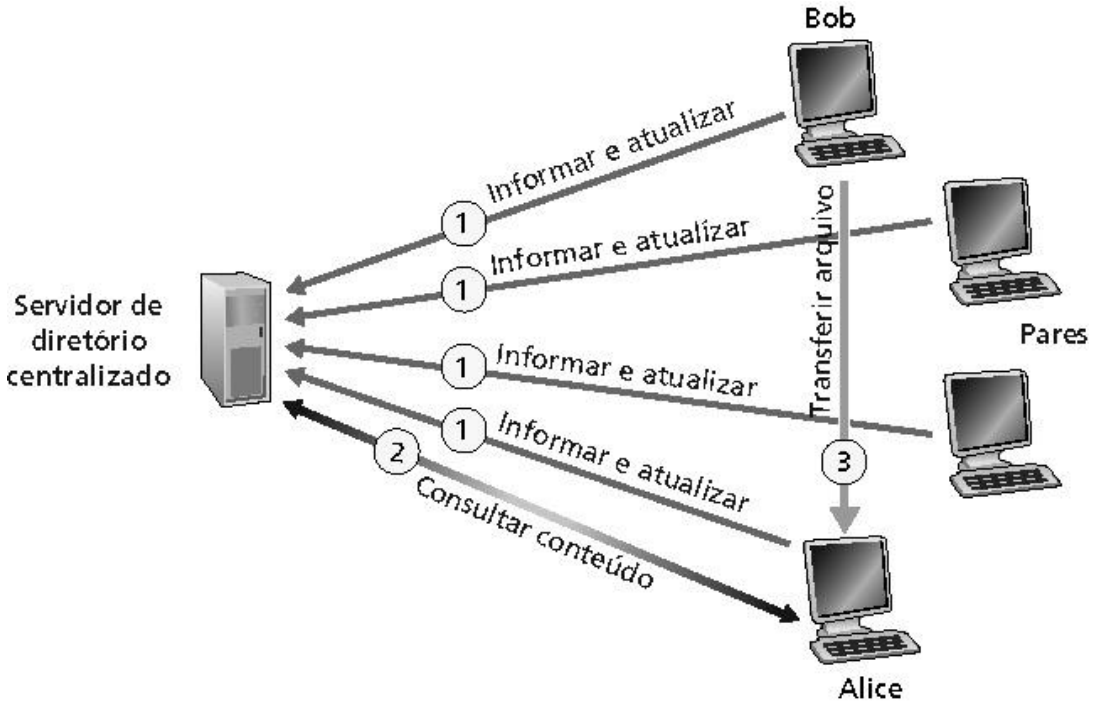
2 P2P: diretório centralizado

Projeto original “Napster”

1) Quando um par se conecta, ele informa ao servidor central:
   - Endereço IP
   - Conteúdo

2) Alice procura por “Hey Jude”

3) Alice requisita o arquivo de Bob



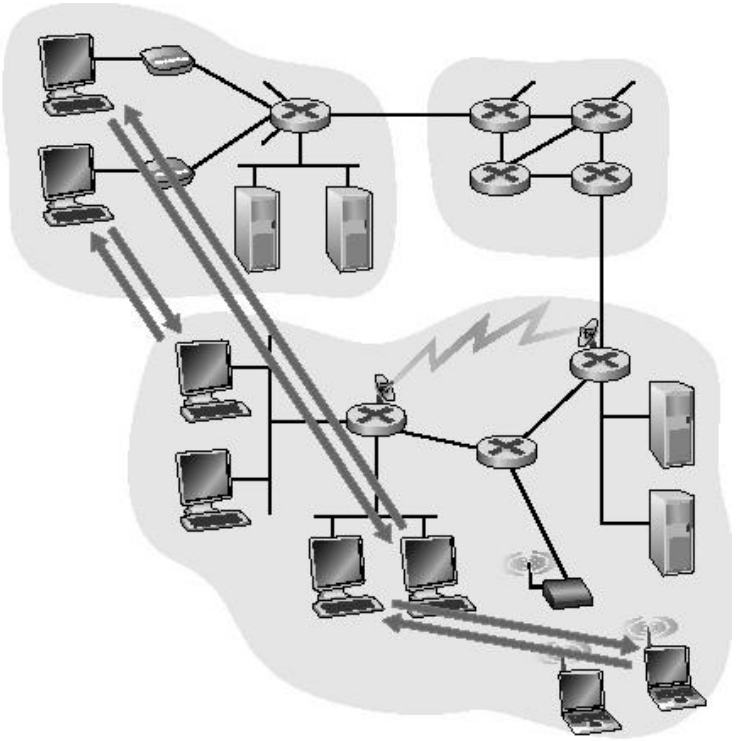
Problemas:

- Ponto único de falhas
- Gargalo de desempenho
- Infração de direitos autorais
- Transferência de arquivo é descentralizada, mas a localização de conteúdo é altamente centralizado

2 Arquitetura P2P pura

- Sistemas finais arbitrários comunicam-se diretamente
- Pares são intermitentemente conectados e trocam endereços IP
- Ex.: Gnutella
  - Totalmente distribuído
    - Sem servidor central
  - Protocolo de domínio público

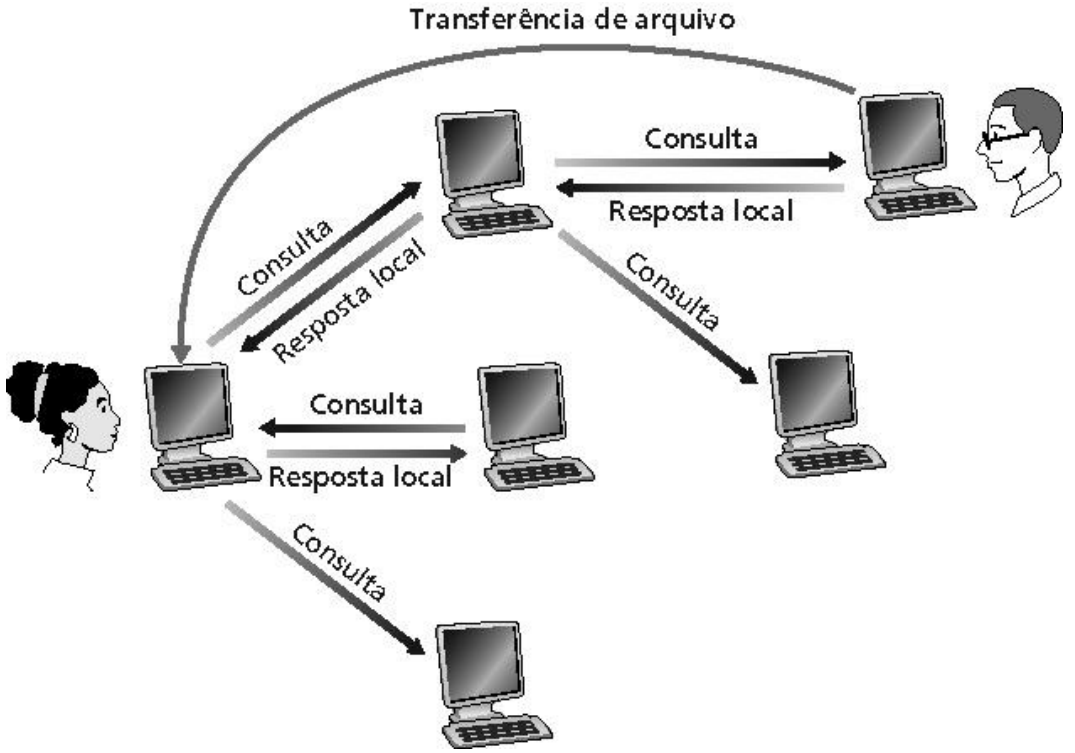
Altamente escaláveis mas difíceis de gerenciar.



b. Aplicação P2P

2 Gnutella: protocolo

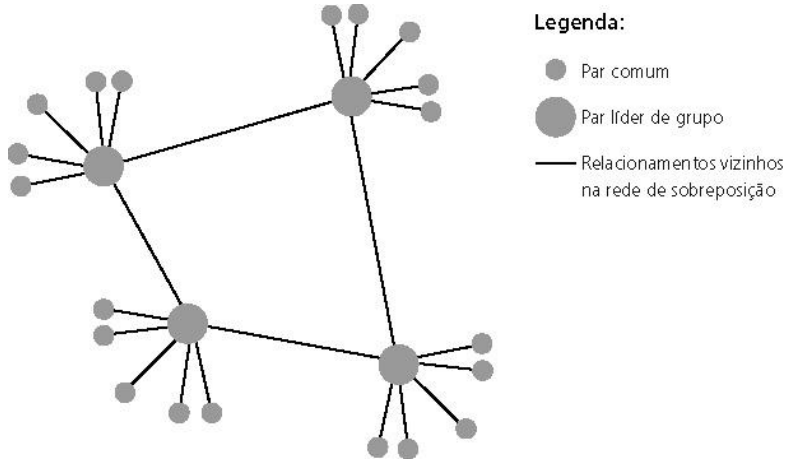
- Mensagem de consulta é enviada pelas conexões TCP existentes
- Os pares encaminham a mensagem de consulta



Escalabilidade: flooding de alcance limitado

2 Explorando heterogeneidade: KaZaA

- Cada par é ou um líder de grupo ou está atribuído a um líder de grupo
  - Conexão TCP entre o par e seu líder de grupo
  - Conexões TCP entre alguns pares de líderes de grupo
- O líder de grupo acompanha o conteúdo em todos os seus "discípulos"



2 Dúvidas?