Universidade Federal de Ouro Preto

Camada de Aplicação

Referências:

- Redes de Computadores. **A. S. Tanenbaum.** Campus/Elsevier, 2011 - Capítulo 7
- Redes de Computadores e a Internet. **J. Kurose, K. Ross.** Pearson, 2010 - Capítulo 2

Camada de Aplicação

- **5.1 Princípios de aplicações de rede**
  - Arquitetura de rede: Cliente-Servidor e P2P
- **5.2 DNS**
- **5.3 A Web e o HTTP**
  - Arquitetura
  - Páginas estáticas e páginas dinâmica

Princípios de aplicação de rede

- Aplicações são a razão de ser de uma rede de computadores.
  - Se não fosse possível inventar aplicações, não haveria necessidade de projetar protocolos de rede para suportá-las.
- A camada de aplicação é um terreno familiar, pois conhecemos muitas aplicações que dependem dos protocolos que estudamos.
- **Unidade de dados:** mensagem da aplicação.
- Algumas aplicações de rede:
  - Web
  - e-mail
  - mensagem instantânea
  - streaming de áudio e vídeo
  - compartilhamento de arquivos P2P
  - jogos em rede multiusuários
  - vídeoconferência

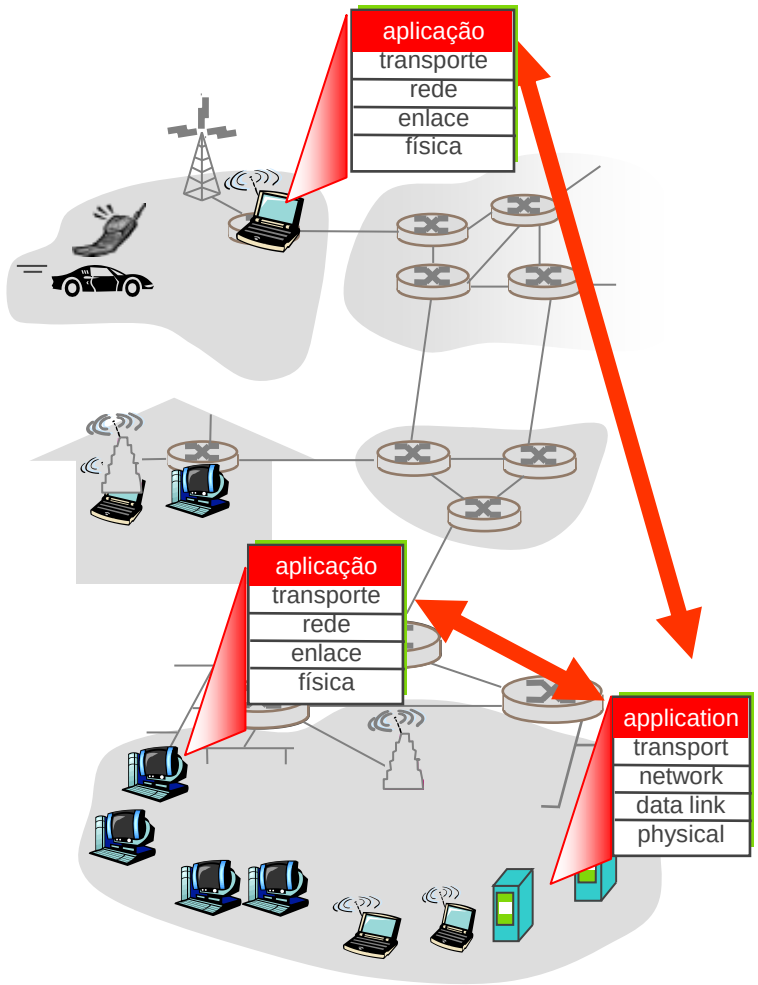
Criando uma aplicação de rede

Escrever programas que

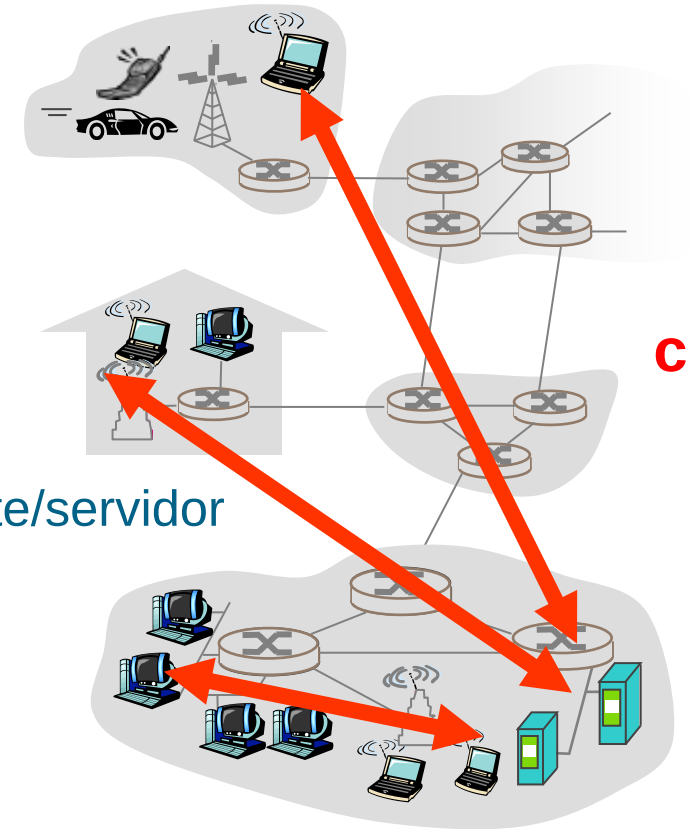
- executem em (diferentes) *sistemas finais;*
- se comuniquem pela rede;
- Ex: software de servidor Web se comunica com software de navegador Web

Não é preciso desenvolver software para dispositivos do núcleo da rede

- dispositivos do núcleo da rede não executam aplicações do usuário;
- as aplicações nos sistemas finais permitem rápido desenvolvimento e propagação;
- Arquitetura da rede é fixa.



Arquitetura cliente-servidor



cliente/servidor

servidor:

- host sempre ligado
- endereço IP permanente (ou não)

clientes:

- comunicam-se diretamente com o servidor
- podem estar conectados intermitentemente
- podem ter endereços IP dinâmicos
- não há comunicação cliente-cliente

Arquitetura cliente-servidor

- **Aplicações mais conhecidas que empregam a arquitetura cliente-servidor:**
  - Web, e-mail, mensagem instantânea, acesso remoto, etc...
- Em muitas aplicações um único servidor não é capaz de atender a todas as requisições de clientes;
  - Criação de Data Centers;
  - Serviços em nuvem;
  - Sistemas distribuídos;
- Alguns serviços exigem muita infraestrutura:
  - Serviços de busca: Google
  - e-commerce: Amazon, Ebay, Americanas, ...
  - Webmail: Gmail, Yahoo, ...
  - redes sociais: WhatsApp, Facebook, Instagram, X, ...
  - Streaming de vídeo: youtube, Netflix, Amazon Prime ...

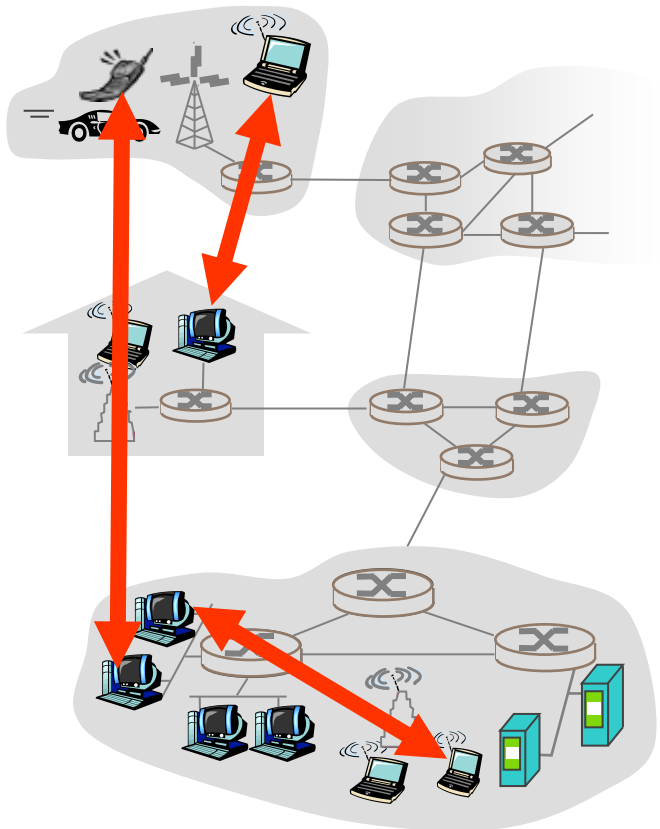
Data Centers da Google

- custo estimado de um Data Center: $600M
- Google gastou $4,3B em 2022 em novos Data Centers
- cada Data Center usa de 50 a 100 megawatts de potência
  - Google consome mais energia que cidade com 200 mil habitantes http://tecnoblog.net/76530/google-consumo-energia/

