BCC361 – Redes de Computadores
Universidade Federal de Ouro Preto
Departamento de Ciência da Computação

Prof. Reinaldo Silva Fortes
www.decom.ufop.br/reinaldo
2011/02

Camada	Nome
5	Aplicação
4	Transporte
3	Rede
2	Enlace
1	Física

Camada de Aplicação

Agenda

- Introdução;
- DNS (*Domain Name System*);
- Correio Eletrônico;
- A *World Wide Web* (WWW);
- *Streaming* de áudio e vídeo.

Introdução;
DNS (*Domain Name System*);
Correio Eletrônico;
A *World Wide Web* (WWW);
Streaming de áudio e vídeo.

INTRODUÇÃO

Tópicos

- A Camada de Aplicação;
- Arquitetura de aplicação de rede;
- Protocolos da camada de aplicação.

A Camada de Aplicação

- Aplicações são a razão de ser de uma rede de computadores;
- Se não fosse possível disponibilizar aplicações úteis, não haveria necessidade de projetar protocolos de rede para suportá-las;
- Assim, a camada de aplicação oferece serviços diretamente para o usuário através de todo o arcabouço estudado até o momento.

A Camada de Aplicação

- O cerne do desenvolvimento de aplicação de rede é escrever programas que rodem em sistemas finais diferentes e que se comuniquem entre si pela rede, exemplos:
  - Numa aplicação Web há dois programas distintos:
    - O *browser* que roda na máquina cliente;
    - E o *servidor Web*, que roda na máquina do servidor;
  - Em um compartilhamento de arquivos P2P:
    - Existem programas em cada máquina que participa do compartilhamento;
    - Estes programas podem ser idênticos ou semelhantes;
- A base das aplicações é a interação *cliente-servidor* e a comunicação entre pares.

Arquitetura de aplicação de rede

- A arquitetura da aplicação determina como a aplicação é organizada nos vários sistemas finais;
  - Arquitetura de aplicação é diferente da arquitetura de rede;
- Duas arquiteturas mais utilizadas em aplicações modernas:
  - Cliente-servidor;
  - *Peer-to-Peer* (P2P).

Arquitetura de aplicação de rede

- **Arquitetura Cliente-Servidor**:
  - Há um *hospedeiro* sempre em funcionamento: o **servidor**;
  - O servidor possui um endereço fixo e bem conhecido;
  - O servidor atende a requisições de muitos outros hospedeiros, os **clientes**;
  - Clientes não precisam estar sempre em funcionamento;
  - Clientes não se comunicam diretamente uns com os outros;
  - Exemplos de aplicações:
    - Web;
    - FTP;
    - Telnet;
    - E-mail.

Arquitetura de aplicação de rede

- **Arquitetura Peer-to-Peer (P2P)**:
  - A comunicação ocorre de forma direta entre ***pares*** de *hospedeiros*;
  - Estes pares não são de propriedade de provedores de serviços, mas são controlados por usuários finais;
  - Assim, não há garantias de que os pares estejam sempre em funcionamento;
  - Exemplos de aplicações:
    - BitTorrent;
    - eMule;
    - LimeWire;
    - Skype;
    - PPLive (aplicação de IPTV).

Protocolos da camada de aplicação

- Definem como os processos de uma aplicação trocam mensagens entre si, em particular:
  - Tipos de mensagens trocadas, por exemplo, de requisição e resposta;
  - A sintaxe dos vários tipos de mensagens, tais como os campos da mensagem e como os campos são delimitados;
  - A semântica dos campos, isto é, o significado da sua informação;
  - Regras para determinar quando e como um processo envia e responde mensagens.

Protocolos da camada de aplicação

- Alguns protocolos da camada de aplicação estão definidos em RFCs, ou seja, são de domínio público;
  - Exemplo: HTTP (protocolo da Web) – RFC 2616;
- Outros são proprietários e não estão disponíveis ao público;
  - Exemplo: A maioria dos sistemas de compartilhamento de arquivos P2P;
- Um protocolo da camada de aplicação é apenas uma parte da aplicação de rede;
  - Exemplo: O HTTP é o protocolo da Web, que por sua vez é composta de vários outros elementos: formato de documentos (HTML), browsers (*Firefox*, *Chrome*), servidores (*Apache*, *Microsoft*), etc...

DNS (*DOMAIN NAME SYSTEM*)

Tópicos

- Introdução;
- O ambiente de nomes;
- Registros de recursos;
- Servidores de nomes.

Introdução

- Imagine acessar os sites de seu interesse através do IP:
  - **UFOP**: 200.131.208.21 (www.ufop.br);
  - **Terra**: 200.154.56.80 (www.terra.com.br);
  - **UOL**: 200.147.67.142 (www.uol.com.br);
  - **Twitter**: 199.59.149.230 (www.twitter.com);
  - **Google**: 74.125.234.19 (www.google.com.br).
- **Problemas**:
  - Se lembrar de cada endereço IP;
  - Endereços IP podem mudar;
- **Solução**: Criar um serviço de nomes de alto nível.

