

# CSI477 – Sistemas para a WEB I



**UFOP**

Prof. Fernando Bernardes de Oliveira  
<https://sites.google.com/site/fboliveiraufop>

Universidade Federal de Ouro Preto  
Instituto de Ciências Exatas e Aplicadas – ICEA  
Departamento de Computação e Sistemas – DECSI

João Monlevade-MG  
1º semestre de 2017

# História da Internet

## ■ ARPANET:

- Projeto – *Department of Defense Advanced Research Projects Agency* – DARPA.
- Desenvolvimento de uma rede que interligasse os computadores do governo americano, de diferentes fabricantes e utilizando diferentes sistemas operacionais.
- Rede → **descentralizada**;
- Ataque militar → continuaria a funcionar normalmente – **rotas alternativas**.

# História da Internet

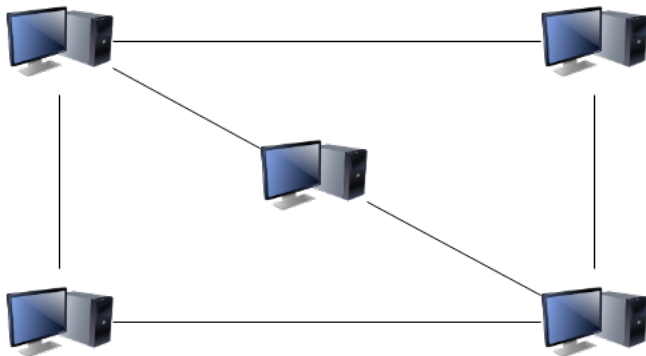
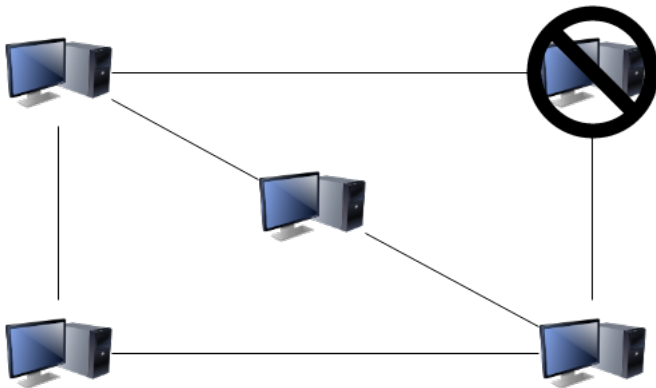


Figura 1: Rede ativa – Imagem construída em <https://www.draw.io/>

# História da Internet



**Figura 2:** Rede parcialmente indisponível – Imagem construída em <https://www.draw.io/>

# História da Internet

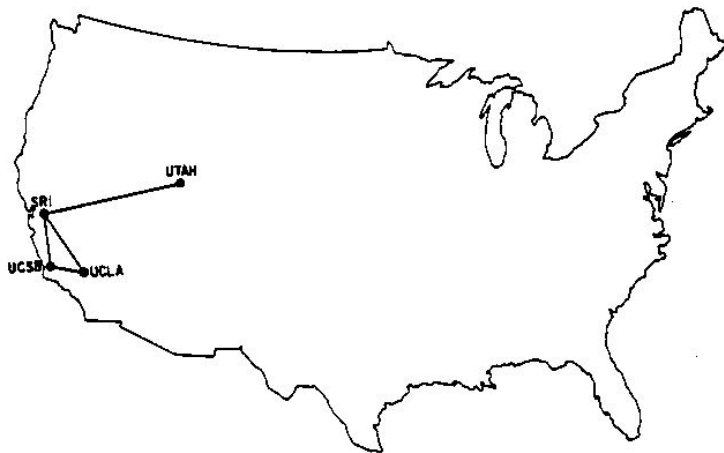


Figura 3: ARPANET: Dezembro, 1969.

# História da Internet

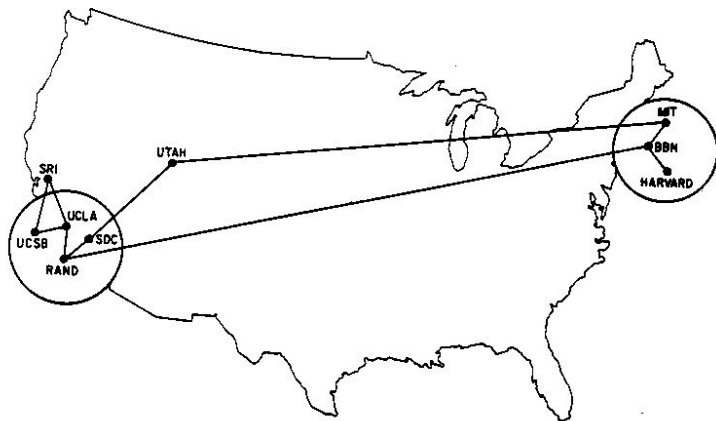


Figura 4: ARPANET: Junho, 1970.