Arquitetura P2P pura

- *nenhum* servidor ligado 100% do tempo
- sistemas finais arbitrários se comunicam diretamente
- pares são conectados intermitentemente e mudam endereços IP

altamente escalável, mas difícil de administrar



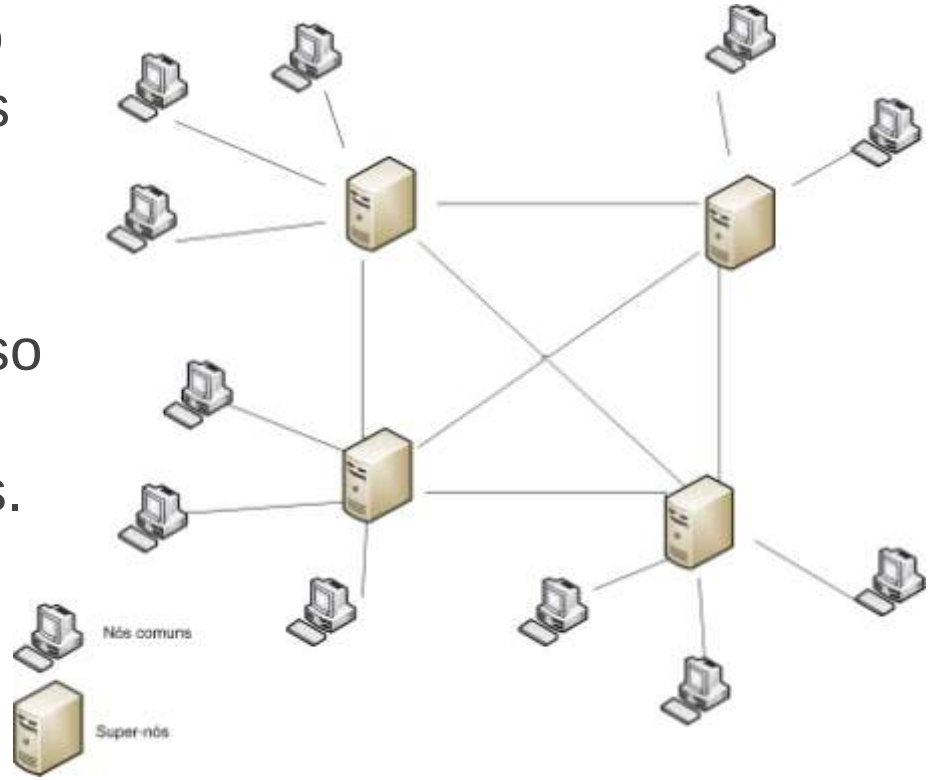
peer-peer

Exemplo:

GNUTELLA
search and find multiple files

P2P Híbrido

- Supernós concedem acesso aos hosts, indexam recursos compartilhados e liberam a busca por estes recursos.
- Quando localizado, o recurso pode é obtido a partir da interação direta entre os nós.
- Uma falha em um supernó pode ser tolerada por outro.



Exemplos:

Endereçando processos

- **para receber mensagens, processo deve ter *identificador***
- **host tem endereço IP exclusivo de 32 bits**
- ***P:*** **Basta o endereço IP do hospedeiro em que o processo é executado para identificar o processo?**

- *R:* Não, *muitos* processos podem estar rodando no mesmo hospedeiro
- ***Identificador* inclui endereço IP e números de porta associados ao processo no hospedeiro.**

Exemplos de número de porta:

- servidor HTTP: 80
- servidor de correio: 25

Que serviço de transporte uma aplicação precisa?

Perda de dados

- algumas aplicações (voz sobre IP) toleram algumas perdas
- Aplicações como transferência de arquivos, http e email exigem transferência de dados 100% confiável

Vazão

- algumas aplicações (ex. videoconf.) exigem uma vazão mínima para serem “eficazes”.
- outras aplicações (elásticas) utilizam qualquer vazão que receberem .

Temporização

- Aplicações como jogos multiusuários e chamadas de voz sobre IP exigem pouco atraso para serem “eficazes”.

Segurança

- criptografia, integridade de dados,…

A camada de transporte oferece opções como o TCP e UDP

DNS: Domain Name System

Pessoas têm muitos identificadores: CPF, nome, identidade, passaporte, matrícula, ...

- "Oi, eu sou o 056.785.251-66. Esta é a minha amiga, a 077.023.074-95!"

- O endereço www.google.com – usado pelos humanos
  - *Fornecem pouco, se é que alguma, informação sobre a localização do host na Internet* ☹

hosts e roteadores da Internet:

- endereço IP – usado para endereçar datagramas

P: Como mapear entre endereço IP e nome?

DNS: Domain Name System

Domain Name System:

- ***banco de dados distribuído*** **implementado na hierarquia de muitos** ***servidores de nomes.***
- ***protocolo em nível de aplicação*** → **hospedeiro, roteadores, servidores de nomes se comunicam para** ***resolver*** **nomes (tradução endereço/nome)**
  - Utiliza a porta UDP 53 - Consultas e respostas

DNS: Domain Name System

- Browser (cliente http), executado na máquina de um usuário, requisita URL *www.google.com.br*
  - Para que a máquina do usuário possa enviar a msg de requisição http ao servidor web, precisa obter o IP de www.google.com.br

1 – Máquina do usuário executa o lado cliente da aplicação DNS;

2 – Browser extrai hostname www.google.com.br e passa para o lado cliente da aplicação DNS;

3 – Cliente DNS envia consulta contendo nome do host para um servidor DNS;

4 – Cliente DNS finalmente recebe resposta, que inclui o IP correspondente ao nome do host (216.239.38.120);

5 – Browser recebe IP do DNS, pode abrir uma conexão TCP com o servidor WEB.

Serviços de DNS

- **tradução nome de hospedeiro -> endereço IP**
- **apelidos de hospedeiro**
  - nomes canônicos: mais fáceis de lembrar.
    - *relayl.west-cost.enterprise.com → www.enterprise.com*
- **apelidos de servidor de correio**
  - Endereços de e-mail devem ser fáceis de guardar.
    - *bob@hotmail.com*
    - *hotmail.com → relay1.west-cost.hotmail.com*
- **distribuição de carga**
  - servidores Web replicados: conjunto de endereços IP para um nome canônico
  - CDNs: Sites muito acessados como os e-commerce são replicados em servidores distintos, com um IP diferente..

DNS

Por que não centralizar o DNS?

- **único ponto de falha**
- **volume de tráfego**
- **banco de dados centralizado distante**
- **manutenção**

Não é *escalável!*

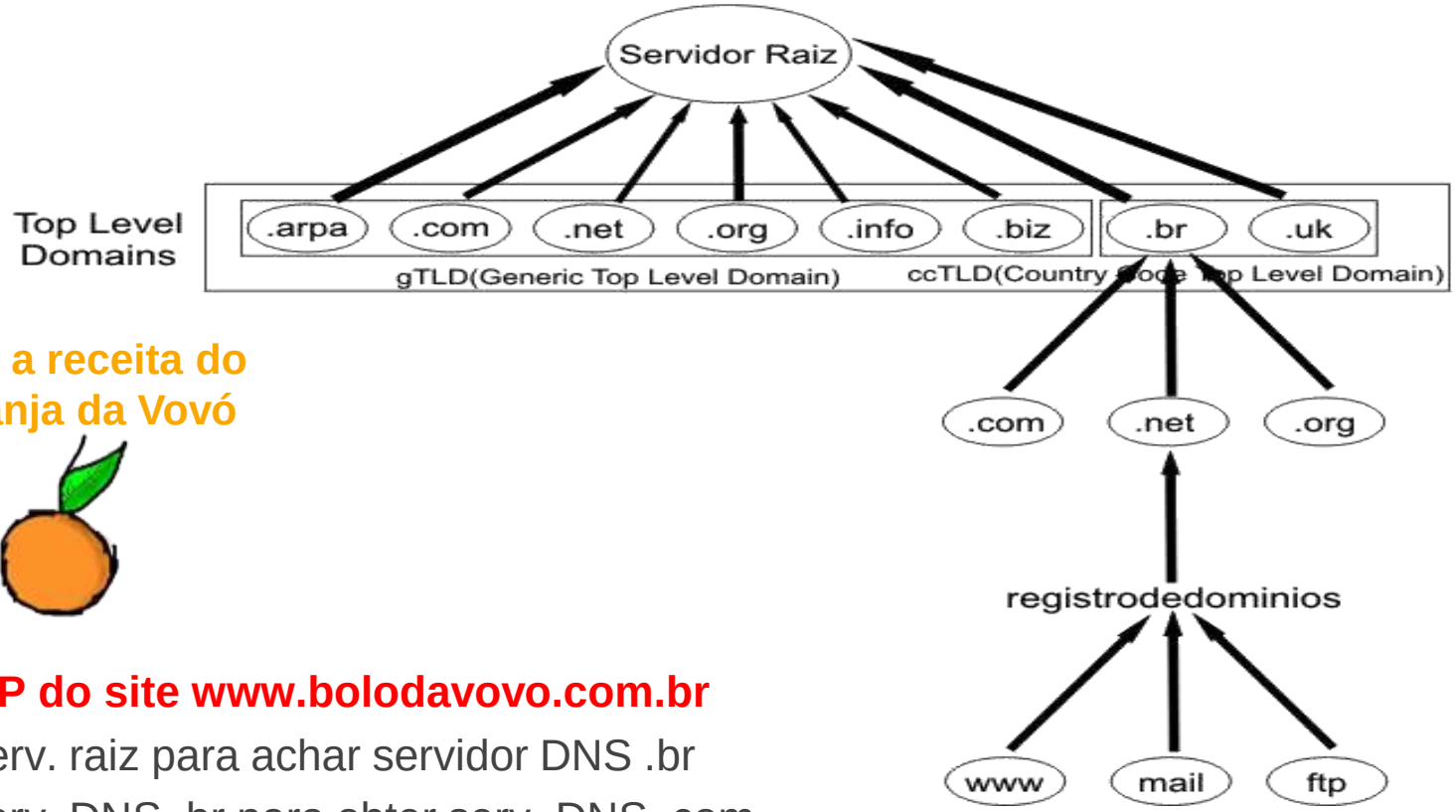
DNS

O DNS é aplicação, mas...