Introdução

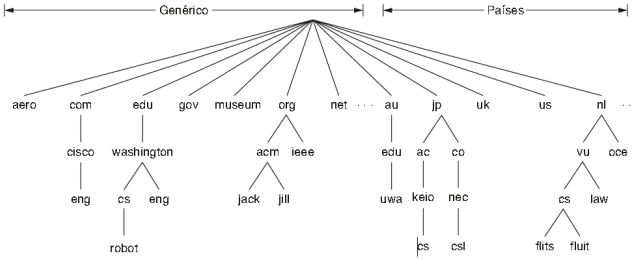
- Na ARPANET havia apenas um arquivo que continha mapeamentos Nome / IP (*hosts.txt*):
  - Para poucos *hosts* isto pode funcionar;
  - Mas para milhões de *hosts* conectados não;
- Assim foi criado o sistema de nomes e domínios, o **DNS**, ***Domain Name System***;
  - Definido nas RFCs 1034, 1035, 2181;
  - Detalhado em várias outras;

Introdução

- Funcionamento básico:
  - Uma aplicação faz uma chamada a um procedimento de biblioteca denominado **resolvedor**, passando como parâmetro o nome a ser "resolvido";
  - O resolvedor envia uma consulta contendo o nome para um servidor DNS local;
  - Este servidor retorna com o endereço IP ao resolvedor;
  - O resolvedor repassa o endereço retornado para a aplicação;
  - As mensagens de consulta e resposta são enviadas como mensagens UDP;
  - De posse do IP a aplicação pode estabelecer o tipo de comunicação de sua escolha.

O ambiente de nomes

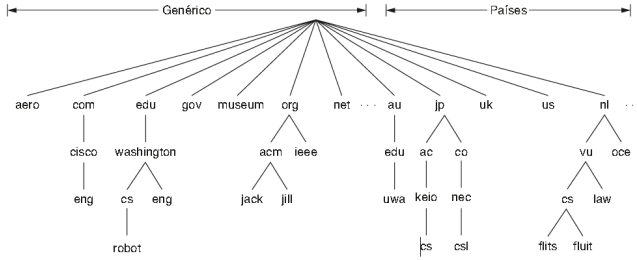
- Os nomes são definidos em uma estrutura hierárquica (1):



- Domínio de nível superior: genéricos e de países;
- Cada nível define um domínio independente e autônomo;
- Cada domínio controla seus próprios subdomínios;

O ambiente de nomes

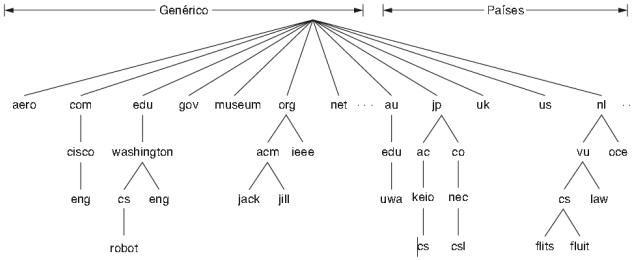
- Os nomes são definidos em uma estrutura hierárquica (2):



- A informação é distribuída pelos vários servidores da rede;
- Escalável (não há centralização de dados);

O ambiente de nomes

- Os nomes são definidos em uma estrutura hierárquica (3):



- O nome do domínio é ascendente e não haverá conflitos;
  - ***eng.cisco.com***, departamento de engenharia da cisco;
  - ***eng.washington.edu***, departamento de língua inglesa da universidade de Washington.

Registros de recursos

- Cada domínio pode ter um **registro de recursos** (um banco de dados DNS) associado a ele;
- Para um *host* comum o registro de recursos costuma ser composto apenas pelo seu endereço IP, mas existem muitos outros tipos;
- Quando um resolvedor repassa um nome de domínio a um servidor DNS, ele recebe na verdade os registros de recursos associados a ele;
- Portanto, a principal tarefa do servidor DNS é mapear nomes de domínios em registros de recursos.

Registros de recursos

- Um registro de recursos é uma tupla de cinco campos:
  1. Nome_dominio;
  2. Tempo_de_vida;
  3. Classe;
  4. Tipo;
  5. Valor.

Registros de recursos

1. **Nome_dominio**:
   - Informa a qual domínio o registro se aplica;
   - Normalmente um mesmo domínio possui vários registros;
   - Cada cópia do banco de dados possui informações de vários domínios;
   - A ordem dos registros não é significativa.

Registros de recursos

2. **Tempo_de_vida**:
   - Define um tempo para validade do registro;
   - Registros mais estáveis recebem tempos maiores;

3. **Classe**:
   - Para informações relacionadas à Internet recebe valor *IN*;
   - Existem outras classes, mas raramente são utilizadas na prática.

Registros de recursos

4. **Tipo**:
   - Informa o tipo do registro;
5. **Valor**:
   - Valor associado ao registro;
- Tipos mais significativos e seus valores:

Tipo	Significado	Valor
A	Endereço IPv4.	Inteiro de 32 bits.
AAAA	Endereço IPv6.	Inteiro de 128 bits.
MX	Troca de mensagens de correio eletrônico.	Prioridade, domínio disposto a aceitar correio eletrônico.
NS	Servido de nomes.	Nome para um servidor para este domínio.
CNAME	Nome canônico	Nome de domínio.
PTR	Ponteiro.	Nome alternativo de um endereço IP.
SRV	Serviço.	*Host* que oferece o serviço.

