
Introduction to Web Engineering

Denis Helic

- ▶ Two faces of the Web
- ▶ The Web as an application platform
- ▶ The Web as a huge database

- ▶ The Web as an application platform
- ▶ E-commerce crossing national boundaries
- ▶ Porting of legacy systems to the Web
- ▶ Wireless applications → mobile Web applications
- ▶ We increasingly depend on a range of Web applications

- ▶ The Web as a huge database
- ▶ New developments in data management on the Web
- ▶ Data is the key of the new Web (2.0) applications
 - ▶ Control over a certain data resource often leads to market control
 - ▶ Amazon is the primary source for bibliographic data on books
 - ▶ Every map served by MapQuest comes from NavTeq

- ▶ Organizations that own data provide services on the top of data
 - ▶ Retrieving, searching, billing
- ▶ Web Services: SOAP and REST
 - ▶ e.g. Amazon Web Services

- ▶ Data integration through mashups
 - ▶ Combine data from two sources and integrate services via AJAX
 - ▶ Mashups: <http://www.programmableweb.com/>
- ▶ The Web is an application platform on top of its database platform

Web Engineering - Definition

- ▶ We rely on Web applications → they need to be reliable and perform well
- ▶ To build such applications we need a sound methodology
 - ▶ Process, tools, guidelines
- ▶ Web engineering
 - ▶ Systematic, scientific, engineering and management approach
 - ▶ Develop, deploy and maintain qualitative Web applications

Web Engineering - Multidisciplinary Process

- ▶ Web software engineering is similar to traditional software engineering
 - ▶ The process follows a similar methodology
 - ▶ But at each step we need to take care about special issues related to the Web
- ▶ Requirements engineering
- ▶ System analysis, system architecture and design
- ▶ Implementation
- ▶ Testing

Web apps UR vs. traditional apps UR(1/3)

- ▶ New dimension of dealing with users
- ▶ In traditional SE you know your users
 - ▶ Easy to derive user requirements
- ▶ In WE you don't know your users
 - ▶ Potentially, the whole world!

Web apps UR vs. traditional apps UR(2/3)

- ▶ Even if you have a target group it can change in the future
- ▶ Additionally, you need to compete with others over the users
 - ▶ Usability, accessibility, graphic design become very important
- ▶ Popularity is important!
 - ▶ Google based its search engine on popularity → a large success
- ▶ No Web success has been advertised
 - ▶ Recommendations propagating directly from one user to another

Web apps UR vs. traditional apps UR(3/3)

- ▶ One very popular approach to collect user requirements on the Web
- ▶ Start a basic version of a Web app, e.g., a beta version
 - ▶ (Constant) Web 2.0 beta
- ▶ Build a user basis, collect the feedback
- ▶ Develop the app by taking into account the feedback
 - ▶ Social aspects of software development
- ▶ No need for usability tests, user simulations before the start of the app

