Topic 01

Software Engineering, Web Engineering, agile methodologies.

What is Software Engineering?
Classic Software Engineering

The IEEE definition:

Software Engineering is the application of a
disciplined,
systematic,
quantifiable approach
to the
development,
operation, and
maintenance
of software; that is, the application of engineering to software.

Software Engineering (a disciplined approach) – CS590
Classic SE: Waterfall Model

Classic SE: Spiral Evolutionary Model
Problems with Classic SE

1) Size of big SW projects: millions of LOC
2) Cost of big SW development project: millions of dollars ($)
3) Failure rate of big SW development projects: about 35% in 2010
4) 60% of developed SW functionality are NEVER or RARELY used
What is

WebApp Engineering (WebE)?

Web Engineering Framework Activities
- CS593

- Communication (initiation)
- Planning
- Modeling/Simulation/Prototyping
  - Analysis of requirements
  - Design Model
- Development (construction)
  - Code generation
  - Testing
- Deployment/Implementation

WebE 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.

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 Web development technologies, tools, applications, with 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.
WebE: Umbrella (Support) Activities

- Software project management (PM)
- Formal technical reviews (FTR)
- Risk management (RM)
- Software quality assurance (SQA)
- Measurement
- Software configuration management (SCM)
- Reusability management
- Work product preparation and production

Software project management (PM) (allows team to assess progress and take corrective action to maintain schedule)
Formal technical reviews (FTR) (assess engineering work products to uncover and remove errors before they propagate to next activity)
Risk management (RM) (assess risks that may affect project outcomes or quality)
Software quality assurance (SQA) (activities required to maintain software quality)
Measurement (define and collect process, project, and product measures to assist the software team in delivering software meeting customer needs)
Software configuration management / change management (SCM / change management) (manage effects of change)
Reusability management (defines criteria for work product reuse and establish mechanisms to achieve component reuse)
Work product preparation and production (activities to create models, documents, logs, forms, lists, etc.)

What is the main difference between Software Engineering and WebApp Engineering?
... and the answer is **Agility**  
(quickness, swiftness)

Main reasons for NOT applying SE principles to WebE:

- **Business strategies and rules change rapidly**

- Management demands near-instantaneous – **almost immediate reaction** --responsiveness (even when such demands are completely unreasonable).

- Stakeholders often don’t understand the consequences of the Web and keep changing their mind 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 a philosophy and a set of development guidelines. The philosophy encourages:
  - customer satisfaction (quality is OK if customer is satisfied)
  - informal methods (aimed to speed up SW development)
  - overall development simplicity
  - early incremental delivery of the WebApp
  - small, highly motivated project teams
  - minimal work products

- An agile process stresses delivery over analysis and design (although these activities are not discouraged), and active and continuous communication between developers and customers.
Classic Software Engineering vs Agile Software Development

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Agile (C2593 class-based approach)</th>
<th>Disciplined (C5950 class-based approach)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application</strong></td>
<td>Rapid value, responding to change</td>
<td>Predictability, stability, high assurance</td>
</tr>
<tr>
<td><strong>Team size</strong></td>
<td>Smaller teams (2-5 persons) and projects</td>
<td>Larger teams and projects</td>
</tr>
<tr>
<td><strong>Company size</strong></td>
<td>Start-up company, small-size company (up to 50 workers)</td>
<td>Middle-size and large-size companies (50+ workers)</td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td>Turbulent, high-change project-focused</td>
<td>Stable, low-change, project/organization focused</td>
</tr>
<tr>
<td><strong>Management</strong></td>
<td>Dedicated on-site customers; focused on prioritized increments</td>
<td>As-needed customer interactions; focused on contract provisions</td>
</tr>
<tr>
<td><strong>Planning</strong></td>
<td>Intermittent plans, qualitative general</td>
<td>Documented plans, quantitative control</td>
</tr>
<tr>
<td><strong>Communications</strong></td>
<td>Text interpersonal knowledge</td>
<td>Explicit documented knowledge</td>
</tr>
<tr>
<td><strong>Requirements</strong></td>
<td>Prioritized informal stories and test cases; undergoing unforeseeable change</td>
<td>Formulated project, capability, workflow, quality, foreseeable evolution requirements</td>
</tr>
<tr>
<td><strong>Development</strong></td>
<td>Simple design, short increment, reflecting assumptions and experiences</td>
<td>Extensive design, longer increments reflecting assumptions and experiences</td>
</tr>
<tr>
<td><strong>Test</strong></td>
<td>Execute test cases defined requirements, testing</td>
<td>Documented test plans and procedures</td>
</tr>
</tbody>
</table>