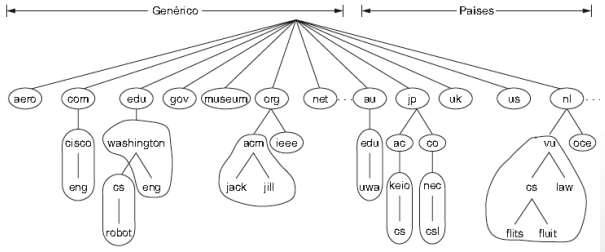
Registros de recursos

- Parte de uma possível base de dados DNS:

```
; Dados oficiais para cs.vu.nl
cs.vu.nl        86400   IN   SOA     star boss (9527,7200,7200,241920,86400)
cs.vu.nl        86400   IN   MX      1 zephyr
cs.vu.nl        86400   IN   MX      2 top
cs.vu.nl        86400   IN   NS      star

star            86400   IN   A       130.37.56.205
zephyr          86400   IN   A       130.37.20.10
top             86400   IN   A       130.37.20.11
www             86400   IN   CNAME   star.cs.vu.nl
ftp             86400   IN   CNAME   zephyr.cs.vu.nl

flits           86400   IN   A       130.37.16.112
flits           86400   IN   A       192.31.231.165
flits           86400   IN   MX      1 flits
flits           86400   IN   MX      2 zephyr
flits           86400   IN   MX      3 top

rowboat                 IN   A       130.37.56.201
                        IN   MX      1 rowboat
                        IN   MX      2 zephyr

little-sister           IN   A       130.37.62.23

laserjet                IN   A       192.31.231.216
```


Servidores de nomes

- Por que usar servidores? Apenas um não resolveria?
  - Na teoria sim, mas na prática seria impossível;
  - Problemas de sobrecarga e alta dependência inviabilizam esta solução;
  - Assim, o espaço de nomes do DNS é dividido em **zonas** não sobrepostas, exemplo:



Servidores de nomes

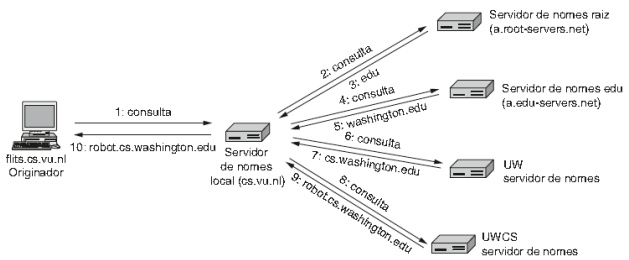
- Cada zona está associada a um ou mais servidores de nomes;
- Estes servidores mantêm o banco de dados da zona;
- Normalmente uma zona terá um **servidor de nomes primário**, que recebe a informação de um arquivo em seu disco, e **servidores de nomes secundários**, que recebem informações do servidor primário;
- Para melhoria de confiabilidade, alguns servidores de nomes podem estar localizados fora da zona.

Servidores de nomes

- O processo de pesquisa de um nome e localização de um endereço é chamado **resolução de nomes**;
- Um **registro oficial** é aquele que vem da autoridade oficial que controla o registro, portanto, está sempre correto;
- Um **registro de cache** é aquele retornado por um servidor que armazenou temporariamente a informação por questão de performance, portanto, pode estar desatualizado;

Servidores de nomes

- Exemplo de resolução de nome:



Servidores de nomes

- Dois mecanismos de consulta:
  - **Consulta recursiva**: o servidor de nomes local retorna a resposta final ao originador, fazendo quantas chamadas forem necessárias a outros servidores de nome;
  - **Consulta iterativa**: o servidor de nomes apenas retorna uma resposta parcial, com a informação que lhe compete, não realiza chamadas a outros servidores para completar a resposta.

Servidores de nomes

- Mecanismo de **caching**:
  - Todas as respostas, incluindo as parciais, são armazenadas em cache para atendimento rápido a novas solicitações;
  - Caso haja solicitações para um *host* diferente de um mesmo domínio, o caminho é *encurtado* fazendo uma solicitação direta ao servidor de nomes oficial, sem passar por servidores de hierarquias mais altas;
  - O cache melhora a performance, mas deve haver cuidado com as possíveis alterações de informações, por isso cada registro de recurso possui um campo TTL (tempo de vida).

CORREIO ELETRÔNICO

Tópicos

- Introdução;
- Arquitetura e serviços;
- Agente do usuário;
- Formato de mensagens;
- Transferência de mensagem;
- Entrega final.

Introdução

- O correio eletrônico, ou e-mail, já existe há muito tempo, mais de duas décadas;
- Muito mais rápido e barato que o sistema de correios convencional;
- Uma das aplicações mais populares desde os primeiros dias da Internet;
- Infelizmente, a maior parte dos e-mails é lixo, ou **spam**;

Introdução

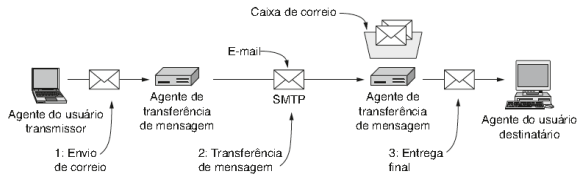
- Existem várias convenções e estilos no uso do e-mail:
  - **Jargões**: AP (a propósito), RTCP (rodando no chão de tanto rir), EMHO (em minha humilde opinião);
  - ***Smileys***: :-), :-(;
  - ***Emotions***;
  - Etc.;
- Os protocolos também evoluíram com o tempo;
  - Dos e-mail com texto ASCII puro a mensagens com formato HTML, som, imagem, etc.;
- Também evoluíram os meios de acesso aos e-mails;
  - Aplicações de leitura de e-mail (*Outlook*, *Mozilla Thunderbird*, etc.) e Webmails;

Arquitetura e serviços

- Principais serviços:
  - Composição;
  - Transferência;
  - Relatórios;
  - Exibição;
- Subsistemas divididos em:
  - Agentes de usuário;
  - Agentes de transferência;