- HTTP, FTP, SMTP e DNS são protocolos da camada de aplicação, já que roda em sistemas finais e dependem de protocolos da camada de transporte para transferir dados.
- DNS é bastante diferente das aplicações WEB, transferência de arquivo e e-mail...
- DNS não é uma aplicação que o usuário interage diretamente.
- Fornece uma função interna da Internet, para aplicações de usuários e outros softwares conectados à Internet.

Banco de dados distribuído, hierárquico

Para tratar da questão da escala, o DNS usa um grande número de servidores, organizados de maneira hierárquica e distribuída.



Website com a receita do Bolo de laranja da Vovó

Cliente quer IP do site www.bolodavovo.com.br

- consulta serv. raiz para achar servidor DNS .br
- consulta serv. DNS .br para obter serv. DNS .com
- consulta serv. DNS .com para obter serv. DNS bolodavovo.com.br
- consulta serv. DNS bolodavovo.com.br para obter endereço IP para o site

DNS: Servidores de nomes raiz

- **contatados por servidores de nomes locais que não conseguem traduzir nome**
- **servidores de nomes raiz:**
  - contata servidor de nomes com autoridade se o mapeamento não for conhecido;
  - obtém mapeamento;
  - retorna mapeamento ao servidor de nomes local

* 13 servidores de nomes raiz no mundo.

* cada servidor é um conglomerado de servidores.

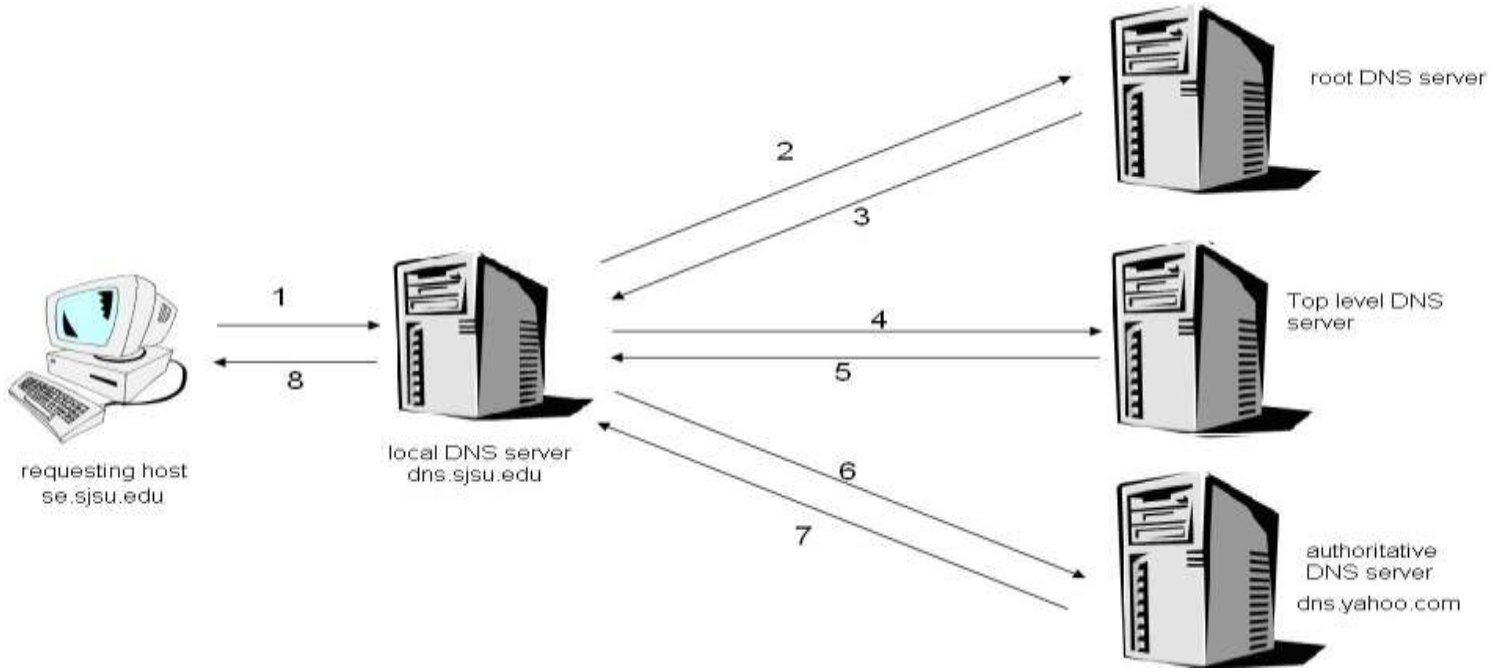
TLD e servidores com autoridade

- **servidores de domínio de alto nível (TLD) :**
  - responsáveis por com, org, net, edu, etc.
  - todos os domínios de país de alto nível: br, uk, fr, ca, jp.

- **servidores DNS com autoridade:**
  - servidores DNS da organização, provendo nome de hospedeiro com autoridade a mapeamentos IP para os servidores da organização (p. e., Web, correio).
  - podem ser mantidos pela organização ou provedor de serviços

Servidor de nomes local

- **não pertence estritamente à hierarquia**
- **cada ISP (ISP residencial, empresa, universidade) tem um.**
  - também chamado “servidor de nomes default”
- **Quando o host faz consulta ao DNS, consulta é enviada ao seu servidor DNS local**
  - atua como proxy, encaminha consulta para hierarquia

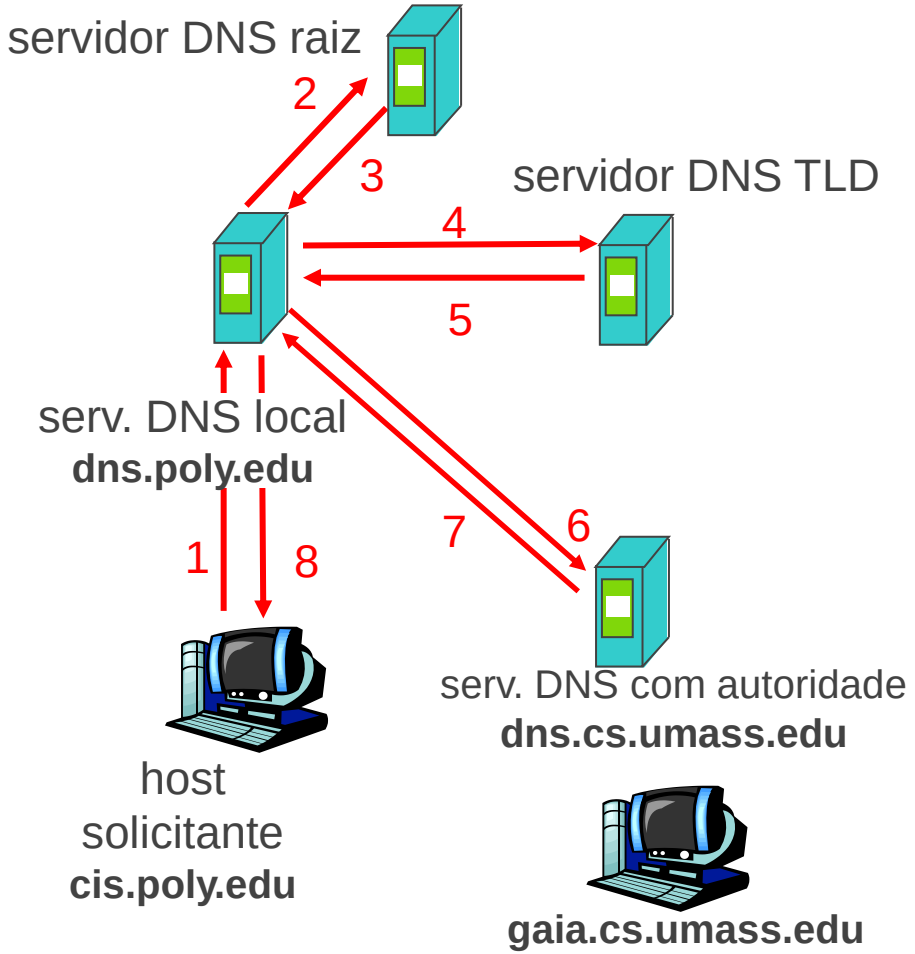


Exemplo de resolução de nome DNS

- **hospedeiro em cis.poly.edu quer endereço IP para gaia.cs.umass.edu**

  consulta repetida:

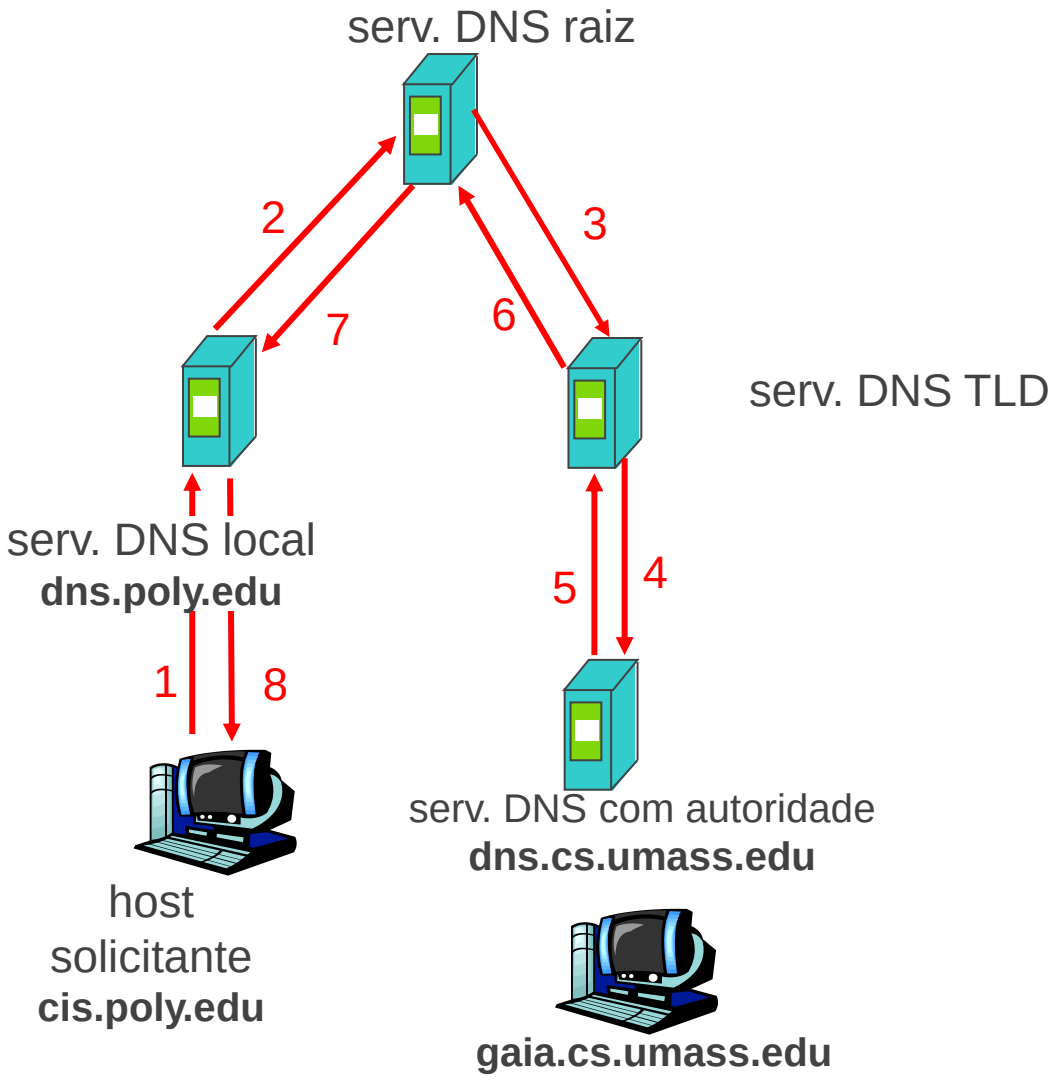
  - servidor contatado responde com nome do servidor a contatar
  - "não conheço esse nome, mas pergunte a este servidor"



Exemplo de resolução de nome DNS

consulta recursiva:

- coloca peso da resolução de nome sobre o servidor de nomes contatado
- carga pesada?



DNS: caching e atualização de registros

- **quando (qualquer) servidores de nomes descobre o mapeamento, ele o mantém em *cache***
  - entradas de cache esgotam um tempo limite (desaparecem) após algum tempo
  - servidores TLD normalmente são mantidos em caches nos servidores de nomes locais
    - Assim, os servidores de nomes raiz não são consultados com frequência
- **mecanismos de atualização/notificação em projeto na IETF**
  - RFC 2136
  - http://www.ietf.org/html.charters/dnsext-charter.html

Protocolo DNS, mensagens

campos de nome e tipo para uma consulta

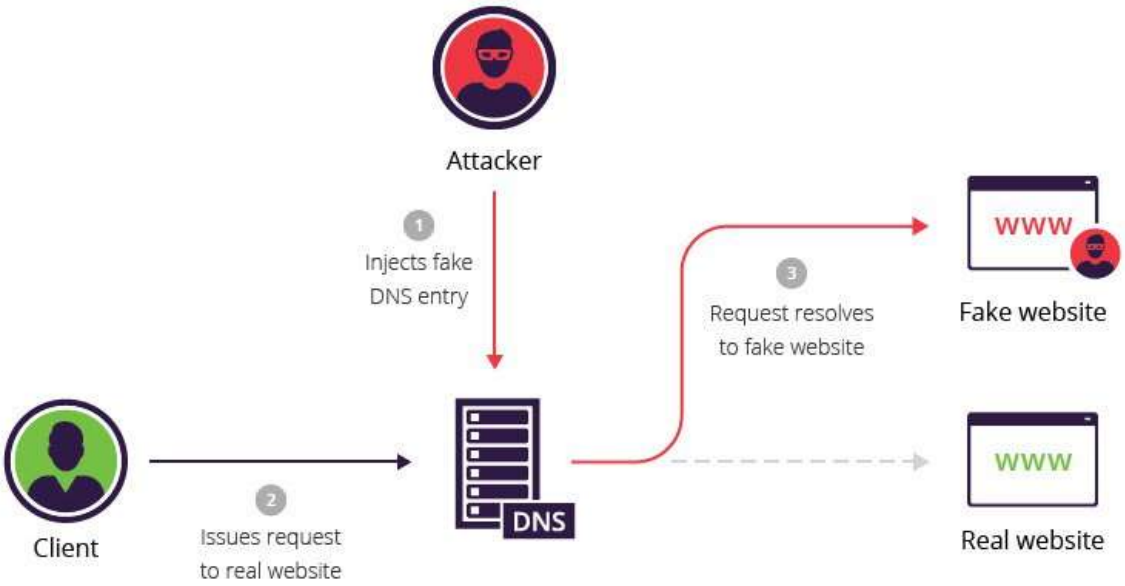
RRs na resposta à consulta

registros para servidores com autoridade

informação adicional "útil" que pode ser usada

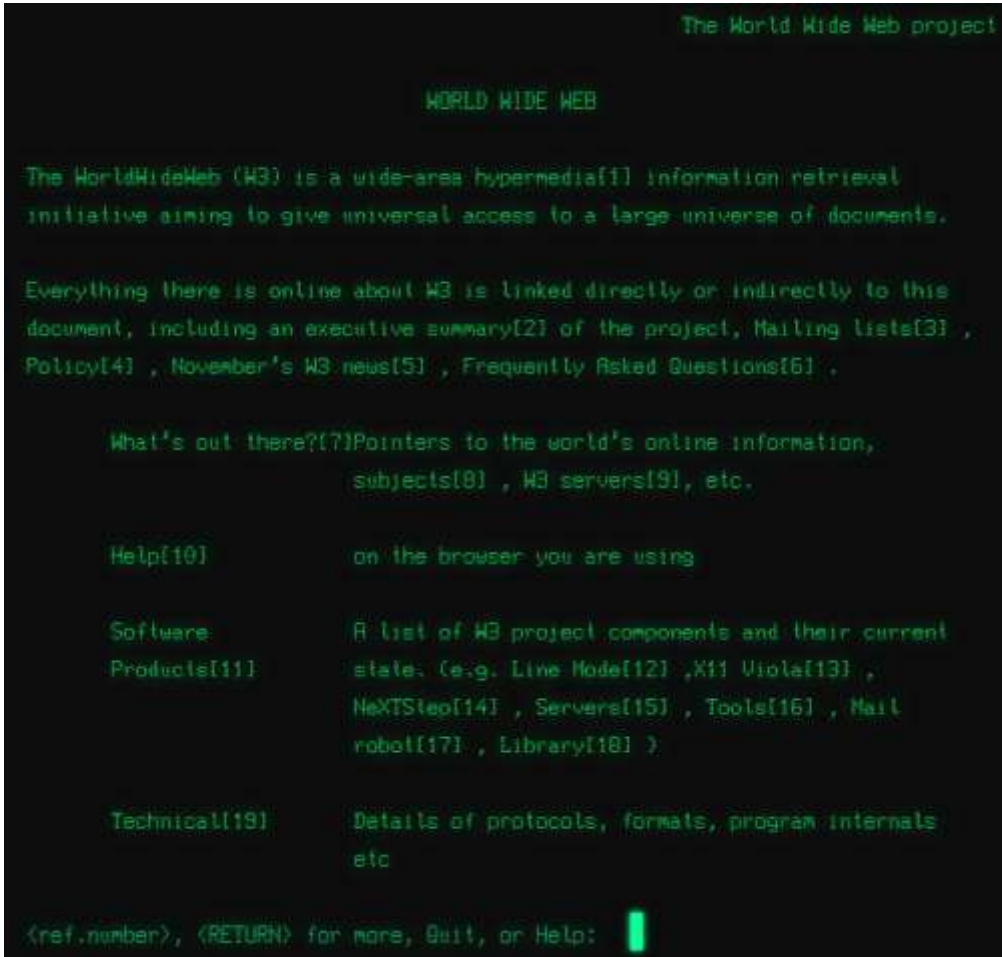
DNS: Segurança

- **Ataques:**
  - Floodind
  - DNS spoofing



A World Wide Web (WWW)

