CSE414: WEB ENGINEERING

Daffodil International University

OVERVIEW

- Course Policies-
  - Zero Tolerance on Plagiarism
  - Grading will be based on university policy
- Contact Details-
  - Office –
  - Cell – [Text Preferred]
  - Email – [Write E-mail subject properly]
- Google Classroom Code - arld4o4
- BLC/Online Module Link- [Will be updated soon]
- More at, my google site

CONTENTS-

- ✔ Internet and the Web
- ✔ Client-Server Paradigm

LECTURE OUTCOME

- ✔ Differentiate between Web and Internet
- ✔ Understand Client-Server Paradigm
- ✔ Select the appropriate tools

WEB ≠ INTERNET

- Internet
  - A physical network of networks connecting billions of computers and other devices using common protocols (TCP/IP) for sharing and transmitting information
- World Wide Web [Old]
  - A collection of interlinked multimedia documents
    (web pages stored on internet connected devices and accessed using a common protocol (HTTP))
- Key distinction:
  - The internet is hardware plus protocols while the world wide web is software plus protocols
  - The world wide web is an application using the internet to transmit information, just like many others, for example, email, SSH, FTP

HISTORY(1/3)

- 1969: ARPANET (precursor of the Internet)
- 1971: First e-mail transmission, File Transfer Protocol
- 1972-1980:
  - Vadic VA3400 modem (1,200 bit/s over phone network)
  - RSA public-key cryptography
  - EPSS/SERCnet (first UK networks between research institutions)
- 1981: IBM PC 5150
- 1982: TCP/IP standardized
- 1985 : FTP on TCP standardized

HISTORY(2/3)

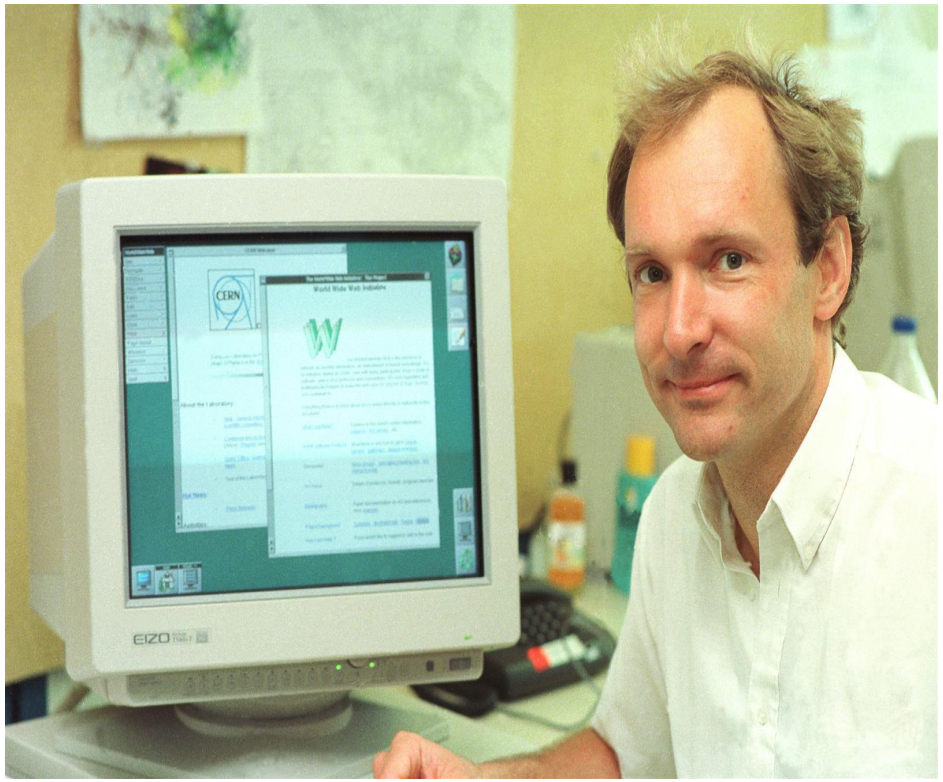
- mid 1980s: Janet (UK network between research institutions with 2 Mbit/s backbone and 64 kbit/s access links)
- 1986:U.S. Robotics HST modem (9600 bit/s) TCP/IP networks expand across the world
- Late 1980s: TCP/IP networks expand across the world
- 1991:
  - Janet adds IP service
  - Gopher / World Wide Web
  - GSM (second generation cellular network) digital, circuit switched network for full duplex voice telephony
- 1995: First public releases of JavaScript and PHP
- 1997: World Wide Web slowly arrives on mobile phones