Arquitetura e serviços

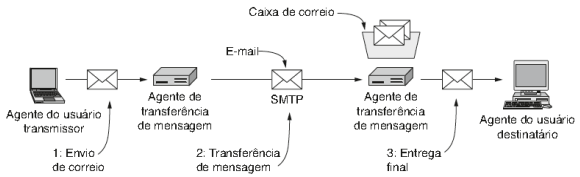
- **Arquitetura do sistema de e-mails** (1):



  - **Agentes do usuário transmissor**: permitem que as pessoas leiam e enviem mensagens;
  - **Agentes de transferência de mensagens**: deslocam as mensagens da origem ao destino. Também denominados **servidores de correio**;

Arquitetura e serviços

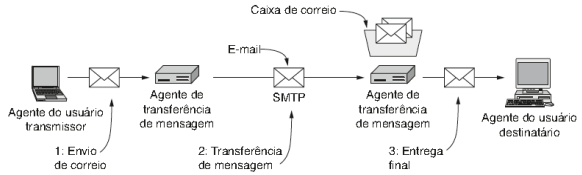
- **Arquitetura do sistema de e-mails** (2):



  - O ato de enviar uma mensagem é denominado **submissão de e-mail**;
  - A etapa de transferência da mensagem é feita utilizando o protocolo **SMTP**;

Arquitetura e serviços

- **Arquitetura do sistema de e-mails** (3):



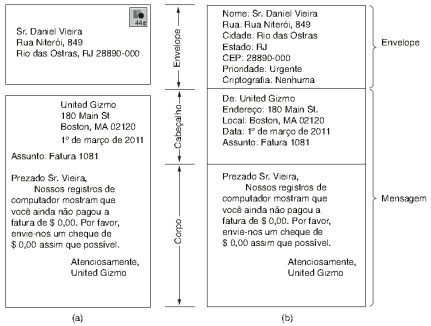
  - As **Caixas de Correio** armazenam os e-mails recebidos para um usuário:
    - São mantidas pelos servidores de correio;
    - Os agentes de usuário simplesmente apresentam aos usuários uma visão das caixas de correio, eventualmente permitindo o armazenamento local e a eliminação das mensagens no servidor.

Arquitetura e serviços

- O e-mail é enviado entre os agentes de transferência seguindo um formato padrão:
  - Formato original é definido na RFC 822;
  - Revisado na RFC 5322, e estendido com suporte para conteúdo multimídia e texto internacional (MIME);
- A ideia principal no formato da mensagem é a distinção entre o envelope da mensagem e seu conteúdo (**corpo**);
  - O envelope contém toda a informação necessária para transportar a mensagem;
  - O conteúdo é inteiramente direcionado para o destinatário;

Arquitetura e serviços

- Envelopes e mensagens:



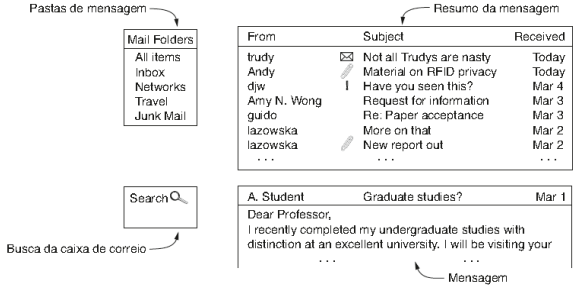
(a) Correspondência em papel.
(b) Correspondência eletrônica.

Agente do usuário

- Um agente de usuário é um programa (às vezes denominado **leitor de e-mail**) que aceita uma série de comandos para compor, receber e responder mensagens, além de manipular caixas de correio;
- Existem muitos agentes de usuário:
  - *Gmail* da *Google*;
  - *Microsoft Outlook*;
  - *Mozilla Thunderbird*;
  - *Apple Mail*;
  - Etc...;

Agente do usuário

- Elementos típicos de interface do agente do usuário:



Formato de mensagens

- As mensagens precisam ser colocadas em um formato padrão para serem manipuladas pelos agentes de transferência;
- Inicialmente as mensagens continham apenas texto em formato ASCII (RFC 5322 / 822);
- Depois foram aprimoradas para possibilitar conteúdo multimídia e múltiplos idiomas (MIME);

Formato de mensagens

- **RFC 5322 – Formato de mensagem da Internet** (1):
  - Mensagens consistem de um envelope básico, alguns campos de cabeçalho, uma linha em branco e o corpo da mensagem;
  - Principais campos do cabeçalho, relacionados ao transporte:

Campo	Significado
To:	Os endereços de correio eletrônico dos destinatários principais.
Cc:	Os endereços de correio eletrônico dos destinatários secundários.
Cco:	Os endereços de correio eletrônico dos destinatários ocultos.
From:	As pessoas que criaram a mensagem.
Sender:	Endereço de e-mail do remetente.
Received:	Linha incluída por cada agente de transferência ao longo da rota (identidade do agente, data, hora de recebimento, entre outras).
Return-Path:	Pode ser incluída para identificar um caminho de volta ao remetente.

Formato de mensagens

- **RFC 5322 – Formato de mensagem da Internet** (2):
  - Adicionalmente outros campos de cabeçalho podem ser utilizados pelos agentes ou destinatários:

Campo	Significado
Date:	Data e hora em que a mensagem foi enviada.
Replay-To:	Endereço de e-mail para onde as respostas devem ser enviadas.
Message-Id:	Número exclusivo que será usado para fazer referência à mensagem posteriormente.
Keywords:	Palavras-chave do usuário.
Subject:	Pequeno texto de resumo da mensagem apresentados em apenas uma linha.

