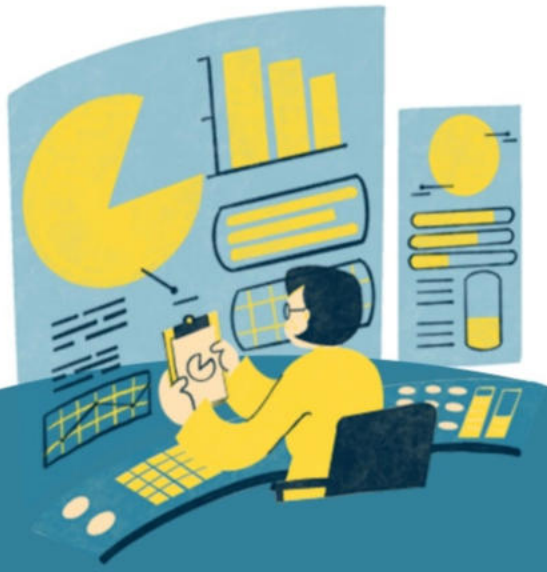
WEB APPS ENGINEERING



Each line brings ideas to life—pixels become stories, buttons become experiences.

NF-2025

Lecture 1 Notes

Introduction to Web Applications Engineering

Course Objectives

Upon successful completion of the course the student will:

- Be able to understand the concepts, principles and methods of Web engineering.
- Be able to apply the concepts, principles, and methods of Web engineering to Web applications development.
- Be familiar with current Web technologies.
- Be familiar with Web application development software tools and environments currently available on the market.

Textbook

Web Engineering: A Practitioner's Approach by Roger Pressman and David Lowe, McGraw-Hill, 2009.

The motivation for studying web engineering

Traditional Software development Project Success Rate

Why? Because of **Software Development Mistakes:**

1. **Executing an Incomplete Strategy**: Many enterprises start building software without a complete strategy for execution. Enterprises must detail out the development stages and create a step-by-step plan of how to build the product.
2. **Considering Cost Over Value**: Companies that focus on price rather than value and quality regret their decision if it is based solely on the cost of development. A focus on skills and experience in development will charge higher, but they will also build better software.
3. **Misunderstanding the Requirement**: Every project has unique needs (requirements), and improper comprehension leads to future errors.
4. **Vague Project Estimates**: One of the biggest mistakes in global custom software development is inaccurate and vague project estimates (time or money).
5. **Excessive Focus on Designing**: The primary focus should be the functionality of the software rather than its design. Clients want software that can help solve their problems rather than look beautiful.
6. **Skipping Product Testing:** Testing ensures that the product goes to the client in a completely functional manner. Not testing the product can lead to long-term damage.
7. **Unclear Communication & Feedback**: Full-cycle custom software development services require a high amount of communication to ensure that each component is working properly. On the other hand, feedback is critical for product improvement.

Definition of Software Engineering

Software Engineering is defined as the application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures, and associated documentation.

Definition of Web Engineering

Web Engineering is the application of systematic and quantifiable approaches (concepts, methods, techniques, tools) to cost effective requirements analysis, design, implementation, testing, operation, and maintenance of high-quality Web applications.

Web-Based Systems

In the **early days of the Web**, we built systems using informality, urgency, intuition, and art

- ***Informality*** leads to an easy work environment—one in which you can do your own thing.
- ***Urgency*** leads to action and rapid decision making.
- ***Intuition*** is an intangible quality that enables you to "feel" your way through complex situations.
- ***Art*** leads to aesthetic form and function—to something that pleases those who encounter it.

Problem is—this approach can and often does lead to problems

As **WebApps become larger** and more **complex**:

1. **Informality** remains, but some degree of requirements gathering and planning are necessary
2. **Urgency** remains, but it must be tempered by a recognition that decisions may have broad consequences
3. **Intuition** remains, but it must be augmented by proven management and technical patterns
4. **Art** remains, but it must be complemented with solid design

Bottom line 😎 (الخلاصة): we must adapt the old-school approach to the realities of a Web 3.0 world.