HISTORY(3/3)

- Current Applications:
  - Communication via e-mail, Twitter, etc
  - Joint manipulation of concepts and actions: Collaborative editing, Crowd sourcing, Wikis (Wikipedia)
  - E-Commerce: Online auctions and markets
  - Social media, social networks,
  - virtual learning environments

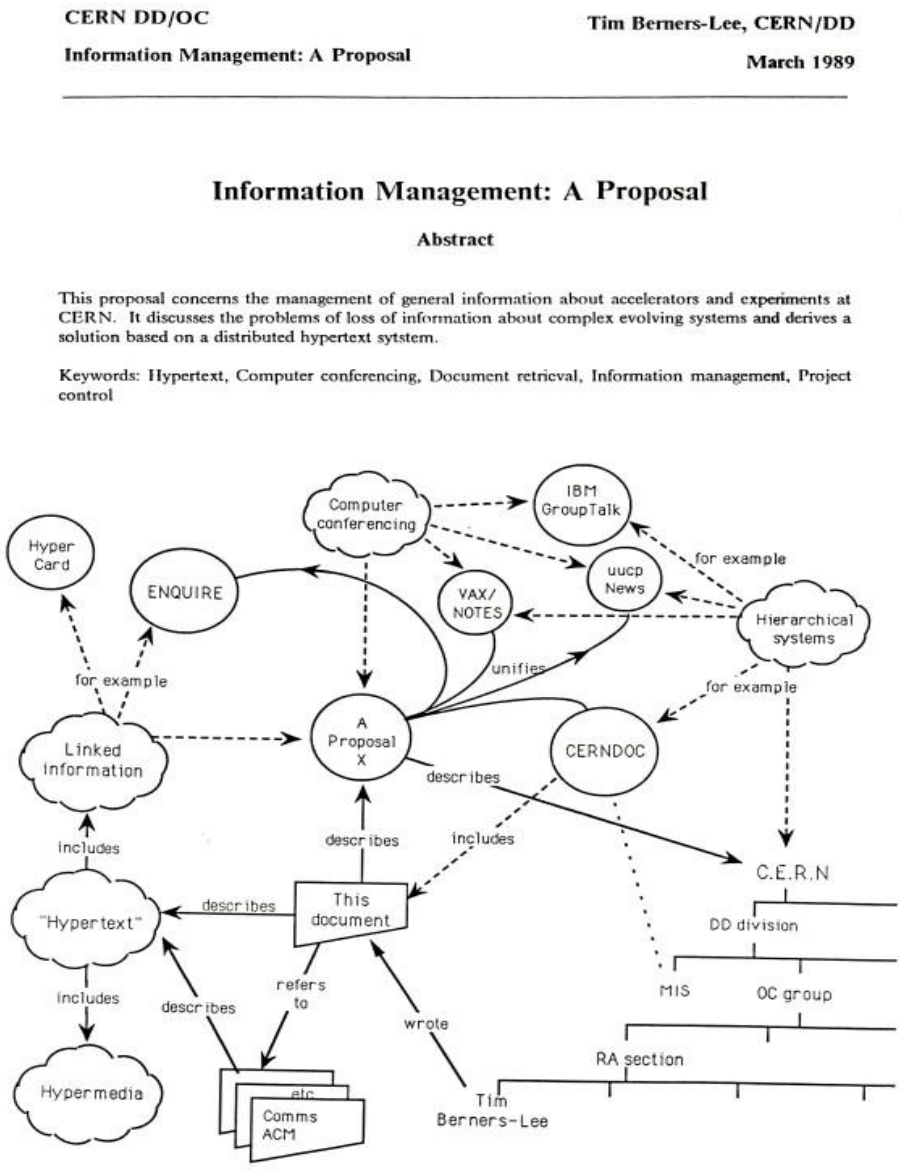
WHERE THE WEB WAS BORN

- "***Tim Berners-Lee***, a British scientist, invented the World Wide Web (WWW) in 1989, while working at ***CERN***. The Web was originally conceived and developed to meet the demand for automated information-sharing between scientists in universities and institutes around the world."
  - Go to CERN's official site



HOW THE WEB BEGAN

- An image of the first page of Tim Berners-Lee's proposal for the World Wide Web in March 1989



CERN DD/OC

Information Management: A Proposal

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 sytstem.