  - Usuários podem criar novos cabeçalhos para seu uso;
  - Eles devem começar com "X-".

Formato de mensagens

- **MIME – *Mutipurpose Internet Mail Extensions*** (1):
  - Nos anos 90 o uso mundial da Internet e a demanda por conteúdo mais rico através do sistema de e-mail mostrou que o protocolo anterior não era mais adequado;
  - Para resolver esta questão foi desenvolvido o MIME, que além de ser utilizado para mensagens enviadas pela Internet também é utilizado para definir conteúdo para outras aplicações, como a navegação Web;
  - Descrito nas RFCs 2045 – 2047, 4288, 4289 e 2049;
  - A ideia é continuar a usar o formato da RFC 822, mas incluir uma estrutura para o corpo da mensagem e definir regras para mensagens que não utilizam o ASCII;

Formato de mensagens

- **MIME – *Mutipurpose Internet Mail Extensions*** (2):
  - Permite:
    - Mensagens em idiomas com acentos;
    - Mensagens em alfabetos não latinos;
    - Mensagens em idiomas sem alfabetos;
    - Mensagens que não contêm textos;
  - Cabeçalhos de mensagens acrescentados:

Campo	Significado
MIME-Version:	Identifica a versão do MIME.
Content-Description:	String inteligível que identifica o conteúdo da mensagem.
Content-Id:	Identificador exclusivo.
Content-Transfer-Encoding:	Como o corpo da mensagem é codificado para transmissão.
Content-Type:	Tipo e formato de conteúdo.

Formato de mensagens

- **MIME – *Mutipurpose Internet Mail Extensions*** (3):
  - Tipos de conteúdo MIME e exemplos de subtipos:

Tipo	Subtipos de exemplo	Descrição
text	plain, html, xml, css	Texto em vários formatos.
image	gif, jpeg, tiff	Imagens.
audio	basic, mpeg, mp4	Sons.
video	mpeg, mp4, quicktime	Filmes.
model	vrml	Modelo 3D.
application	octect-stream, pdf, javascript, zip	Dados produzidos por aplicações.
message	http, rfc822	Mensagem encapsulada (exemplo: encaminhamento de mensagem).
multipart	mixed, alternative, parallel, digest	Combinação de vários tipos.

Transferência de mensagem

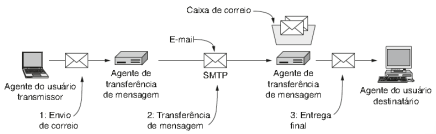
- A transferência de mensagens é feita utilizando-se do protocolo **SMTP** (***Simple Mail Transfer Protocol***);
- A transferência no SMTP é feita através de uma conexão TCP na porta 25;
- Trata-se de um protocolo simples, que usa o ASCII;
- Não são necessários *checksums*, porque o TCP fornece fluxo de bytes confiável.

Transferência de mensagem

- O SMTP básico funciona bem, mas possui algumas limitações:
  1. Não inclui autenticação:
     - Aceita qualquer valor para FROM por exemplo (ótimo para *spammers*);
  2. Transfere mensagens ASCII, e não dados binários:
     - Daí a necessidade do MIME;
     - Uso ineficaz da largura de banda (um problema para grandes mensagens);
  3. Envia mensagens às claras:
     - Não usa criptografia (ruim para provacidade);
- Para solucionar alguns dos problemas do SMTP foi criado o **ESMTP** (***Extended SMTP***).

Transferência de mensagem

- Existem dois usos do SMTP:
  1. **Envio de correio**:
     - Etapa 1 da arquitetura de e-mail;
     - Meio pelo qual um agente do usuário envia uma mensagem para o sistema de e-mail de entrega;
  2. Transferência entre agentes de transferência de mensagens:
     - Etapa 2 da arquitetura de e-mail;
     - Entrega do e-mail do agente de transferência de mensagem emissor para o agente receptor;
     - A entrega final é feita com protocolos diferentes.



Entrega final

- A última etapa na arquitetura de e-mail trata-se da entrega do e-mail ao agente do usuário;
- Os usuários desejam acessar seus e-mails remotamente, onde e quando isso for necessário:
  - No trabalho, em casa, no quarto de um hotel, *LAN House*, etc...;
  - De um PC, notebook, celular, etc...;
  - Também podem desejar trabalhar *off-line*, sincronizando com o servidor quando estiver novamente *on-line*;
- O SMTP não foi projetado para estes objetivos, para isso são utilizados outros protocolos.

Entrega final

- Um dos principais protocolos usados para este fim é o **IMAP** (***Internet Message Access Protocol***);
- O cliente IMAP se conecta ao servidor IMAP e inicia a execução de uma série de comandos:
  - Organização das mensagens em pastas;
  - Listagem de pastas e mensagens;
  - Buscar mensagens ou parte delas;
- O IMAP é uma melhoria de um protocolo mais antigo, o **POP3** (***Post Office Protocol, version 3***), especificado na RFC 1939:
  - POP3 é mais simples e fornece menos recursos e segurança;
  - Normalmente o e-mail é baixado no computador do agente de usuário e deletado do servidor.

Entrega final

- Existem também protocolos fechados, como o ***Microsoft Exchange***;
- Outra alternativa para a entrega final é o uso de ***Webmail***:
  - Trata-se de um software fornecido com um serviço de uso da Web;
  - Usuários podem utilizar qualquer navegador que desejarem, em qualquer máquina conectada na Internet, para acessar e enviar mensagens;
  - Exemplos: *Google Gmail*, *Microsoft Hotmail* e *Yahoo! Mail*;
  - Normalmente estes serviços fornecem opções de servidores IMAP e POP3 para possibilitar aos usuários o acesso aos e-mails a partir de outras aplicações.