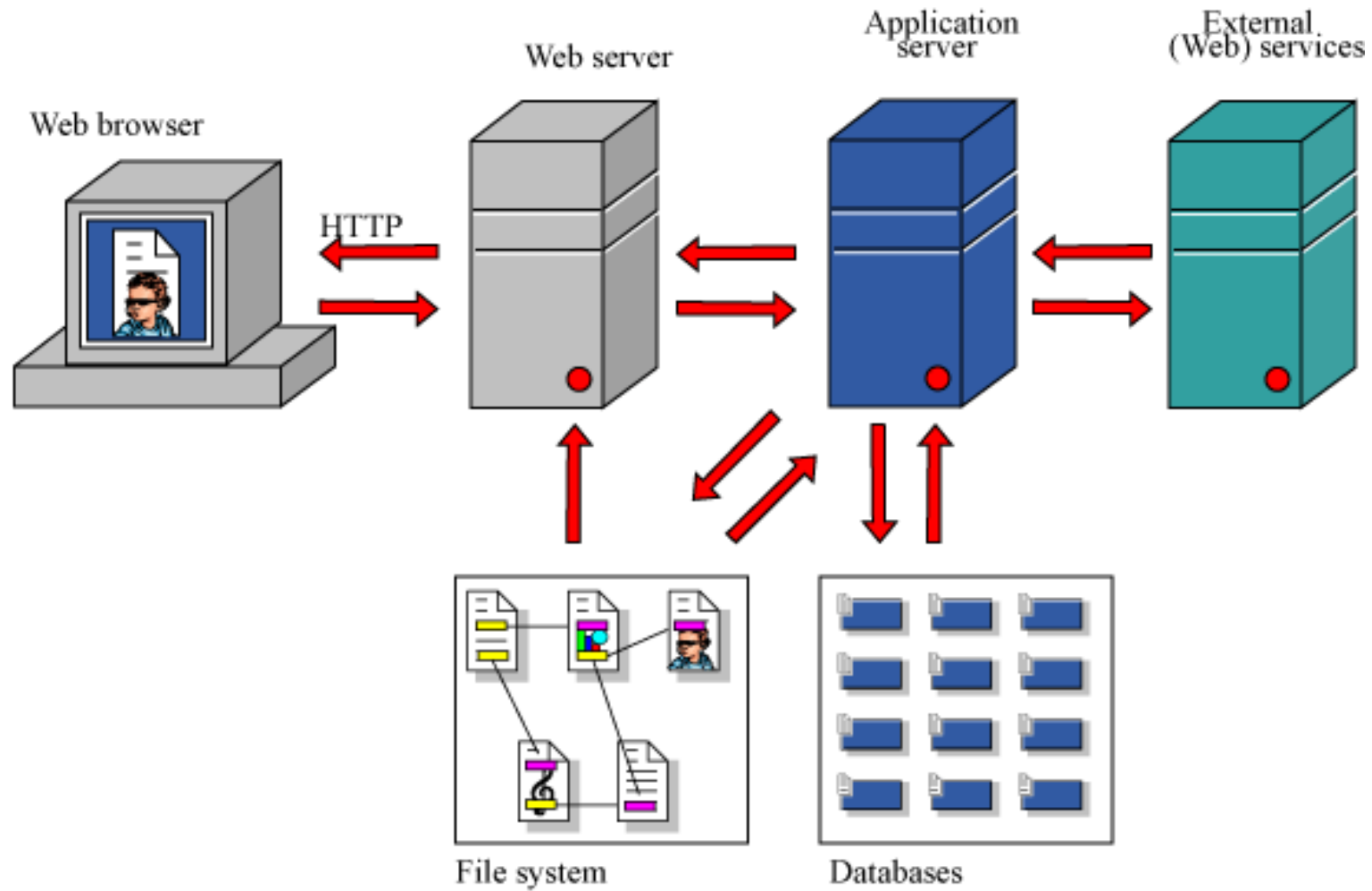
Web apps SA vs. traditional apps SA(1/2)

- ▶ Generic architecture of user-oriented software applications
- ▶ Process (application) logic
- ▶ Data management
- ▶ User interface

Web apps SA vs. traditional apps SA(2/2)

- ▶ A typical example
- ▶ Managing student records in university
- ▶ Process logic: add a student, delete a student, insert grades, exams, ...
- ▶ Data management: tables of students, grades, exams (relational databases)
- ▶ User interface: GUI with menus, buttons, text fields, etc.

Generic architecture of Web applications



User interface in Web applications(1/3)

- ▶ Traditionally, developers deal mainly with process logic and data management
- ▶ User interface lately in the process and in a hurry
 - ▶ Poor quality of interfaces → users complain
- ▶ The Web is appealing because of its consistent user interface
 - ▶ HTML, platform independence
 - ▶ One of the reasons for the success!
 - ▶ It is a huge advantage for developers!

User interface in Web applications(2/3)

- ▶ New developments in UI for the Web started as Google introduced GMail and Google Maps
 - ▶ The collection of technologies used by Google was AJAX
- ▶ AJAX = Asynchronous JavaScript and XML
 - ▶ Standards-based presentation using XHTML and CSS
 - ▶ Dynamic display and interaction using the Document Object Model
 - ▶ Data interchange and manipulation using XML (recently also JSON)
 - ▶ Asynchronous data retrieval using XMLHttpRequest
 - ▶ JavaScript binding everything together

User interface in Web applications(3/3)

- ▶ These new UIs combine
 - ▶ Accessibility of the Web
 - ▶ Responsivness of standalone GUI applications
 - ▶ Remedy for usability issues (no waiting, rich widgets, event-based, etc.)