And What's the Response? Web Engineering

The Web is:

- **An indispensable technology:** In virtually every aspect of modern living. **Ex:** business, commerce, communication, education, engineering, entertainment, finance, government, industry, media, medicine, politics, science, and transportation.
- **A transformative technology:**
  - Changes the way we do things
  - Changes the way we acquire and disseminate information

- **An evolving technology**

Bottom line—high impact on everyone in the modern world

Web Apps

The term *Web application* (WebApp) encompasses:

- Everything from a simple Web page that might help a consumer compute an automobile lease payment to a comprehensive website that provides complete travel services for businesspeople and vacationers.
- Included within this category are complete websites, specialized functionality within websites, and information-processing applications that reside **on** the Internet or on an Intranet or Extranet.
    - An **intranet** is a private, internal network for an organization's employees to share information and collaborate.
    - An **Extranet** is a private network that allows controlled access to external parties, like suppliers and partners, to share information and collaborate with the organization.
    - The **key difference** lies in the user base: intranets are for insiders, and extranets are for specific outsiders.

WebApp Attributes

1. **Network intensiveness:** Every WebApp resides on a network and must serve the needs of a diverse community of clients. The network will enable communication between client-based features of the WebApp and the server(s) hosts the WebApp.
2. **Concurrency:** A large number of users may access the WebApp at one time. In some cases, the actions of one user or one set of users may have an impact on the actions of other users or the information presented to other users.
3. **Unpredictable load:** The number of users of the WebApp may vary from day to day. But WebApp must be capable of handling an indeterminate number of events simultaneously.

4. **Performance:** If a WebApp user must wait too long (for access, for server side processing, for client-side formatting and display), he or she may decide to go elsewhere.
5. **Availability:** Although an expectation of 100 percent availability is unreasonable, users of popular WebApps often demand access on a "24/7/365" basis. WebApp must be designed to achieve this ideal (or something very close to it).
6. **Data driven:** The primary function of many WebApps is to use hypermedia to present text, graphics, audio, and video content to the end user. In addition, WebApps are commonly used to access information that exists on databases that are not an integral part of the Web-based environment (e.g., e-commerce or financial applications).
7. **Content sensitive:** The quality and aesthetic nature of content remains an important determinant of the quality of a WebApp. Nontechnical people demand simple, yet meaningful content presentation.
8. **Continuous evolution:** Unlike conventional application software that evolves over a series of planned, chronologically spaced releases, WebApps evolve continuously.
9. **Immediacy:** Although *immediacy*—the compelling need to get software to market quickly—is a characteristic of many application domains, WebApps often exhibit a time-to-market that can be a matter of a few days or weeks.
10. **Security:** Because WebApps are available via network access, it is difficult, if not impossible, to limit the population of end users who may access the application. In order to protect sensitive content and provide secure modes of data transmission, strong security measures must be implemented throughout the infrastructure that supports WebApp and within the application itself.
11. **Aesthetics.** An undeniable part of the appeal of a WebApp is its look and feel. When an application has been designed to market or sell products or ideas or provide services that generate revenue, aesthetics may have as much to do with success as technical design.

WebApp Types/ Categories

1. **Informational WebApp is** one that contains read-only content with simple navigation and links.
2. **Download WebApps have a** *download* capability for allowing visitors to the WebApp to download files like the product specs pdfs, images, videos, ….
3. **Customizable WebApps** are *customizable* for each user. The content presented at the website to the specific needs of each customer type, using jargon and presenting content that will meet their needs.
4. **Interaction WebApps** You want to create a feeling of community among your visitors, a place where people can chat, ask and answer questions, provide product testimonials, and the like.
5. **User Input WebApps** implement forms-based input so that every request is organized in a predictable manner.
6. **Transaction-Oriented WebApps** are web applications designed to handle a series of related operations as a single, atomic unit of work, ensuring data integrity and consistency. Examples include e-commerce sites for placing orders and online banking systems for fund transfers, where a sequence of actions (like charging a credit card and updating inventory) must either succeed entirely or fail without any changes.
7. **Service-Oriented WebApps:** are web apps built on a service-oriented architecture, where functionality is delivered as independent, loosely coupled services that can be reused across different applications. These services communicate over a network (like the web) and are separate from the user interface, allowing for flexibility and integration with other systems.
8. **Web Portals:** is a specific type of web app that aggregates information from multiple sources into a single, centralized interface. Portals are designed to be a "gateway" to a variety of services and data, providing users with a consolidated and often customizable single access point.
9. **Database Access** in web applications involves connecting a web application's backend to a database system to store, retrieve, update, and delete data. This is crucial for dynamic web applications that need to manage user information, content, or other persistent data.