# História da Internet

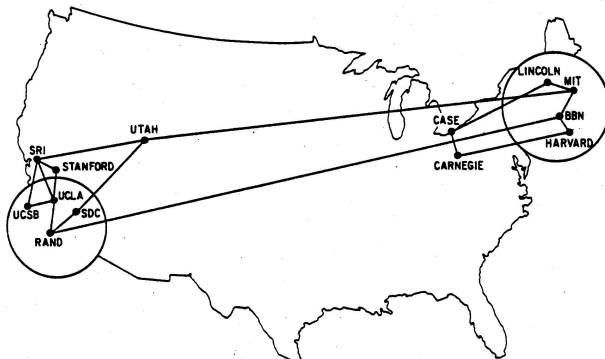


Figura 5: ARPANET: Dezembro, 1970.

# História da Internet

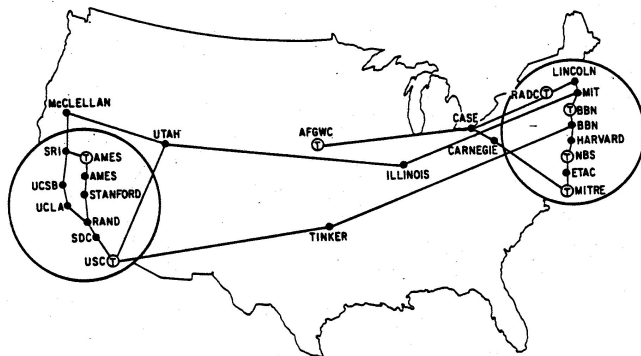


Figura 6: ARPANET: Março, 1972.



# História da Internet

- Mapas ARPANET → retirados de:  
(<http://som.csudh.edu/fac/lpress/history/arpamaps/>)
- Pesquisadores → “compartilhamento” de computadores uns dos outros.
- Comunicação rápida e eficiente → **e-mail**.
- Operação da rede → **troca de pacotes** (*packet switching*).
- Protocolo de transmissão → **TCP** (*Transmission Control Protocol*);
- Outras redes foram criadas;
- Intercomunicação → **IP** (*Internet Protocol*);
- Conjunto combinado de protocolos → **TCP/IP**.
- Mais informações: Prates e Palma (2000)

# História da Internet

*“In the Beginning, ARPA created the ARPANET.*

*And the ARPANET was without form and void.*

*And darkness was upon the deep.*

*And the spirit of ARPA moved upon the face of the network  
and ARPA said, ‘Let there be a protocol,’ and there was a  
protocol. And ARPA saw that it was good.*

*And ARPA said, ‘Let there be more protocols,’ and it was so.  
And ARPA saw that it was good.*

*And ARPA said, ‘Let there be more networks,’ and it was so.”*

— Danny Cohen

Retirado de: <http://www.computerhistory.org/internethistory/>

# História da World Wide Web

- **Tim Berners-Lee**, do CERN (Organização Europeia para Pesquisa Nuclear), 1989 → proposta de um sistema de gerenciamento de informação baseado em *hyperlinks*.
- **Mike Sendall** → chefe/superior de Berners-Lee.
- **Proposta:** [〈http://info.cern.ch/Proposal.html〉](http://info.cern.ch/Proposal.html)

CERN DD/OC  
Information Management: A Proposal

Vague but exciting ...  
Tim Berners-Lee, CERN/DD  
March 1989

## Information Management: A Proposal

### Abstract

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext system.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control