- ▶ Traditionally, app logic manages the app state
 - ▶ E.g., the current state of the data, user inputs, etc.
- ▶ Typically, Web browser supports only HTML and does not have direct connection to the app logic
 - ▶ Communication over network and HTTP with the app logic
- ▶ HTTP stateless (connection-less)
- ▶ Web server or client needs to track users and sessions

- ▶ However, Web server provides only low-level tracking
 - ▶ Responsibility of the app logic to manage sessions
 - ▶ Manage it within the application server (or recently within AJAX app logic)
- ▶ App server has other responsibilities as well
 - ▶ Provides the basic app functionality
 - ▶ Communicates with data management modules
 - ▶ Communicates with external functionality (Web services)

- ▶ We need to take care of separation of concerns
 - ▶ Separate user interface and session management from the app logic
 - ▶ Separate user interface and session management from data management
 - ▶ ...
- ▶ Design patterns

Hypertext engineering on the Web(1/4)

- ▶ Hyperlinks are a specific component of a Web-based user interface
- ▶ Hyperlinks are also the way to relate data items to each other
 - ▶ Similar to foreign keys in relational databases

Hypertext engineering on the Web(2/4)

- ▶ For Web applications
- ▶ URL/URI for addressing of Web pages (of Web data items)
- ▶ Important to provide meaningful URLs
- ▶ Single URL (e.g. /app.cgi)
 - ▶ No bookmarks, difficult for humans, bad user interface
 - ▶ Problems with search engines

Hypertext engineering on the Web(3/4)

- ▶ Multiple URLs
 - ▶ /student/add /student/show
 - ▶ /student/exam/add /student/exam/show
 - ▶ Meaningful, easier for humans
 - ▶ Bookmarks
 - ▶ Search engines can retrieve different parts and index it

Hypertext engineering on the Web(4/4)

- ▶ For data management
- ▶ Combination of URLs and data operations (HTTP)
- ▶ GET /student/ GET /student/exam/
- ▶ PUT /student/ PUT /student/exam/
- ▶ Semantics from HTTP - automatic integration possible

- ▶ Web server, application server, Web services
 - ▶ All manage data → design patterns to separate concerns
- ▶ Disparate and numerous data sources
 - ▶ File system, databases, interlinked documents, document formats, metadata, etc..
 - ▶ Design patterns to abstract access to data sources

- Typically, Web applications deal with relational databases
- Need to manage relational data in object-oriented applications
- Use design patterns like Data Access Object (DAO)
- Use object/relational mapping, like Hibernate framework

- ▶ Information retrieval
 - ▶ How to find what I'm looking for?
- ▶ One approach are search engines with full-text processing
- ▶ Another approaches manage links
 - ▶ Links in databases, or within documents
- ▶ Mixed approach: full-text and links, e.g. Google

- ▶ Yet another approach is managing metadata
- ▶ Metadata is data about other data
- ▶ On the web
 - ▶ Tag information items (everything that you can access via URL) in a structured manner
 - ▶ Search inside metadata
- ▶ Social Web 2.0 app `http://del.icio.us/` and `http://www.flickr.com/`

- ▶ Data and document formats
- ▶ XML!
 - ▶ You should think about HTML as yet another application of XML
 - ▶ XHTML
- ▶ XML as a technology for managing documents
 - ▶ For presentation we need style sheets (CSS, XSLT, XSL-FO)

- ▶ General info on Web engineering
- ▶ Web Engineering: An Introduction
<http://ieeexplore.ieee.org/iel5/93/19981/00923949.pdf?isNumber=19981&prod=JNL&arnumber=923949&arSt=14&ared=18&arAuthor=Ginige%2C+A.%3B+Murugesan%2C+S>.
- ▶ Web Engineering: Creating a Discipline among Disciplines
<http://ieeexplore.ieee.org/iel5/93/19845/00917974.pdf?isNumber=19845&prod=JNL&arnumber=917974&arSt=82&ared=87&arAuthor=Deshpande%2C+Y.%3B+Hansen%2C+S>.

Further Readings(2/3)

- ▶ Both articles from special issues of IEEE Multimedia from 2001
<http://ieeexplore.ieee.org/xpl/tocresult.jsp?isNumber=19981&puNumber=93>
<http://ieeexplore.ieee.org/xpl/tocresult.jsp?isNumber=19845&puNumber=93>
- ▶ Part of the IEEE Explore digital library
 - ▶ Free access for IPs from the TUG

- ▶ Article on architecture of Web applications
- ▶ Grady Booch: The Architecture of Web Applications
http://www-106.ibm.com/developerworks/ibm/library/it-booch_web/
- ▶ Grady Booch is one of the creators of UML