10. **Data Warehousing** web apps are web-based tools that access, store, and analyze data from multiple sources (Databases) for reporting and business intelligence. Examples: cloud-based platforms like Snowflake, Google BigQuery, AWS Redshift, and Azure Synapse Analytics. These applications are designed to support decision-making by providing a central, organized repository for historical data that can be queried and analyzed through a web browser.

Lecture 2 Notes

Web Engineering

Web Engineering

- **We define it this way:** Web engineering proposes an *agile*, yet *disciplined framework* for building *industry-quality* WebApps.
- We must understand the meaning of:
    - Agile
    - Disciplined framework
    - Industry quality

Why Agility?

- Business strategies and rules change rapidly
- Management demands near-instantaneous responsiveness (even when such demands are completely unreasonable).
- Stakeholders often don't understand the consequences of the Web and keep changing their minds even as they demand rapid delivery
- An agile approach helps cope with this fluidity and uncertainty.

What is an Agile Process?

- Agile Web engineering combines philosophy and a set of development guidelines (Process). The **philosophy** encourages:
    - customer satisfaction
    - early incremental delivery of the WebApp

- small, highly motivated project teams
- informal methods
- minimal work products
- overall development simplicity.

- An **agile process** stresses delivery over analysis and design (although these activities are not discouraged), and active and continuous communication between developers and customers.

Underlying Agility Principles

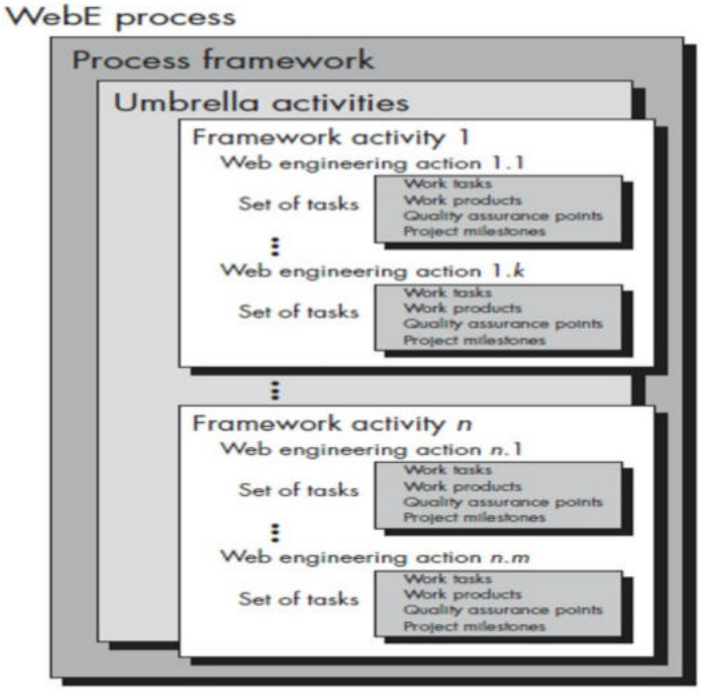
- Our highest priority is to **satisfy the customer** through early and continuous delivery of valuable software.
- **Welcome changing requirements**, even late in development. Agile processes harness continuous change for the customer's competitive advantage.
- Delivering working software **increments frequently**, from as often as every few days to every few months, with a preference to the shorter timescales.
- Businesspeople and developers must **work together daily** throughout the project.
- Build projects around **motivated people**. Give them the environment and support they need and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation**.
- **Working software** is the primary measure of progress.
- Agile processes promote **sustainable development**. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to **technical excellence and good design** enhances agility.
- **Simplicity**—the art of maximizing the amount of work not done—is essential.
- The best architecture, requirements, and designs emerge from **self-organizing teams**.