A *WORLD WIDE WEB* (WWW)

Tópicos

- Introdução;
- Arquitetura;
- Páginas estáticas;
- Páginas dinâmicas;
- Protocolo de transferência.

Introdução

- A ***World Wide Web***, ou **WWW**, ou **Web**, é uma estrutura arquitetônica que permite o acesso a documentos vinculados espalhados por milhões de máquinas na Internet;
- Sua enorme popularidade se deve, principalmente, a dois fatores:
  - Interface gráfica colorida e de fácil utilização para principiantes;
  - Uma imensa variedade de informações sobre quase todos os assuntos imagináveis;
- Teve seu início em 1989 no CERN (*European Organization for Nuclear Research*), para ajudar grandes equipes de membros espalhados por vários países a colaborar compartilhando relatórios, plantas, desenhos, fotos e outros documentos;

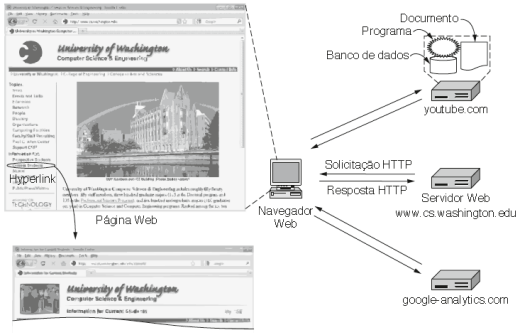
Introdução

- A proposta para uma teia de documentos interligados veio do físico **Tim Berners-Lee**;
- O primeiro protótipo foi apresentado em 1991, chamando a atenção de muitos pesquisadores;
- Em 1993, **Marc Andressen**, da Universidade de Illinois lançou um navegador chamado ***Mosaic***;
- Um ano depois ele formava sua empresa, a ***Netscape Communications Corp.***, que "lutou" durante alguns anos contra a ***Microsoft*** e seu navegador, o ***Internet Explorer***;

Introdução

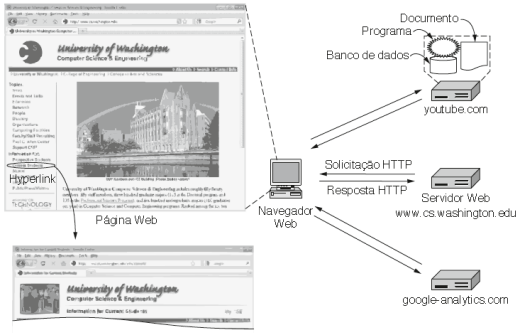
- No decorrer das décadas de 1990 e 2000, sites e páginas Web cresceram exponencialmente, atingindo milhões de sites e bilhões de páginas;
- Algumas delas se tornaram tremendamente populares:
  - ***Amazon***, 1994, mercado de US$ 50 bilhões;
  - ***eBay***, 1995, US$ 30 bilhões;
  - **Google**, 1998, US$ 150 bilhões;
  - **Facebook**, 2004, US$ 15 bilhões;
- Em 1994, o CERN e o MIT criaram o **W3C** (***World Wide Web Consortium***), www.w3.org ou www.w3c.br, uma organização responsável por organizar e padronizar o desenvolvimento Web.

Arquitetura (1)



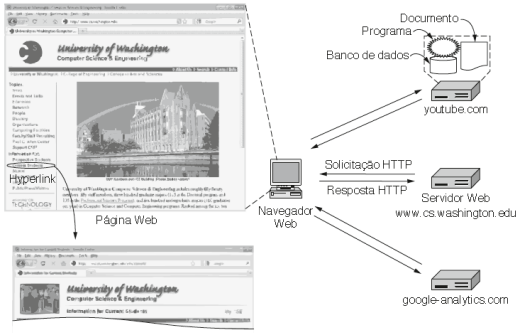
- Usuário acessando a página da Universidade de Washington através de um *browser* (**navegador**);
- Esta página contém elementos de mídias variadas e links para outras páginas;

Arquitetura (2)



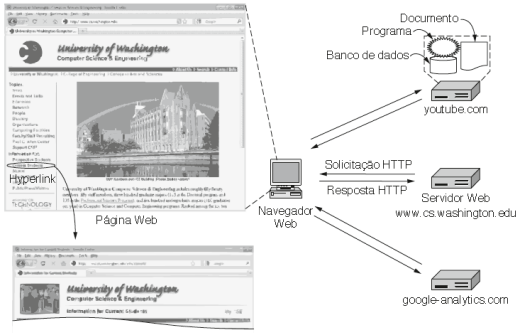
- Quando um link é clicado o navegador busca a nova página e faz a sua exibição;
- Seu conteúdo pode estar na mesma máquina da página anterior, mas também pode não estar, isso é transparente para o usuário;

Arquitetura (3)



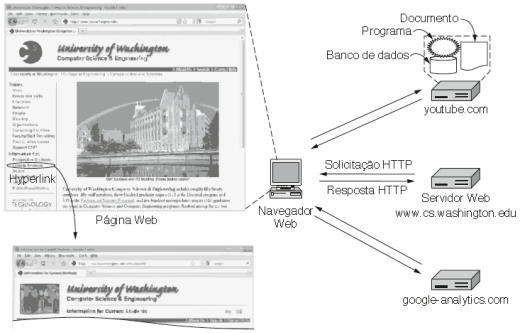
- Cada página é obtida enviando solicitações a um ou mais servidores, que respondem com o conteúdo da página;
- O protocolo usado é simples, implementado sobre o TCP, é denominado **HTTP** (***HyperText Transfer Protocol***);