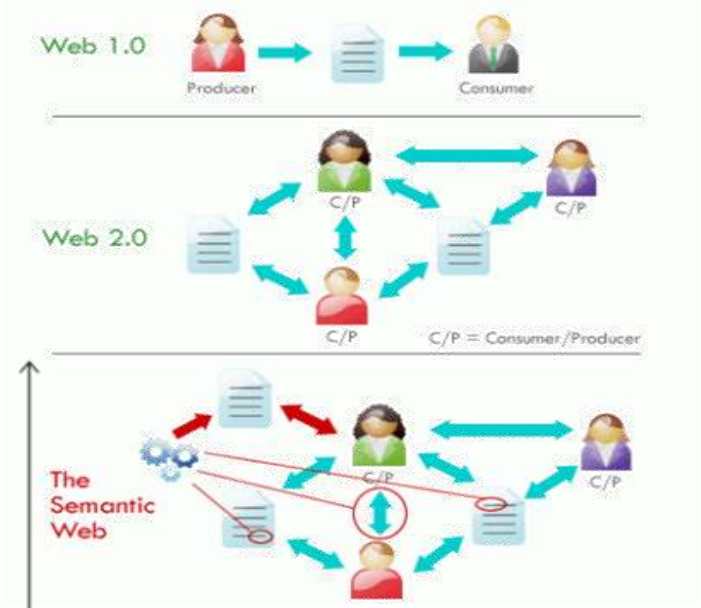
- Até a década de 90 a Internet só era conhecida por pesquisadores, acadêmicos e grandes corporações:
- Transferência de arquivos, correio eletrônico, acesso remoto, ...
- Em 20/12/90 Tim Berners-Lee colocou em operação a primeira página Web no endereço http://info.cern.ch/, e que ainda hoje pode ser acessada!!

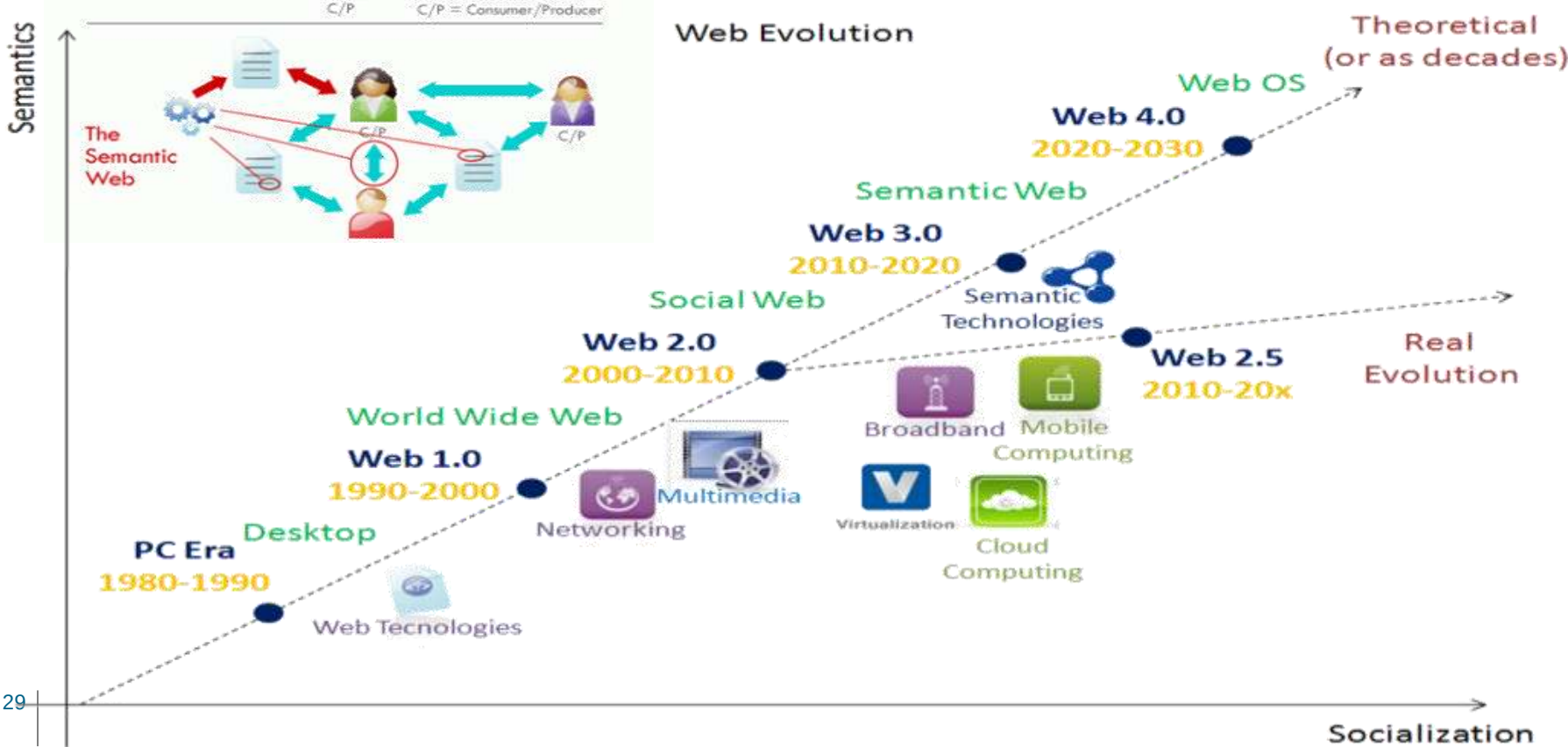


A Web

- Muito atrativa aos usuários: **a WEB funciona *sob demanda*!**
- Além disso é:
  - Muito fácil para usar;
  - Uma imensa variedade de informações sobre quase todos os assuntos imagináveis;
  - Qualquer um pode se transformar em um editor ou produtor de conteúdo a um baixo custo;
- Ao longo das últimas décadas web sites cresceram exponencialmente, atingindo quase 5.38 bilhões em 2024.
- Alguns se tornaram extremamente populares:
  - Amazon (1994) - mercado de US$ 575 bilhões;
  - eBay (1995) – US$ 40 bilhões;
  - Google (1998) - US$ 305 bilhões;
  - Facebook (2004) – US$ 135 bilhões;