**Personnel**
- Customers: Distributed, colocated, CRACK* performers
- Developers: At least 30% full-time, Cocomo level 2 and 3 projects: no Level 18 or -1 personnel**

**Culture**
- Comfort and empowerment via many degrees of freedom (thinking on chairs)
- Comfort and empowerment via framework of policies and procedures (tulip on order)

---

Evolutionary Map of Agile Methods in Software Development

---

**Factors of successful methods**
- Evolutionary (Kitchen, 1994)
- Incremental development (Kleppe, 1984)
- Incremental refinement and realization (Kleppe, 1992)
- Modified waterfall model (Kleppe, 1999)
- Open source software development (Brach, 2000)
- Kaizen (Sato, 2001)
- Kanban (Sakai, 2001)
- Extreme programming (Klein, 2001)
- Scrum (Searle, 2002)
- Agile Modeling (Saly, 2003)

---

**Source:**
Types of Agile Software Development Methodologies

- Extreme Programming (XP)
- Industrial XP (IXP)
- Adaptive Software Development (ASD)
- Scrum
- Crystal programming
- Feature Driven Development (FDD)
- Lean Software Development (LSD)
- Dynamic Systems Development Method (DSDM)

Software Engineer vs WebApp Engineer
Software Engineer: Job #1 in the U.S.A.

WebApp Engineer vs Software Engineer

We've found 417 Web Software Engineer jobs!

We've found 397 Software Engineer jobs!
### CS593 Course Framework

<table>
<thead>
<tr>
<th>Course parts</th>
<th>Part title</th>
<th>Main topics</th>
</tr>
</thead>
</table>
| 1            | Web systems. Web systems engineering (WebE). | 1. Web systems characteristics  
2. Communication, analysis, design models  
3. Diagramming tools (Lab 01)  
4. Interaction model and GUI  
5. Information model |
| 2            | Web technologies for Web systems (including student research projects) | 1. Mash-Up technology  
2. Second Life Project technology  
3. Cloud computing  
4. Web services  
5. Web 2.0 technology and tools  
6. Streaming technology |
| 3            | Modern software development methodologies (including student research projects) | 1. Agile methodology  
2. RAD (rapid application development) methodology  
3. Scrum methodology  
4. XP (extreme programming) methodology  
5. Open-source software methodology  
6. RUP (rational unified process) methodology  
7. Crystal family of methodologies  
8. Pragmatic programming methodology |
| 4            | Mobile software systems | 1. Mobile Web  
2. Mobile software systems characteristics  
3. Design of mobile software systems  
4. IDE for mobile software development |

Course components of CS593 course: 1) lectures, 2) student research projects, 3) computer labs, 4) midterm – computer diagrams, 5) course project on Web or mobile application, 6) final exam (Web taped presentation)
Topic 01

Software Engineering and Web Engineering

Additional information.

Underlying Agility Principles - I

- 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.
- Deliver working software increments frequently, from as often as every few days to every few months, with a preference to the shorter timescales.
- Business people 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.
Underlying Agility Principles - II

- **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 architectures, 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
- Actions encompass
  - work tasks
  - work products
  - quality assurance points, and
  - project milestones
- A framework also has a set of “umbrella activities”
Umbrella Activities of WebE

1) SE rests on a goal - an organizational commitment to produce top quality product.
2) Process layer (that uses formal process models) is the glue that holds the SE technology layers together.
3) Methods to provide the concrete technical HOW-TO’s for building SW.
4) Tools to provide automated or semi-automated support for SW development.
What about Tools and Technology?

WebE 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 the WebApp using a scenario-based approach.

- Always develop 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.
WebE Methods

- Communication methods
- Requirements analysis methods
- Design methods
- Construction/development methods
- Testing methods

WebE Pages, Journals

- World Wide Web Consortium: http://www.w3.org
- International Society for Web Engineering: http://www.iswe-ev.de/