Arquitetura (4)



- As páginas pode ser **estáticas**, ou seja, exibem sempre o mesmo conteúdo;
- Ou podem ser **dinâmicas**, ou seja, o conteúdo é formado sob demanda, gerada por um programa;

Arquitetura (5)



- Na figura, o navegador contata três servidores para montar as páginas, ***cs.washington.edu*** fornece o conteúdo principal, ***youtube.com*** um conteúdo de vídeo, e google-anaytics.com não fornece dados visíveis ao usuário, mas é usado para estatísticas;

Arquitetura

- Atualmente o site da Universidade de Washington é assim:

Arquitetura

- **O lado cliente** (1):
  - Para identificar uma página é utilizada a **URL** (***Uniform Resource Locator***), que é definido por três partes:
    - O protocolo (também conhecido como **esquema**);
    - O nome DNS da máquina onde está localizada;
    - O caminho, que especifica exclusivamente onde está a página;

Arquitetura

- **O lado cliente** (2):
  - 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;

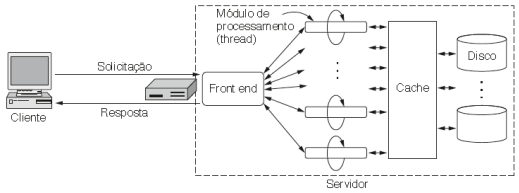
Arquitetura

- **O lado cliente** (3):
  - Existem diferentes protocolos que podem ser usados em uma URL:

Nome	Usado para	Exemplo
http	Hipertexto (HTML).	http://www.decom.ufop.br/reinaldo
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

- **O 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;
    - Encerrar a conexão.

Arquitetura

- **Cookies** (1):
  - Em algumas aplicações é necessário identificar certas informações do usuário para personalizar conteúdo;
    - **Exemplo**: produtos em uma cesta de compras de um site comercial;
  - Este problema é resolvido com um mecanismo chamado **cookie**, que trata-se apenas de uma string pequena contendo algumas informações;
  - Quando o cliente solicita uma página, o servidor pode fornecer informações adicionais, na forma de um cookie junto com a página retornada;

Arquitetura

- **Cookies** (2):
  - O cookie é armazenado no cliente para ser utilizado em novas requisições ao mesmo servidor;
  - Um cookie pode conter até cinco campos, alguns exemplos de cookies:

Domínio	Caminho	Conteúdo	Expira	Seguro
toms-casino.com	/	CustomerI=D297793521	15-0-10-17:00	Sim
jiils-store.com	/	Cart=1-00501;1-07031	11-1-11 14:22	Não
aportal.com	/	Prefs=Stk;CSCO	31-12-20 23:50	Não
sneaky.com	/	UserId=4627239101	31-12-19 23:59	Não

  - Cookies estão envolvidos com algumas questões de privacidade e segurança, muitos gostam de desativar o seu uso no navegador.

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 a 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;

Páginas estáticas

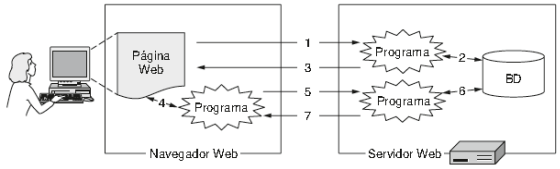
- Para resolver os problemas de formatação foi criado o conceito de folha de estilo, o **CSS** (***Cascade Style Sheets***), que faz a separação do código estrutural (HTML) do código de formatação (CSS);
- Existem outros recursos para páginas estáticas, como o ***Flash*** por exemplo;
- Note que vídeo, 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 dinâmicas

- **APIs para páginas dinâmicas do lado SERVIDOR** (1):
  - **CGI** (***Common Gateway Interface***):
    - Definido na RFC 3875;
    - Oferece uma interface para permitir que os servidores "falem" com programas de *back-end* e *scripts* que aceitam entrada (formulários por exemplo) e gerem páginas em resposta;
    - Os programas pode ser escritos em variadas linguagens: *Python*, *Ruby*, *Perl*, etc...;
  - **PHP** (***Hypertext Preprocessor***):
    - A técnica consiste em inserir código script dentro das páginas HTML;
    - Os scripts serão executados no próprio servidor, originando uma página HTML como resultado;
    - PHP é uma linguagem de programação poderosa para a interface entre a Web e bancos de dados do servidor;
    - Ela contém variáveis, *strings*, *arrays* e a maior parte das estruturas de controle encontradas no C;

Páginas dinâmicas

- **APIs para páginas dinâmicas do lado SERVIDOR** (2):
  - **JSP** (***Java Server Pages***):
    - Semelhante ao PHP, mas utiliza linguagem Java;
  - **ASP.NET** (***Active Server Pages .NET***):
    - Versão da Microsoft para PHP e JSP;

Páginas dinâmicas

- **Páginas dinâmicas do lado CLIENTE** (1):
  - Para responder a movimentos do mouse ou interagir diretamente com os usuários é necessário incorporar scripts em páginas HTML executadas na máquina cliente;
  - A partir do HTML 4.0 esses scripts são permitidos, através do uso da *tag <script>*;
  - A linguagem mais popular é a ***JavaScript*** (nenhuma relação com a linguagem Java, além do nome);

Páginas dinâmicas

- **Páginas dinâmicas do lado CLIENTE** (2):
  - Uma alternativa ao *JavaScript*, na plataforma *Windows* é o ***VBScript***, que é baseado no *Visual Basic*;
  - Outro método popular é os ***applets***, que são pequenos programas escritos em Java (incorporados a partir da *tag <applet>*);
  - Uma solução da *Microsoft* semelhante aos *applets* é denominada de ***controles ActiveX***, que são programas compilados na linguagem de máquina X86 e executados no *hardware* bruto.

Páginas dinâmicas

- ***AJAX*** (***Asynchronous JavaScript and XML***) (1):
  - Trata-se de um conjunto de tecnologias que trabalham juntas para criar aplicações Web tão interativas e poderosas quanto aplicações de desktop tradicionais;
  - O AJAX possibilita que páginas Web sejam atualizadas assincronamente através da troca de pequenas quantidades de dados com seus servidores;
    - Isto possibilita que apenas partes da página sejam atualizadas, sem a necessidade de recarregar todo o conteúdo da página;

Páginas dinâmicas

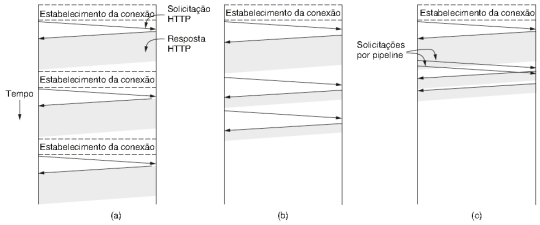
- ***AJAX*** (***Asynchronous JavaScript and XML***) (2):
  - Tecnologias envolvidas:
    1. **HTML** e **CSS**: para apresentar as informações como páginas;
    2. **DOM** (***Document Object Model***): árvore que reflete a estrutura dos elementos HTML, utilizado para alterar partes da página quando elas são exibidas;
    3. **XML** (***eXtensible Markup Language***): para permitir a troca de dados entre cliente e servidor;
    4. Um modo assíncrono para os programas enviarem e receberem dados XML;
    5. **JavaScript**: como uma linguagem para juntar toda esta funcionalidade;
- Exemplos de páginas que usam AJAX: *Gmail*, *Maps* e *Docs* da *Google*, *Youtube* e *Facebook*.

Protocolo de transferência

- O protocolo utilizado para transportar toda a informação entre os servidores Web e os clientes Web é o **HTTP** (***HyperText Transfer Protocol***), especificado na RFC 2116;
- HTTP é um protocolo simples:
  - Funciona no estilo *solicitação-resposta*, rodando sobre o TCP;
  - Especifica quais mensagens os clientes podem enviar para os servidores e quais respostas recebem de volta;
  - Assim como no SMTP, os cabeçalhos são dados em ASCII e o conteúdo é dado em formato tipo MIME;
- Parte do sucesso da Web é creditado à simplicidade do HTTP, que facilitou o seu desenvolvimento e implantação.

Protocolo de transferência

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



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

Protocolo de transferência

- **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.

Protocolo de transferência

- **Códigos de erro**:
  - Toda solicitação obtém uma resposta que possui uma linha de *status*, com um código de três dígitos informando se a solicitação foi atendida ou qual foi o erro:

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.

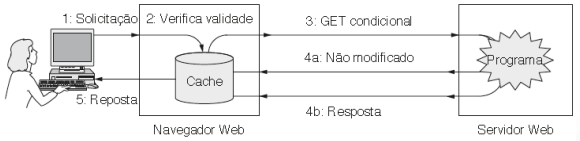
Protocolo de transferência

- **Cabeçalhos de mensagens**:
  - Toda solicitação pode ser seguida de linhas adicionais contendo mais informações, chamadas de ***cabeçalhos de solicitação***;
  - De forma análoga, as respostas podem ser seguidas de linhas denominadas ***cabeçalhos de resposta***;
  - Alguns possíveis cabeçalhos (a lista é extensa):

Cabeçalho	Tipo	Conteúdo
User-Agent	Solicitação	Informações sobre o navegador e sua plataforma.
Accept	Solicitação	O tipo de páginas que o cliente pode manipular.
Accept-Charset	Solicitação	Os conjuntos de caracteres aceitáveis para o cliente.
Accept-Encoding	Solicitação	As codificações de páginas que o cliente pode manipular.
Cookie	Solicitação	Cookie previamente definido, enviado de volta ao servidor.
Set-Cookie	Resposta	Cookie para ser armazenado no cliente.
Expires	Resposta	Data e hora de quando a página deixa de ser válida.
Last-Modified	Resposta	Data e hora da última modificação da página.
Cache-Control	Ambos	Diretivas para o modo de tratar caches.

Protocolo de transferência

- **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:



Introdução;
DNS (*Domain Name System*);
Correio Eletrônico;
A World Wide Web (WWW);
***Streaming* de áudio e vídeo**.

STREAMING DE ÁUDIO E VÍDEO

Streaming de áudio e vídeo

- Para os interessados:
  - Material disponibilizado no xerox.

Fim!

REFERÊNCIAS:

- **A.S. TANENBAUM, *Redes de Computadores*, Prentice Hall, 5a. edição, 2011;**
- KUROSE e ROSS, *Redes de Computadores e a Internet – uma abordagem top-down*, 5a. Edição, 2010.
- Materiais didáticos dos professores:
  - **Rande A. Moreira**, UFOP / 2011-01
    Disponível em: http://randearievilo.com.br/redes/ (acesso em 17/08/2011);