- At **regular intervals**, the team reflects on how to become more effective, then tunes and **adjusts** its behavior accordingly.

What is a WebE Framework?

- A framework is a set of activities that will always be performed for every Web engineering project – though the nature of the activities might vary to suit the project.
- Each framework activity is composed of a set of actions (e.g., design is a WebE action).
- **Actions encompass:**
  - work tasks: accomplish some part of the work implied by the action.
  - work products
  - quality assurance points, and
  - project milestones
- A framework also has a set of "**umbrella activities**": that are applicable across the entire WebE process.

A Generic Framework



The WebE Framework: Activities

The following WebE activities are part of a *generic framework* and are applicable to the vast majority of WebApp projects:

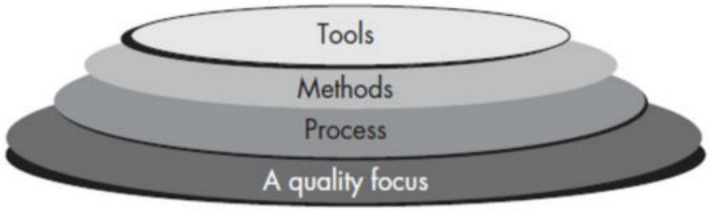
- **Communication:** Involves heavy interaction and collaboration with the customer (and other stakeholders) and encompasses requirements gathering and other related activities.
- **Planning:** Establishes an incremental plan for the WebE work.
- **Modeling:** Encompasses the creation of models that assist the developer and the customer to better understand WebApp requirements and the design
- **Construction:** Combines both the generation of HTML, XML, Java, and similar code with testing that is required to uncover errors in the code.
- **Deployment:** Delivers a WebApp increment to the customer who evaluates it and provides feedback based on the evaluation.

Adapting the Framework

- Adapt to the problem, to the project, to the team, and to the organizational culture
    - And continue to adapt throughout the project as circumstances change!
- Adaptation leads to:
    - Overall flow of activities, actions, and tasks and the interdependencies among them
    - Degree to which work tasks are defined within each framework activity
    - Degree to which work products are identified and required
    - Manner in which quality assurance activities are applied
    - Manner in which project tracking and control activities are applied
    - Overall degree of detail and rigor with which the process is described
    - Degree to which customers and other stakeholders are involved with the project

- Level of autonomy given to the software project team
- Degree to which team organization and roles are prescribed

The Influence of Software Engineering on WebApps



Software engineering is a layered technology:

1. **Quality:** foster a continuous process improvement culture.
2. **Process:** the glue that holds the technology layers together and enables rational and timely development of computer software.
3. **Methods:** provide the technical how-to's for building software.
4. **Tools:** provide automated or semiautomated support for the process and the methods.

WebE Methods

WebE methods can be categorized into:

- **Communication methods:** facilitate communication between Web engineers and stakeholders.
- **Requirements analysis methods:** for understanding the content to be delivered by a WebApp.
- **Design methods:** address WebApp content, application and information architecture, interface design and navigation structure.
- **Construction methods:** Apply languages, tools, and related technology to the creation of WebApp content and functionality.
- **Testing methods:** technical reviews of both the content and design model. Testing techniques address component-level and architectural issues, navigation testing, usability testing, security testing, and configuration testing.

What about Tools and Technology?

… tools and technology are very important, but they'll work well only if they're used within the context of an agile framework for Web engineering and in conjunction with proven methods for understanding the problem, designing a solution, and testing it thoroughly.

WebE Best Practices

- Take the time to understand business needs and product objectives, even if the details of the WebApp are vague.
- Describe how users will interact with WebApp using a scenario-based approach.
- *Always d*evelop a project plan, even if it's very brief.
- Spend some time modeling what it is that you're going to build.
- Review the models for consistency and quality.
- Use tools and technology that enable you to construct the system with as many reusable components as possible.
- Don't reinvent when you can reuse.
- Don't rely on early users to debug the WebApp—design and use comprehensive tests before releasing the system.