Figura 7: Proposta

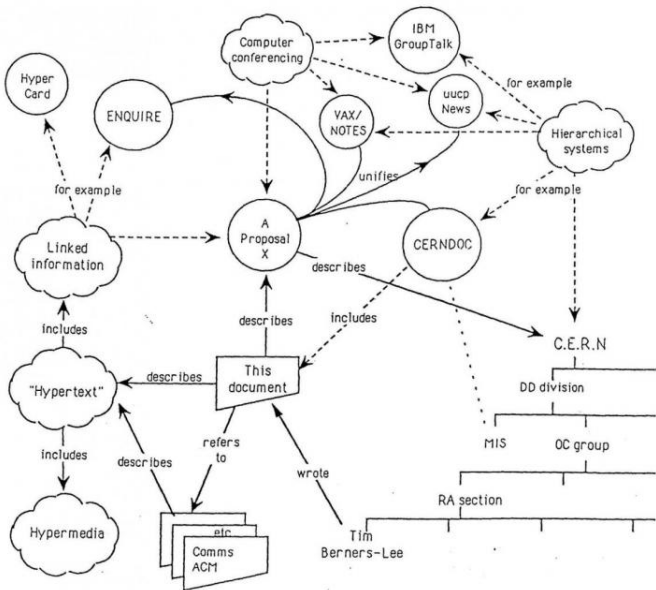


Figura 8: Diagrama

# World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) , [Frequently Asked Questions](#) .

## [What's out there?](#)

Pointers to the world's online information, [subjects](#) , [W3 servers](#), etc.

## [Help](#)

on the browser you are using

## [Software Products](#)

A list of W3 project components and their current state. (e.g. [Line Mode](#) , [X11 Viola](#) , [NeXTStep](#) , [Servers](#) , [To](#)

## [Technical](#)

Details of protocols, formats, program internals etc

## [Bibliography](#)

Paper documentation on W3 and references.

## [People](#)

A list of some people involved in the project.

## [History](#)

A summary of the history of the project.

## [How can I help ?](#)

If you would like to support the web..

## [Getting code](#)

Getting the code by [anonymous FTP](#) , etc.

Figura 9: Primeiro site: <http://info.cern.ch/hypertext/WWW/TheProject.html>

# História da World Wide Web

Outros links de interesse:

- <https://home.cern/topics/birth-web>
- <https://home.cern/about/topics/birth-web/where-web-was-born>
- <https://home.cern/about/updates/2014/03/world-wide-web-born-cern-25-years-ago>

# História da World Wide Web

- **30 de Abril de 1993** → CERN torna a Web de Domínio Público: *the basic line-mode client, the basic server and the library of common code*
- Retirado de: [〈https://home.cern/topics/birth-web/licensing-web〉](https://home.cern/topics/birth-web/licensing-web)

*“CERN relinquishes all intellectual property rights to this code, both source and binary and permission is given to anyone to use, duplicate, modify and distribute it.”*



# World Wide Web Consortium (W3C)

- **1 de outubro de 1994** → Tim Berners-Lee fundou a **World Wide Web Consortium (W3C)** (MIT, CERN e DARPA).
- **Objetivo** → desenvolver tecnologias interoperantes, de domínio público, para a **World Wide Web**.
- Uma das **metas** principais → tornar a **Web universalmente acessível** – independente de quaisquer deficiências de natureza pessoal, linguística, cultural, dentre outros.

# World Wide Web Consortium (W3C)

- Uma organização de **padronização** → chamadas de **Recomendações**.
- *HyperText Markup Language (HTML)*, a *Cascading Style Sheets (CSS)*, *eXtensible Markup Language (XML)*, o *Hypertext Transfer Protocol (HTTP)*, dentre outros.
- Recomendações → **documentos** que **especificam** o papel, a sintaxe, as regras, dentre outros, de uma tecnologia.

# World Wide Web Consortium (W3C)

- Antes de se tornar uma recomendação → **três fases principais**:
  - **Working Draft** (“Minuta”) – especifica um **rascunho ou minuta** que está em evolução;
  - **Candidate Recommendation** (“Recomendação Candidata”) – uma **versão estável** do documento que a **indústria pode começar a implementar**;
  - **Proposed Recommendation** (“Recomendação proposta”) – uma **Candidate Recommendation** considerada **madura** (ou seja, foi **implementada e testada** por um período de tempo) e pronta para ser **considerada uma recomendação do W3C**.
- Mais informações sobre o W3C: [www.w3c.org](http://www.w3c.org).
- Outras referências: Deitel (2003) e Deitel e Deitel (2009).