A evolução da Web

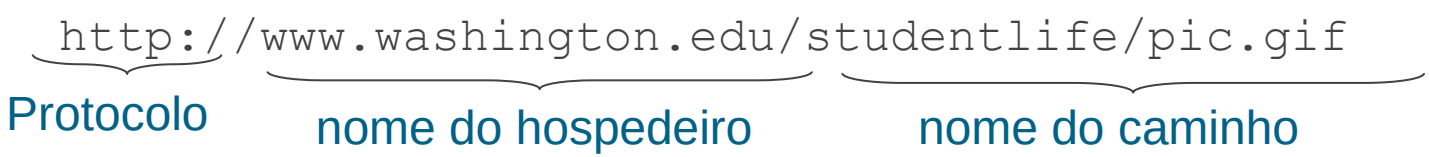




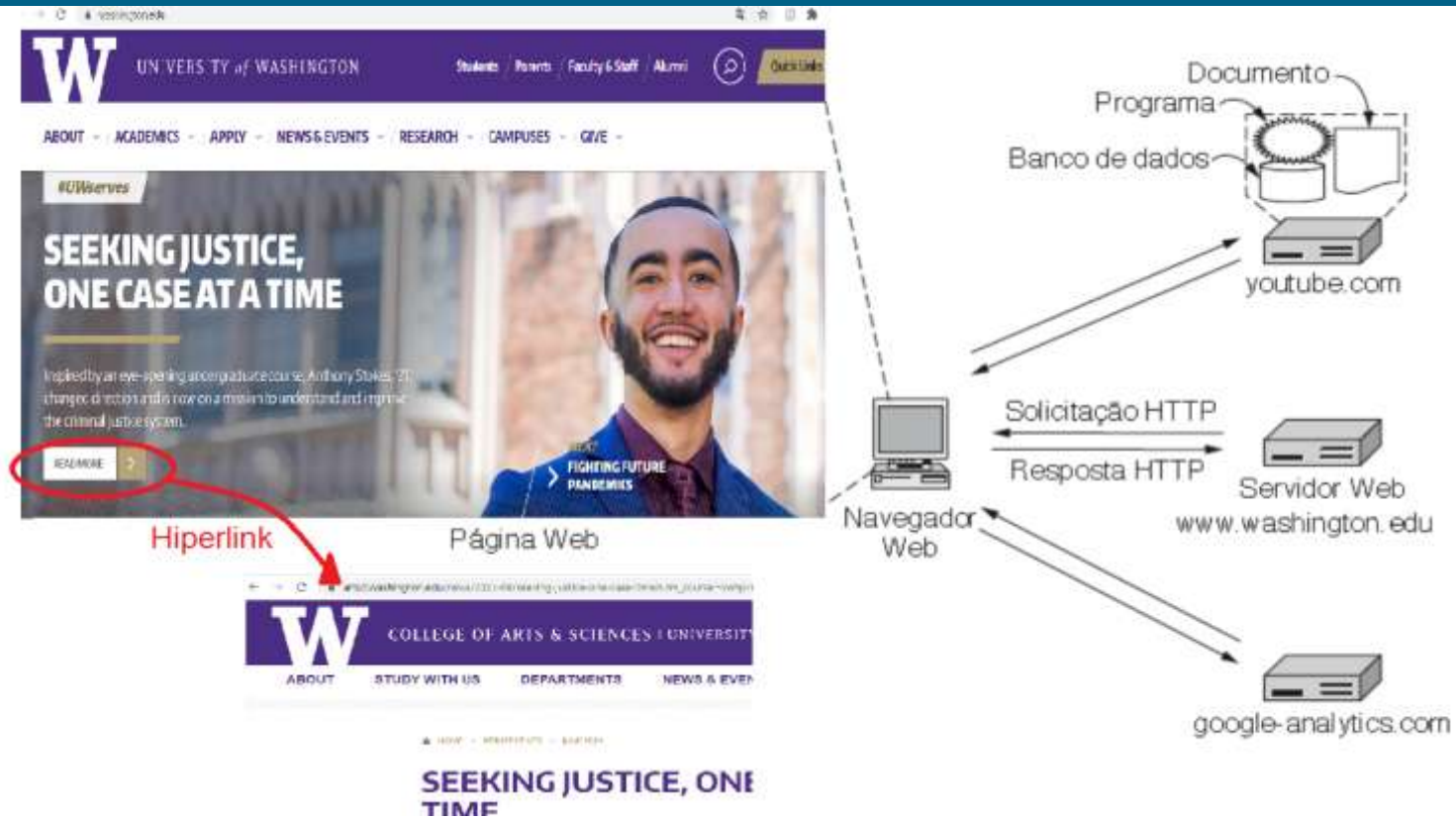
Arquitetura da Web

primeiro, algum jargão

- página Web consiste em objetos
- objeto pode ser arquivo HTML, imagem JPEG, applet Java, arquivo de áudio,…
- página Web consiste em arquivo HTML básico que inclui vários objetos referenciados
- cada objeto é endereçável por um URL (Uniform Resource Locator)
- exemplo de URL:



Arquitetura da Web



- Usuário acessando página da Universidade de Washington em um browser;
- Esta página contém elementos de mídias variadas e links para outras páginas;
- Seu conteúdo pode estar no mesmo host da página anterior ou não, e isso é transparente para o usuário;

Arquitetura da Web

Lado Cliente:

- Exemplo de URL: http://www.cs.washington.edu/index.html;
- O navegador realiza uma série de tarefas para exibir a URL:
  1. Obtém o IP do servidor solicitando ao DNS o endereço de *www.cs.washington.edu*;
  2. Estabelece uma conexão TCP com o servidor na porta 80;
  3. Solicita a página *index.html* usando um comando HTTP;
  4. Caso a página incluir links para outros recursos para exibição (URLs), buscará estes recursos da mesma maneira;
  5. Exibe a página;
  6. Encerra a conexão após um tempo.

Lado Cliente:

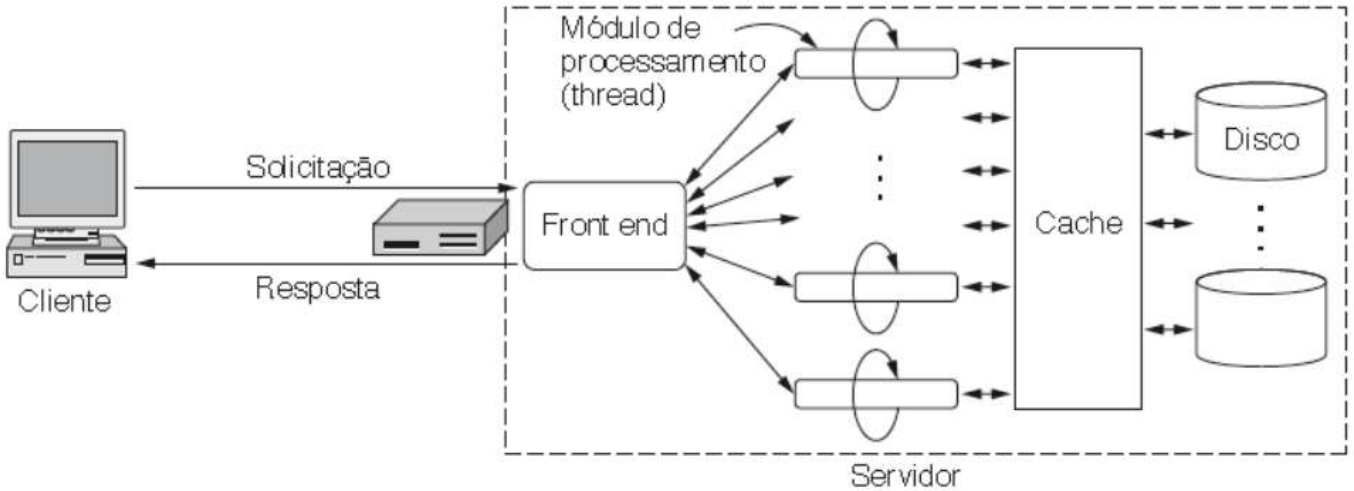
Diferentes protocolos que podem ser usados em uma URL:

Nome	Usado para	Exemplo
http	Hipertexto (HTML).	http://www.decsi.ufop.br/
https	Hipertexto com segurança.	https://www.bank.com/accounts/
ftp	FTP.	ftp://ftp.cs.vu.nl/pub/minix/README
file	Arquivo local.	file://usr/suzana/prog.c
mailto	Envio de e-mail.	mailto:fulano@acm.org
rtsp	*Streaming* de mídia.	rtsp://youtube.com/montypython.mpg

Arquitetura da Web

Lado Servidor:

- Arquitetura de um servidor Web:



- Tarefas:
  - Aceitar uma conexão TCP de um cliente (um navegador);
  - Obter o caminho até a página (ou programa);
  - Obter o arquivo (ou gerar o conteúdo dinâmico);
  - Enviar o conteúdo ao cliente;

Cookies

- Cookies são arquivos criados pelos sites que você visita.
- Eles tornam sua experiência on-line mais fácil, economizando informações de navegação.
  - Ex: produtos em uma cesta de compra em um site de e-commerce.
- Os sites podem manter seu login, lembrar suas preferências do site e fornecer conteúdo relevante localmente.

- Cookies estão envolvidos com algumas questões de privacidade e segurança.
- Muitos desativam seu uso no browser.

Tipos de Cookies

Cookies Essenciais:

☐ Necessários para funcionamento do site: autenticação e carrinhos de compras.

☐ Ex: e-commerce que lembra os itens do carrinho.

Cookies de Desempenho:

☐ Coletam dados sobre o uso do site, como páginas mais visitadas e tempo de navegação.

☐ Não armazenam informações pessoais, apenas estatísticas.

☐ Ex: Google Analytics para entender o comportamento dos usuários.

Cookies Funcionais:

☐ Armazenam preferências do usuário, como idioma, tema e região.

☐ Melhoram a experiência do usuário ao personalizar a navegação.

Cookies de Publicidade (Marketing):

☐ Usados para rastrear os hábitos de navegação do usuário e exibir anúncios personalizados.

☐ Utilizados por redes de publicidade como o Google Ads e o Facebook Ads.

☐ Ex: Você Pesquisa um celular e depois aparece anúncios do produto em sites.

Cookies

- A LGPD (Lei Geral de Proteção de Dados) no Brasil e a GDPR (*General Data Protection Regulation*) na Europa, os sites precisam obter consentimento explícito do usuário antes de armazenar cookies, especialmente os de rastreamento e publicidade..

Páginas estáticas

- Forma mais simples de página Web, elas são armazenadas em um servidor, que as retorna para serem diretamente exibidas no browser quando solicitadas;

- Normalmente as páginas estáticas são desenvolvidas em linguagem **HTML** (***HyperText Markup Language***):
  - Linguagem de marcação que utiliza ***tags*** para determinar a estrutura e formatação do conteúdo;
  - Exemplos de *tags*:
    - ** **negrito** **;
    - *<img> ... </img>* para inserir imagem;
    - *<body> ... </body>* para determinar o conteúdo da página;
    - *<a> ... * para definir hiperlinks.

Páginas estáticas

- A HTML já passou por várias evoluções:

Item	HTML 1.0	HTML 2.0	HTML 3.0	HTML 4.0	HTML 5.0
Hiperlinks	X	X	X	X	X
Imagens e listas	X	X	X	X	X
Mapas e imagens ativas		X	X	X	X
Formulários		X	X	X	X
Equações			X	X	X
Barras de ferramentas			X	X	X
Tabelas			X	X	X
Recursos de acessibilidade				X	X
Objetos inseridos				X	X
Folhas de estilo				X	X
Scripting				X	X
Vídeo e áudio					X
Gráficos e vetores em linha					X
Representação XML					X
Threads em segundo plano					X
Armazenamento pelo navegador					X
Tela de desenho					X