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

WEB PROGRAMMING VERSUS APP PROGRAMMING

- Web Programming relies on web browsers as means to render user interfaces that are coded in HTML/CSS
- Web Programming relies on HTTP as the main protocol to exchange information within a distributed system
- Web-based apps use a mix of server-side and client-side computing
- Web-based apps can be changed almost instantaneously and on a per-user / per-use basis
- App Programming relies on directly coded 'native' interfaces (Swift/Java)
- App Programming can rely on arbitrary protocols to exchange information within a distributed system
- Programmers have more flexibility and more control when developing 'traditional' apps
- It is not obvious which approach is better and in which situation

STATIC VS. DYNAMIC WEBPAGES(1/2)

- **STATIC WEBPAGE**
- Many Web pages are still static in nature.
  - Contents (text/links/images) are the same each time the webpage is accessed.
  - e.g., online documents, many personal homepages
- HyperText Markup Language (HTML) and Cascading Style Sheets (CSS) are used to specify text, image, and page format, along with styling the page for various effects (backgrounds, colors, table layout, page margins, etc.).

STATIC VS. DYNAMIC WEBPAGES(2/2)

- **DYNAMIC PAGE**
- As the Web continues towards more and more online services and e-commerce continues to grow, Web pages must also provide dynamic content.
  - Pages can be fluid, changeable (e.g., rotating banners, inclusion of "real-time" data, etc.).
  - Must be able to react to the user's actions, request and process info, tailor services.
    - e.g., amazon.com, YouTube, any e-commerce website, online email services, etc.

DISTRIBUTED SYSTEM

- Also known as distributed computing
- System with multiple components
- Located on different machines that communicate and coordinate actions
- Appear as a single coherent system to the end-user.
- **Benefits and challenges of distributed systems**
- Computing happens independently on each node
  - easy and generally inexpensive to add additional nodes and functionality as necessary
- Fault-tolerant as they can be made up of hundreds of nodes that work together
  - The system generally doesn't experience any disruptions if a single machine fails
- Efficient because work loads can be broken up and sent to multiple machines

CLIENT-SIDE PROGRAMMING(2/2)

- **Jquery**
- first released in 2006, is a JavaScript library to help with cross-browser compatibility issues.
- **Java applets**
- Can define small, special-purpose programs in Java called applets.
- Provides (almost) full expressive power of Java (but with more overhead).
- Good for data-heavy tasks or more complex tasks such as graphics

TYPES OF DISTRIBUTED SYSTEMS

- **Client-server**—Clients contact the server for data, then format it and display it to the end-user. The end-user can also make a change from the client-side and commit it back to the server to make it permanent.
- **Peer-to-peer**—There are no additional machines used to provide services or manage resources. Responsibilities are uniformly distributed among machines in the system, known as peers, which can serve as either client or server.
- **Three-tier**—Information about the client is stored in a middle tier rather than on the client to simplify application deployment. This architecture model is most common for web applications.
- ***n*-tier**—Generally used when an application or server needs to forward requests to additional enterprise services on the network.

CLIENT-SERVER PARADIGM

- You already know this from CSE313: Computer Networks
- Let's revisit,
  - Server program sends copies of documents on request
  - Requires computer on Internet and server software always running
  - Client program sends message to server to request copy of document
  - Clients and servers communicate via TCP/IP
  - Client and server may establish "persistent connection" so that all pages after first arrive more quickly
- More…

CLIENT-SIDE PROGRAMMING(1/2)

- Can download program with a webpage, execute the program on the client's machine.
- Simple, generic, but sometimes insecure (e.g. cross-site scripting attacks).
- **JavaScript**
  - A scripting language for Web pages, developed by Netscape in 1995.
  - Uses a C++/Java-like syntax, so familiar to many programmers, but simpler.
  - JavaScript is good for adding dynamic features to Web page, controlling forms, and the GUI.
  - Requires users to have this technology enabled on their browsers.