# Evolução da Web

- **Web 1.0** → puramente baseada em HTML.
- **Web 2.0** cada vez mais utiliza a **XML**, especialmente em tecnologias como **RSS feeds** e **serviços Web**.
- **Web 3.0:**
  - Web semântica → uso de **XML**, criando “**uma teia de significados**”.
  - Uso de maneira mais inteligente de todo o conhecimento já disponível na Internet.
- **Web 4.0/5.0?**
  - Agentes inteligentes;
  - Pesquisa distribuída;
  - Mobile Web? IoT?

# Gestão da Internet – Brasil

- Vídeo – *Fórum da Internet no Brasil - O que é o CGI.br*
- <http://youtu.be/F38J9R5wuqo>
- <http://www.cgi.br/>

# Indicadores

- `<http://data.cetic.br/cetic> (?)`
- `<http://cetic.br/pesquisa/domicilios/indicadores>`
- `<http://www.internetworldstats.com/stats.htm>`
- `<http://registro.br/estatisticas.html>`

# Arquitetura Cliente-Servidor

- **Servidor e cliente** → redes baseadas em TCP/IP.
- Arquitetura cliente-servidor.
- Estações de trabalho → processamento local.
- **Descentralização** → limites
  - Bancos de dados?
- Modelo de execução adequado?
- O que executar no servidor e o que executar no cliente?

# Protocolo HTTP

- **HTTP** → *Hypertext Transfer Protocol*.
- Modelo **cliente-servidor**.
- **Porta** padrão → 80.
- **Versão** mais recente → HTTP/2 (anterior: HTTP 1.1, em substituição)
- **Recurso** → algum tipo de **informação** identificada por um **URL** (*Uniform Resource Locator*).
  - HTML, Imagem, PHP, CSS, JS, PDF, ZIP, ...
- **Processo**:
  - **Navegador** → envia um comando HTTP ao servidor solicitando recurso (**REQUEST**);
  - **Servidor** → verifica recurso e retorna ao navegador (**RESPONSE**);
  - **Navegador** → interpreta e exibe recurso;



# Protocolo HTTP

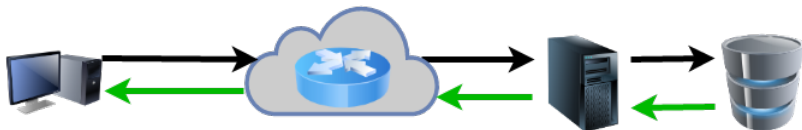


Figura 10: Solicitação e retorno de arquivo pelo servidor ao cliente

# Protocolo HTTP

- Protocolo **sem conexão** → cada comando é executado independentemente, sem qualquer conhecimento dos comandos anteriores.
  - Uma página Web simples com 4 imagens:
  - Acesso → 5 conexões TCP independentes: uma para a página HTML e uma para cada imagem.
- **HTTP 1.1** → conexões são mantidas abertas (*idle*).

# Servidores Web

- Aguardam solicitações → acesso aos recursos.
- Processa recurso → *response*.
- **Exemplos de servidores Web** (aplicativos):
  - **Internet Information Services (IIS)** → criado pela Microsoft para seus sistemas operacionais para servidores.
  - **Servidor Apache** (ou Servidor HTTP Apache) → um dos servidores Web livre mais populares, desenvolvido pela *Apache Software Foundation*.
  - **Nginx** → servidor e proxy reverso HTTP de alta performance, gratuito e livre, bem como um servidor proxy para IMAP/POP3.
- *Survey and Market share:*  
<<https://news.netcraft.com/archives/category/web-server-survey/>>

# Clientes Web

- **Navegador** (browser) → acesso recurso no servidor.
- Recebem resposta.
- Interpretam e exibem o resultado.

# Clientes Web

## ■ Exemplos de navegadores:

- Google Chrome
- Mozilla Firefox
- Internet Explorer/Edge
- Opera
- Safari
- Tor Project

## ■ **Web browser engines:** *WebKit*, *MSHTML (Trident)*, *Gecko*, *Blink*, dentre outros.

## ■ Quais **dispositivos** também possuem acesso à Internet?

# Arquitetura de sistemas Web dinâmicos

## Páginas Web Estáticas e Dinâmicas

- Solicitação de recursos → **estático** ou **dinâmico**.
- **Estático:**
  - Requisição → recurso/arquivo direto.
  - Resposta → “sempre” a mesma (HTML, imagem, CSS, JS).
- **Dinâmico:**
  - Requisição → depende do processamento – dados de entrada, horário, conteúdo atual do banco de dados, dentre outros.
  - Resposta → o *script* gera o resultado e transmite ao cliente (PHP, *Ruby on Rails*, ASP.NET, JSP).
- Referências: (DEITEL; DEITEL, 2009)