Páginas estáticas

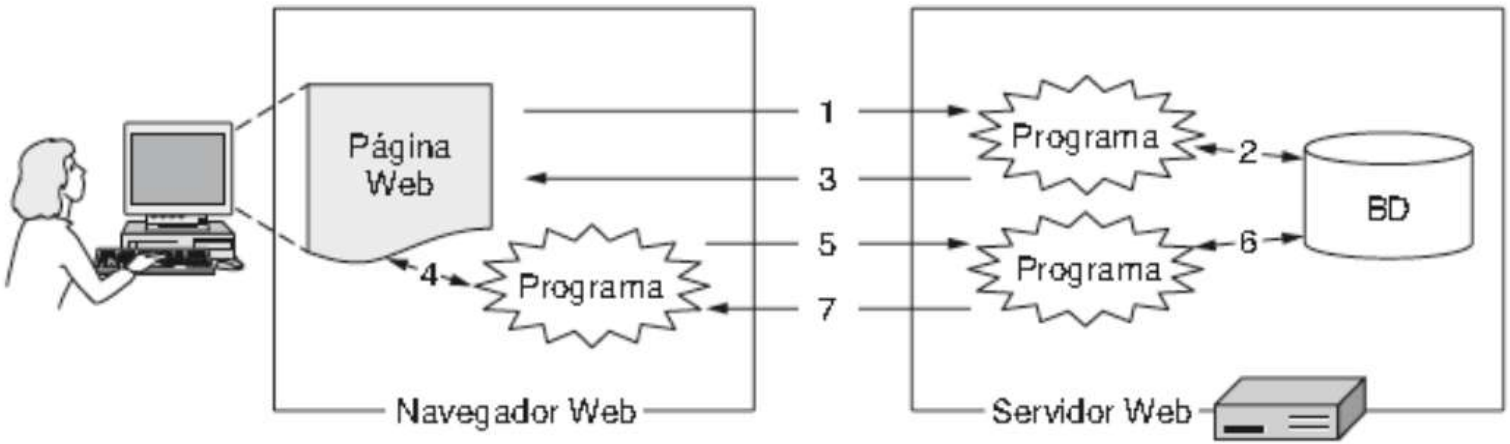
- O objetivo original do HTML era de apenas estruturar as páginas:
  - A formatação ficaria a cargo dos navegadores;
  - No entanto, desenvolvedores queriam alterar formatação, e vários recursos de formatação foram inseridos;
- Estes recursos levaram a alguns incômodos:
  - Poluição do código HTML;
  - Problema de portabilidade.
- Para resolver os problemas de formatação foi criado o conceito de folha de estilo, o **CSS** (***Cascade Style Sheets***), que separa o código estrutural (HTML) do código de formatação (CSS);
- Existem outros recursos para páginas estáticas, como o ***Flash***;
- Note que vídeo, animações, ou outras mídias, não são suficientes para classificar uma página como dinâmica.

Páginas Dinâmicas

- O modelo de páginas estáticas foi útil nos primeiros momentos da Web, quando um grande volume de informação foi inserido;
- Atualmente, grande parte do uso da Web está voltado para aplicações e serviços:
  - Comércio eletrônico;
  - Pesquisa em catálogos de bibliotecas ou na própria Web;
  - Leitura e envio de e-mails;
  - Colaboração e redes sociais;
- Neste novo modelo, as páginas são construídas dinamicamente, com base em dados fornecidos pelos usuários.

Páginas Dinâmicas

- Geração de páginas dinâmicas:



- Pode ocorrer no lado do cliente (navegador);
- Ou no lado servidor.

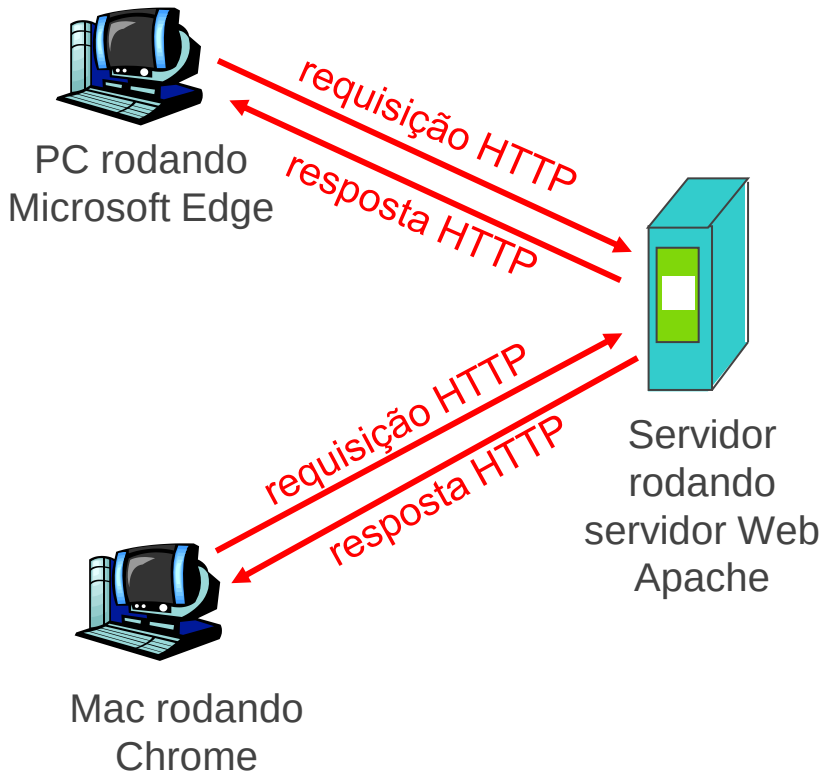
Páginas Estáticas X Dinâmicas

🔷 Característica	🌐 Página Estática	🚀 Página Dinâmica
Geração do Conteúdo	Conteúdo fixo, criado manualmente em HTML e CSS.	Gera conteúdo automaticamente com base em interações do usuário e banco de dados.
Interação	Nenhuma ou muito limitada. Apenas exibe informações.	Interativa, permite login, buscas, comentários e personalização.
Tecnologias Usadas	HTML, CSS simples, sem uso de scripts no servidor.	HTML, CSS, JavaScript, PHP, Python, Node.js, bancos de dados.
Exemplo	Site institucional, portfólio, blog sem CMS.	Redes sociais, e-commerce, sites com formulários e dashboards.
Armazenamento de Dados	Não há conexão com banco de dados; as informações estão no código-fonte.	Usa banco de dados (MySQL, MongoDB, PostgreSQL) para armazenar informações.
Velocidade de Carregamento	Mais rápida, pois os arquivos são simples e não exigem processamento.	Pode ser mais lenta, pois precisa processar informações antes de exibir o conteúdo.
Personalização	Não há personalização de conteúdo para o usuário.	Pode exibir conteúdo personalizado com base no usuário e suas preferências.
Exemplos de Tecnologias	Apenas HTML e CSS.	PHP, JavaScript, Python, Node.js, React, Angular, Vue.js.
Exemplo Visual	Uma página com texto fixo e imagens.	Um site de e-commerce que mostra produtos recomendados com base no histórico

HTTP: HyperText Transfer Protocol

- **protocolo da camada de aplicação da Web**

Parte do sucesso da Web se deve à simplicidade do HTTP, que facilitou seu desenvolvimento e implantação!



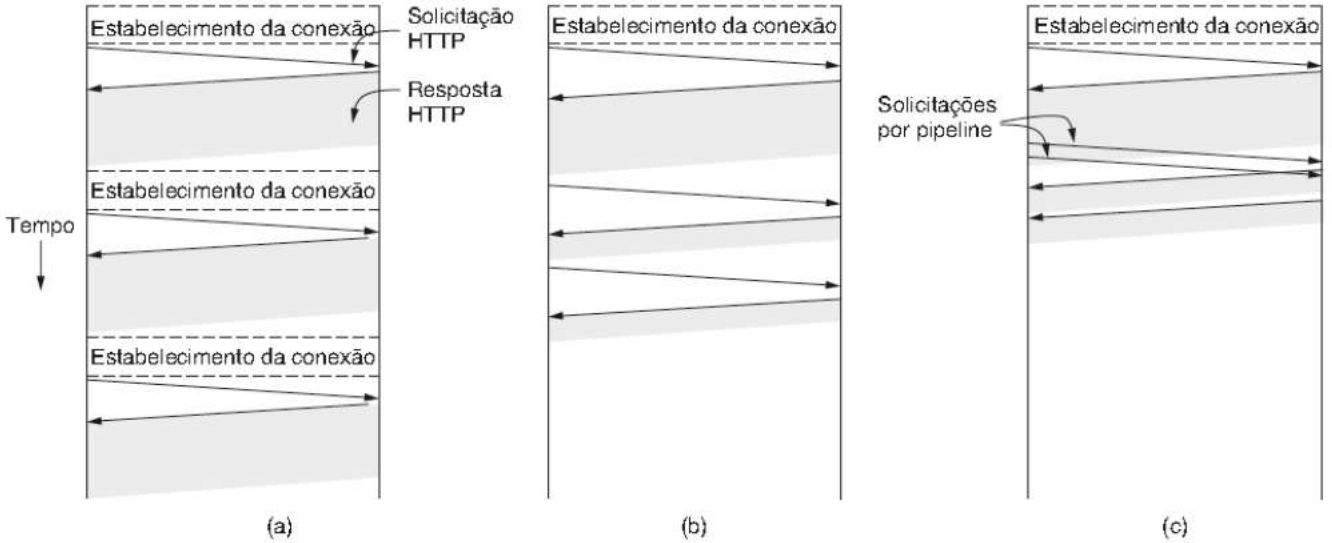
- Define a estrutura das mensagens trocadas
- **modelo cliente/servidor**
  - *cliente:* navegador que requisita, recebe, “exibe” objetos Web
  - *servidor:* servidor Web envia objetos em resposta a requisições

Visão geral do HTTP

- **Conexões**:
  - Utiliza protocolo TCP na porta 80;
  - Conexões são persistentes;
  - Os dados podem ser requisitados em *pipeline*;

HTTP é "sem estado"