SERVER-SIDE PROGRAMMING(1/2)

- Can store a program on a web server, and supply a link from a webpage to execute that program. And you can also accept input from a user in terms of "filling in blanks" and/or file upload(s), etc.
- The process of doing this can be more complex, requires server privileges, but can still be (mostly) secure with proper precautions.
- Common Gateway Interface (CGI) programming
- Programs are written to conform to the CGI.
- When a webpage submits, the data from the page is sent as input to the CGI program. CGI program executes on the server and sends its results back to browser as a webpage.
- Good if computation is large/complex or requires access to private data.

SERVER-SIDE PROGRAMMING(2/2)

- Other server-side programming technologies include:
- Active Server Pages (ASP)
- Java Servlets
- **PHP (You will learn this one in this course!)**
- Server Side Includes
- Ajax (using JavaScript on the client side too)
- Some of these are vendor-specific alternatives to CGI (such as Microsoft's ASP).
- They often provide many of the same capabilities as CGI programs but use HTML-like tags (such as PHP).
- Some of these technologies might require functionality to be enabled in the client's browser (e.g. Ajax)

Security

- Identification
- Confidentiality
- Authentication
- Integrity
- Non-Repudiation
- Availability
- Reliability
- Accountability
- Access Control

Web Attacks

- **Cross-site scripting (XSS).** That involves an attacker uploading a piece of malicious script code onto your website that can then be used to steal data or perform other kinds of mischief. Although this strategy is relatively unsophisticated, it remains quite common and can do significant damage.
- **SQL Injection (SQLI).** This happens when a hacker submits destructive code into an input form. If your systems fail to clean this information, it can be submitted into the database, changing, deleting, or revealing data to the attacker.
- **Path traversal.** Also resulting from improper protection of data that has been inputted, these webserver attacks involve injecting patterns into the webserver hierarchy that allow bad actors to obtain user credentials, databases, configuration files, and other information stored on hard drives.
- **Local File Inclusion.** This relatively uncommon attack technique involves forcing the web application to execute a file located elsewhere on the system.
- **Distributed Denial of Service (DDoS) attacks.** Such destructive events happen when an attacker bombards the server with requests. In many cases, hackers use a network of compromised computers or bots to mount this offensive. Such actions paralyze your server and prevent legitimate visitors from gaining access to your services.

Protection Against Attack

- **Automated vulnerability scanning and security testing.** These programs help you to find, analyze, and mitigate vulnerabilities, often before actual attacks occur. Investing in these preventive measures is a cost-effective way to reduce the likelihood that vulnerabilities will turn into cyber disasters.
- **Web Application Firewalls (WAFs).** These operate on the application layer and use rules and intelligence about known breach tactics to restrict access to applications. Because they can access all layers and protocols, WAFs can be highly effective gatekeepers when it comes to shielding resources from attack.
- **Secure Development Testing (SDT).** This instruction is designed for all security team members, including testers, developers, architects, and managers. It provides information about the newest attack vectors. It assists the task force in establishing a baseline and developing a practical, dynamic approach to preventing website attacks and minimizing the consequences of breaches that cannot be stopped.

AN EXERCISE

- Pick some of your favorite websites and try to identify
  - the static components
  - the dynamic components
  - Which sites are using JavaScript?
  - Which are using Java applets?
  - Which are using server-side elements such as CGI programs?
- **READINGS**
  - http://cgi.csc.liv.ac.uk/~ullrich/COMP519/notes/lect01.pdf
  - https://cgi.csc.liv.ac.uk/~martin/teaching/comp519/NOTES/overview.pdf

REFERENCES

- https://home.cern/science/computing/birth-web/short-history-web
- https://www.cs.purdue.edu/homes/bxd/inter/tableOfContents.html

ACKNOWLEDGEMENT

- This module is designed and created with the help from following sources-
  - https://cgi.csc.liv.ac.uk/~ullrich/COMP519/
  - http://www.csc.liv.ac.uk/~martin/teaching/comp519/
- My sincere grattitude to Professor Hustadt and Professor Martin for their support and materials.
- Following lecture materials also use different sources including this and will be mostly mentioned at reference section.