# Arquitetura de sistemas Web dinâmicos

## Criação de scripts no lado do cliente e do servidor

### ■ Scripts do lado do cliente:

- **Aplicações** → validar a entrada do usuário, para interagir com navegador, para aprimorar páginas Web, dentre outros.
- **Exemplo** → JavaScript.
- **Limitações** → dependência do suporte/autorização no navegador.
- **Segurança** → impedidos de acessar o *hardware* e o sistema de arquivos locais – atenção para possíveis vulnerabilidades.
- **Usuários** → podem visualizar o código-fonte dos *scripts* – não utilizar informações sigilosas/de alto risco.
- **Boa prática** → espelhar validações no servidor.

# Arquitetura de sistemas Web dinâmicos

## Criação de scripts no lado do cliente e do servidor

### ■ Scripts do lado do servidor:

- Maior **flexibilidade** → respostas personalizadas para os clientes.
  - Por exemplo, a **lista de vôos** num determinado período;
  - Essa tecnologia permite que clientes obtenham as **informações** do banco de dados mais **atuais** sobre os vôos, conectando-se ao servidor Web de uma empresa aérea.
- **Mais recursos** que do lado do cliente → podem acessar a estrutura de **diretórios** e **arquivos** do servidor, enquanto no lado cliente isso não é possível.
- Referências: (DEITEL; DEITEL, 2009)



# Gestão de Conteúdo

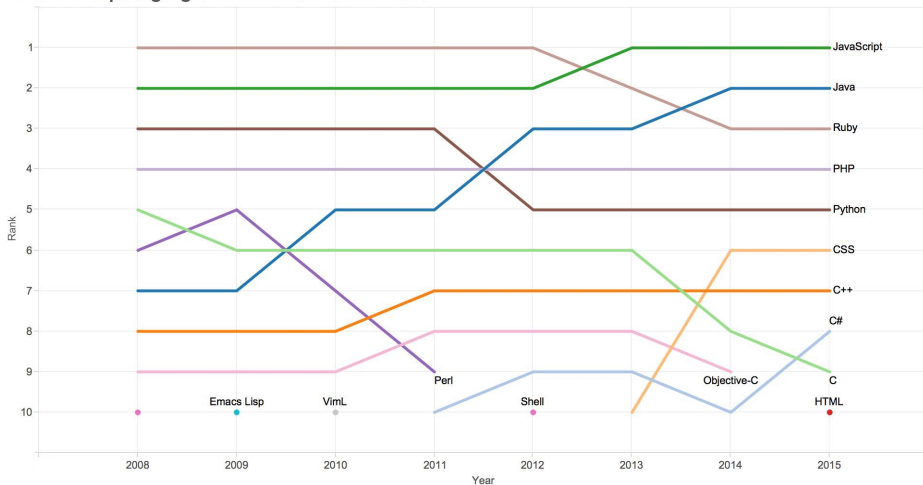
- Sistema de gestão de conteúdo → *Content Management Systems* – CMS.
- Integração de **ferramentas** → criar e editar conteúdos **sem a necessidade de programação de código**;
- **Objetivo** → estruturar e facilitar a criação, administração, distribuição, publicação e disponibilidade da informação.
- Possuem diversos **complementos** → galerias de fotos, gerenciadores de enquetes, gerenciadores de formulários, dentre outros.

# Gestão de Conteúdo

## ■ Exemplos:

- Joomla
  - WordPress
  - Drupal
  - Mambo
  - OpenCMS
  - Plone
  - PhpBB
- 
- Criação de sites diretamente na Web → Google Sites, SquareSpace, Wix, dentre outros.
  - Outras informações e referências: (PEREIRA; BAX, 2002)

Rank of top languages on GitHub.com over time



Source: GitHub.com

Figura 11: <https://goo.gl/XKVzyH>

# Linguagens

- `<https://www.tiobe.com/tiobe-index/>`
- `<https://pypl.github.io/PYPL.html>`
- `<http://desenvolvimentoparaweb.com/miscelanea/desenvolvimento-orientado-a-modinha-dom/>`
- `<https://blog.daftcode.pl/hype-driven-development-3469fc2e9b22>`

# Atualidades

## ■ Limite de dados na internet fixa:

- <http://goo.gl/r9ohJY>
- <http://goo.gl/WKCxHv>
- <https://goo.gl/A0iFRE>
- EUA também enfrenta polêmica com limitação da internet e Netflix se posiciona: <https://goo.gl/0hYpHz>

## ■ Deep Web:

- <https://goo.gl/Uu7lcX>
- <https://goo.gl/2qx1Nk>

## ■ Privacidade? Vazamentos de dados?

- Sugestão: ouvir podcast → AntiCast 245 – Vigilância e Segurança: a sua Privacidade Online

# Atualidades

## Performance e velocidade:

- 1 Banda (quantidade) x Latência (distância) – a distância pode prejudicar.
  - CDN – *Content Delivery Network*.
- 2 Imagens – retirar informações não relevantes ao contexto.
- 3 *Scripts*, CSS – minificação e *cache* (fontes).
- 4 Compactação de arquivos para envio – GZip.
- 5 Evitar/organizar recursos que causam bloqueio/espera – fontes, JS, CSS.
  - CSS – separar os que são fundamentais no início?
  - Usar *inline*?
  - Mudar ordem de carregamento de *banners*, *scripts* e outros conteúdos.

# Atualidades

- Percepção do usuário – rápido ou lento?
- Teste de desempenho:
  - <https://www.webpagetest.org/>
  - <https://developers.google.com/speed/pagespeed/> (Sugestão de Camila Ferreira)
- Referências e mais informações:
  - NerdTech 03 – Velocidade, performance e otimização  
(<https://jovemnerd.com.br/nerdcast/nerdtech-03-velocidade-performance-e-otimizacao/>);
  - Links (<https://www.alura.com.br/podcast-nerdtech/links>)

# Opiniões e Discussões



# Atividades de Revisão e Pesquisa

- 1 Apresente as vantagens e desvantagens acerca da utilização de páginas estáticas e páginas dinâmicas.
- 2 O que é interessante na utilização de gestores de conteúdos? Em quais pontos eles podem ser negativos?
- 3 Descreva questões sobre a Arquitetura de aplicação com múltiplas camadas na Web.
- 4 Apresente sua opinião sobre a limite de dados na internet fixa, bem como seus efeitos e consequências.
- 5 Escolha um site e realize o teste de desempenho por meio das aplicações sugeridas. Avalie o resultado e as possibilidades de melhoria a serem utilizadas.

# Atividades de Revisão e Pesquisa

## Assistir os seguintes vídeos

- 1 HTML5 – Uma web para todos – [http://youtu.be/DHya\\_zl4kXI](http://youtu.be/DHya_zl4kXI)
- 2 Padrões web – <http://youtu.be/xSGhV3ynmm4>
- 3 Canal NIC.br – <http://www.youtube.com/NICbrvideos>
- 4 CPBR5 – Como funciona a internet brasileira?:  
<https://youtu.be/yXlSdSSrcMY>
- 5 Como funciona a Internet?
  - Parte 1: O protocolo IP – <https://youtu.be/HNQD0qJ0TC4>
  - Parte 2: Sistemas Autônomos, BGP, PTTs. –  
[https://youtu.be/C5qNAT\\_j63M](https://youtu.be/C5qNAT_j63M)
  - Parte 3: DNS – <https://youtu.be/ACGuo26Mswl>
  - Parte 4: Governança da Internet – <https://youtu.be/ZYsjMEISR6E>

# Atividades de Revisão e Pesquisa

**Pesquisa:** Procure sobre um dos temas a seguir para discussão na próxima aula:

- 1 HTML4 e HTML5
- 2 Protocolo HTTP
- 3 Protocolo BGP e PTTs
- 4 Sistema de Gestão de Conteúdo
- 5 Bootstrap

# Encerramento


## Muito obrigado !





Imagens retiradas de: <https://goo.gl/oajVyp> e <https://goo.gl/3H1DM7>

# Referências

 DEITEL, H. M. *XML: Como Programar*. São Paulo: Bookman, 2003.

 DEITEL, H. M.; DEITEL, P. J. *Ajax, Rich Internet Applications e desenvolvimento Web para programadores*. São Paulo: Pearson, 2009.

 PEREIRA, J. C. L.; BAX, M. P. Introdução à gestão de conteúdos. *Gestão & Tecnologia*, v. 1, 2002. Disponível em: <http://www.unipel.edu.br/periodicos/index.php/get/article/view/104>.

 PRATES, R.; PALMA, L. *TCP/IP - Guia de Consulta Rápida*. São Paulo: Novatec, 2000.