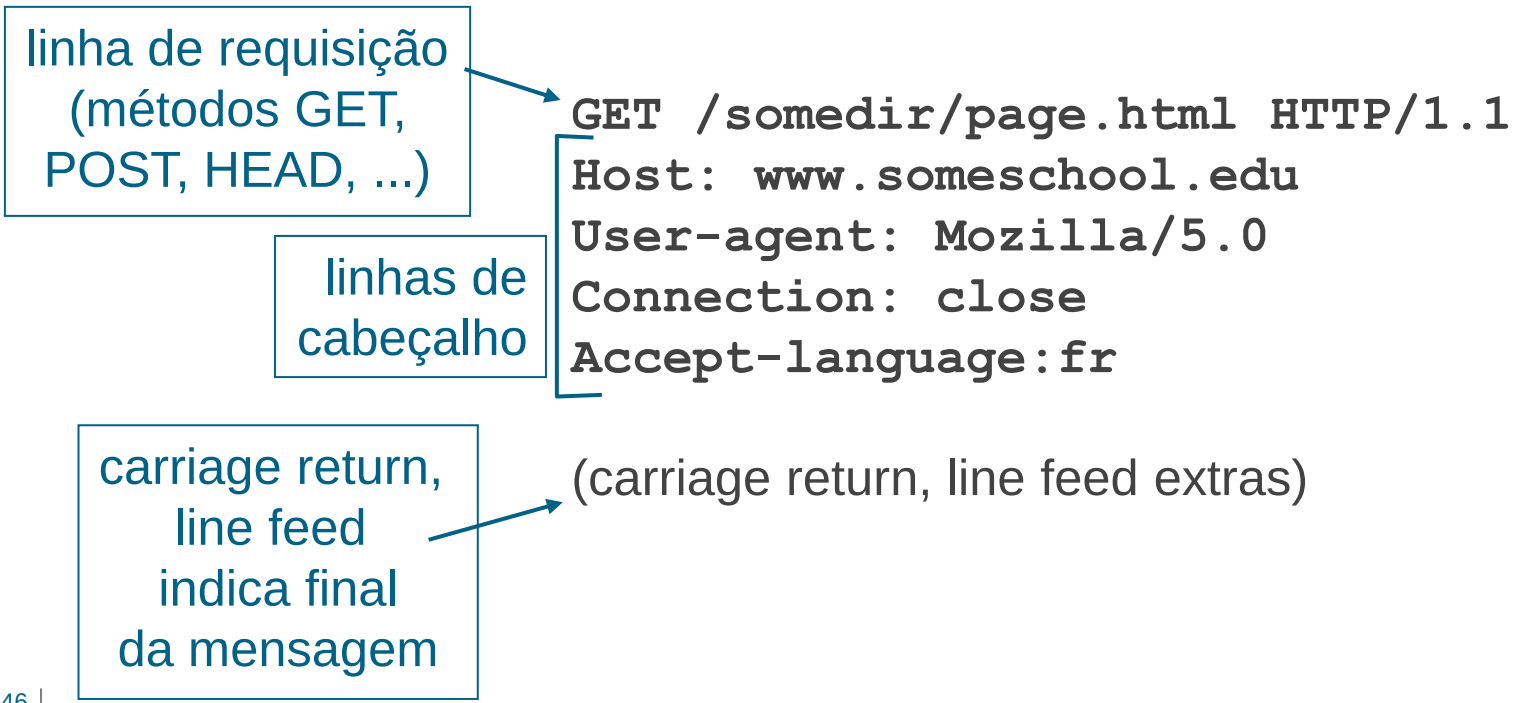
- servidor não guarda informações sobre requisições passadas do cliente.



(a) múltiplas conexões e solicitações sequenciais.
(b) Conexão persistente e solicitações sequenciais.
(c) Conexão persistente com solicitações em pipeline

Mensagem de requisição HTTP

- **dois tipos de mensagens HTTP: *requisição*, *resposta***
- **mensagem de requisição HTTP:**
  - ASCII (formato de texto legível)

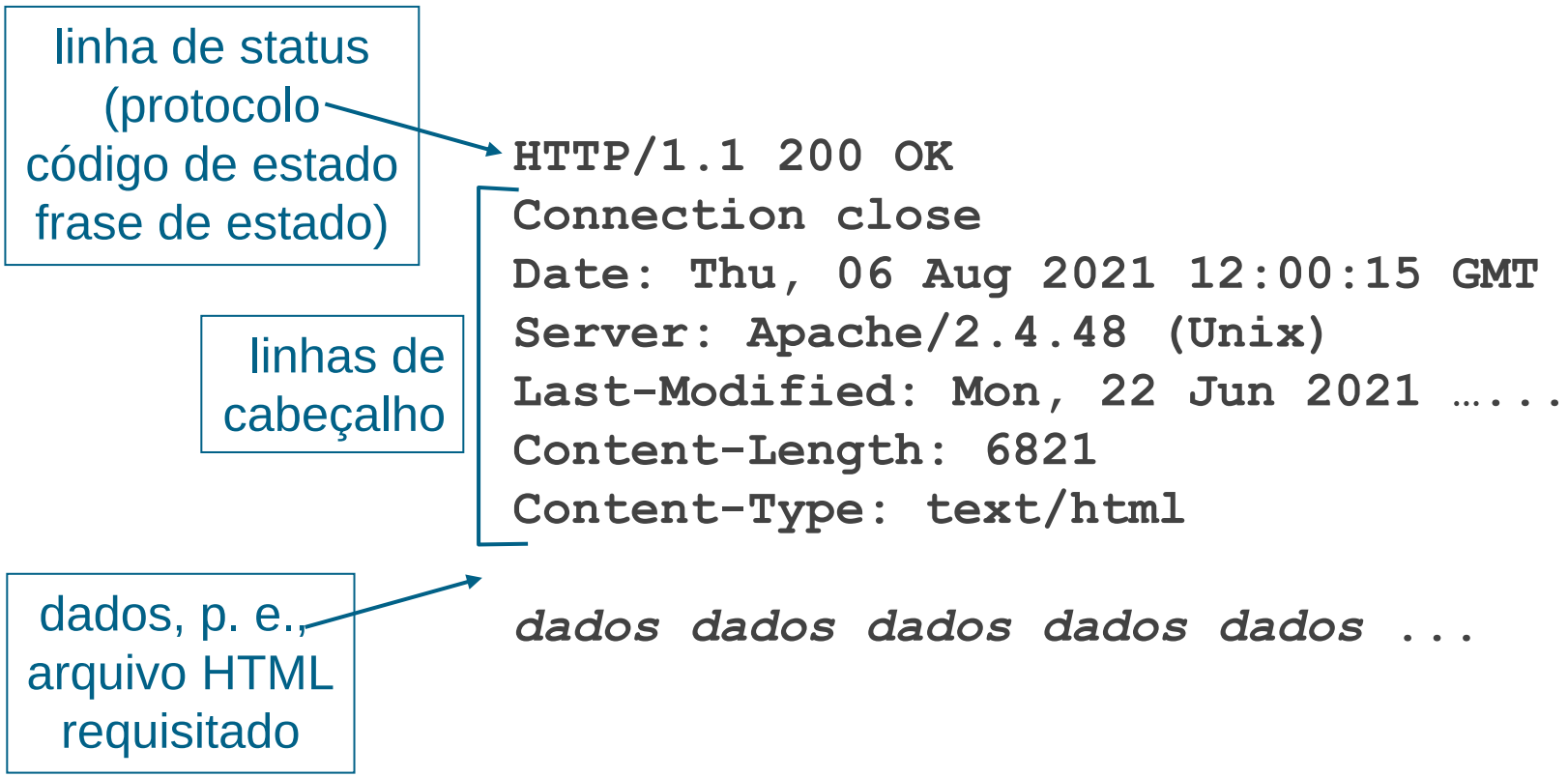


Mensagem de requisição HTTP

- **Métodos**:
  - O HTTP aceita operações chamadas ***métodos***;
  - Cada solicitação consiste de uma ou mais linhas de texto ASCII, sendo a primeira palavra da primeira linha o método solicitado:

Método	Descrição
GET	Lê uma página Web.
HEAD	Lê um cabeçalho de página Web. Pode ser usado para indexação ou testar a validade de um URL.
POST	Acrescenta algo a uma página Web. Usado para envio de dados de formulários para o servidor.
PUT	Armazena uma página Web. É o contrário de GET, possibilita criar uma coleção de páginas Web em um servidor remoto.
DELETE	Remove a página Web.
TRACE	Ecoa a solicitação recebida. Serve para depuração, envia o servidor a enviar de volta a solicitação para saber qual solicitação o servidor recebeu de fato.
CONNECT	Conecta através de um *proxy*.
OPTIONS	Consulta opções para uma página. Possibilita descobrir quais são os métodos e cabeçalhos que podem ser usados com uma página.

Mensagem de resposta HTTP



Códigos de estado da resposta HTTP

Códigos de estado da resposta HTTP

primeira linha da mensagem de resposta servidor->cliente

Código	Significado	Exemplos
1xx	Informação	100 = servidor concorda em tratar da solicitação do cliente.
2xx	Sucesso	200 = solicitação com sucesso; 204 = nenhum conteúdo presente.
3xx	Redirecionamento	301 = página movida; 304 = página em cache ainda válida.
4xx	Erro do cliente	403 = página proibida; **404 = página não localizada.**
5xx	Erro do servidor	500 = erro interno do servidor; 503 = tente novamente mais tarde.

HTTP vs HTTPS

	HTTP	HTTPS
URL	http://	https://
Security	Unsecure	Enhanced security
Port	PORT 80	PORT 443
OSI Layer	Application Layer	Transport Layer
TLS Certificates	No	Yes
Domain Validation	Not required	Domain validation (+ legal validation)
Encryption	No	Yes

Caching

- Normalmente os usuários retornam às páginas visitadas com frequência;
- Muitos recursos utilizados nunca mudam, ou mudam pouco, seria um desperdício capturar todos eles toda vez que uma página fosse novamente solicitada;
- O HTTP usa duas estratégias para enfrentar este problema: