Where we are?

<table>
<thead>
<tr>
<th>#</th>
<th>Date</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5th March</td>
<td>Web Engineering Introduction and Overview</td>
</tr>
<tr>
<td>2</td>
<td>12th March</td>
<td>Requirements Engineering for Web Applications</td>
</tr>
<tr>
<td>3</td>
<td>19th March</td>
<td>Web Application Modelling</td>
</tr>
<tr>
<td>4</td>
<td>9th April</td>
<td>Web Application Architectures</td>
</tr>
<tr>
<td>5</td>
<td>16th April</td>
<td>Developing Applications with WebML</td>
</tr>
<tr>
<td>6</td>
<td>23rd April</td>
<td>Testing and Usability of Web Applications</td>
</tr>
<tr>
<td>7</td>
<td>30th April</td>
<td>Maintenance and Performance of Web Applications</td>
</tr>
<tr>
<td>8</td>
<td>7th May</td>
<td>Web Technologies I</td>
</tr>
<tr>
<td>9</td>
<td>14th May</td>
<td>Web Technologies II</td>
</tr>
<tr>
<td>10</td>
<td>21st May</td>
<td>Web Application Development Process</td>
</tr>
<tr>
<td>11</td>
<td>28th May</td>
<td>Project Management for Web Applications</td>
</tr>
<tr>
<td>12</td>
<td>4th June</td>
<td>Web Application Security</td>
</tr>
<tr>
<td>13</td>
<td>11th June</td>
<td>Mobile Application Development I</td>
</tr>
<tr>
<td>14</td>
<td>18th June</td>
<td>Mobile Application Development II</td>
</tr>
<tr>
<td>15</td>
<td>25th June</td>
<td>Final Exam</td>
</tr>
</tbody>
</table>
Overview

- Introduction
- Fundamentals of Testing on the Web
- Methods and techniques to Test Web Applications
- Automatic Web application Testing
- Fundamentals of Usability on the Web
- Summary

INTRODUCTION
Testing and Usability

• Testing
  – Checking the conformance of the application versus its design requirements
  – Oriented to functional aspects

• Usability
  – Designing and verifying the conformance of the application versus its user ability and capability of interaction
  – Oriented to non-functional aspects

• Testing and Usability are orthogonal aspects
  – In some cases they slightly overlap!

The Importance of Testing

• Traditionally, testing has focused on functional requirements – not enough for Web applications.

• On the Web, testing is a critical measure of quality assurance.
  – Meeting users’ expectations
  – Finding errors and shortcomings
  – Many users, many platforms
  – Behavior of third-party software
The Importance of Usability

- “Mission critical” Web applications
- Poor design leads to lost time, productivity
- Your website speaks for your organization
  - Customers have choices
  - Easy come, easy go
- Diverse contexts
  - Proliferation of web-enabled devices
  - Increasing adoption by special needs groups – ex. seniors

What are the peculiarity of testing Web applications?

FUNDAMENTALS OF TESTING ON THE WEB
Terminology

- Some definitions
  - **Testing**: An activity conducted to evaluate the quality of a product to improve it by identifying defects and problems.
  - **Error**: the actual result deviates from the expected.
    - Our expected results should (theoretically) come from our requirements definition.
    - Most often, the goals/concerns/expectations of stakeholders serve as the testing basis.
  - **Test case**: a set of inputs, execution conditions, and expected results for testing an object.

Testing Objectives

- Main objective: find errors, NOT show that none exist.
- Complete test coverage is impossible, so testing focuses on mitigating the largest risks.
  - Where’s the greatest potential for loss?
  - What are the sources of this risk?
- Start testing as early as possible – even with restricted resources and time.
Testing Levels

- Unit tests:
  - Testing the “atomic” units - classes, Web pages, etc. - independently. (Developer)

- Integration tests:
  - Test the interaction of units (Tester & Developer)

- System tests:
  - Testing the whole, integrated system (Dedicated team)

- Acceptance tests:
  - “Real-world” tests - testing under conditions that are as close to the “live” environment as possible (Client)

- Beta tests:
  - Informal, product-wide tests conducted by “friendly” users.

The Tester Role

- The ideal tester has a “destructive” attitude.

- Very difficult for developers to “destroy” their own work.

- However, Web projects focus heavily on unit tests, making them more prone to errors.

- Thus, some guidelines:
  - Have others in the Web team perform tests.
  - Best tester is the one who gets the most bugs fixed.
Web Engineering Specifics 1

- Errors in Web content
  - Found mainly through proofreading - very costly
  - Alternative tests: Spell-checking, meta-information

- Hypertext structure
  - Is each page accessible via a link?
  - Does each page link to the hypertext structure?
  - Are there any broken links?
  - What happens when the user hits “Back” in their browser?

Web Engineering Specifics 2

- Subjective requirements for presentation
  - Often in the eye of the beholder (e.g., aesthetics).
  - Tester must distinguish accepted behavior from faulty.
  - Presentation testing on the Web borrows from print publishing.

- Multi-platform delivery
  - Can you test on every device?
  - Can you create test cases on every device?
  - Simulators are frequently available, but may be buggy.
Web Engineering Specifics 3

• Global availability
  – Testing dynamic content in multiple languages
  – Testing for layout difficulties due to varying text lengths.

• Juvenility & Multidisciplinarity of Web team
  – Reluctance to accept testing methods.
  – Lack of testing knowledge.
  – Consensus-building is required.
  – May do too much testing; just as bad as too little.

Web Engineering Specifics 4

• Multiple System Components
  – Third-party; different platforms.
  – Testing of the components’ integration and configuration is also required.

• Immaturity of test methods
  – Suitable test suites for new technologies often don’t exist, or are poorly designed.

• Continuous change
  – Requirements, hardware, software changes.
  – Retest following each major upgrade.
How can I test my Web application?

METHODS AND TECHNIQUES TO TEST WEB APPLICATIONS

Link Testing

• Finding broken links
  – Can be automated through a spider
  – Doesn’t help for pages with no incoming links.

• Finding orphan pages
  – Orphans are pages with no links back to the navigation structure.
  – Users get frustrated and leave.

• Capturing statistics
  – Depth & breadth of navigation.
  – Distance between two related pages.
  – # of links.
  – Load times.
Browser Testing

- Browsers vary by:
  - Manufacturer
  - Version
  - Operating system
  - Device
  - Configuration (stylesheets, JavaScript on/off)
  - W3C Standard compliance

- Important questions to ask:
  - How is state managed?
  - Can a (dynamic) web page be bookmarked?
  - Can users open multiple windows?
  - What happens when cookies and/or scripting is turned off?

Load Testing

- Does the system meet required response times and throughput?

- Load profile - expected access types, visits per day, transaction types, transactions per session, etc.

- Must determine the range of values for response times and throughput.

- Evaluate the results to look for bottlenecks.
Stress Testing

• How does the system behave under abnormal/extreme conditions?

• The test should tell you…
  – If the system meets the target responses times and throughputs
  – If the system responds with an appropriate error message. (i.e. graceful degradation)
  – If the system crashes (it should NOT!)
  – How quickly the system recovers to normal operation.

Continuous Testing

• Simulates usage over a long period of time

• Testing for errors that “pop up” because resources aren’t released by an operation.
  – Unreleased database connections
  – Other memory leaks

• Typically, running the operation a few times doesn’t produce an error, hence the need for continuous testing.
Security Testing

- A systematic test scheme is strongly encouraged.
- Testing for correctness is not sufficient
  - Is confidential data inadvertently exposed?
  - What happens if we input incomplete data?
  - What happens if we inject malicious code?
  - SSL-encrypted pages
    - Is our SSL certificate working?
    - What happens if I try to access a protected page/site in a non-secure way (i.e., http://)?

![XKCD Comic](http://imgs.xkcd.com/comics/exploits_of_a_mom.png)

Test-Driven Development

- Inspired by the test-first approach used in XP (eXtreme Programming); can be used in any type of project.
- Tests must be written before implementation.
  - Every unit has a test.
  - When a test fails, the developer must only change the code to successfully run the test.
- Developers can concentrate on small steps, while still making clean code that works.
- More pressure leads to more testing.
How can we reduce the cost of Web application testing?

**AUTOMATIC WEB APPLICATION TESTING**

**JUnit Overview**

- Open source Java testing framework used to write and run repeatable automated tests
- A structure for writing test drivers
- JUnit features include:
  - Assertions for testing expected results
  - Test features for sharing common test data
  - Test suites for easily organizing and running tests
  - Graphical and textual test runners
- JUnit is widely used in industry
- JUnit can be used as stand alone Java programs (from the command line) or within an IDE such as Eclipse
Cactus Overview

- Built on JUnit framework
- Intended to test JSP, Servlets, EJBs, Filters, and custom tags
- Complex architecture that has client JVM call the J2EE application server JVM
- Testcase classes must reside on client and server

Other JUnit Extensions

- HttpUnit
  - Parses HTML results into DOM
  - Easy link navigation and form population
  - Useful for automated acceptance tests
- Canoo WebTest
  - HttpUnit inside Ant
- JUnitPerf
  - Wrap any JUnit tests
  - Measure desired performance and scalability tolerances
Automating Testing - Advantages

• Some tests are impossible to perform manually.
  – Load & stress tests.
  – Link testing for large websites.

• More tests can be run in less time.

• When updating an application, can detect errors caused by side-effects to unchanged functionality.

Automating Testing - Disadvantages

• Expectations of automated testing is often too high.
  – Automation does NOT improve effectiveness.
  – If tests are poorly devised, automating them does not magically improve them.

• Automation is expensive
  – Test execution infrastructure must be maintained.
  – License fees & training costs
How to design usable Web applications?

FUNDAMENTALS OF USABILITY ON THE WEB

Usability Defined

• ISO/IEC standard definition (1998):
  – “The extent to which a product can be used by specified users within a specified usage context to achieve specified goals effectively, efficiently, and satisfactorily.”

• Usability engineering is an ongoing, but critical process
  – Define user and task models
  – Iteratively test and reevaluate
  – User-based vs. expert methods
Defining Usability in Web Applications

• Traditional software usability specifics do not necessarily carry over to the Web:
  – People use your application immediately.
  – No manual or trainers.
  – No salespeople.

• How to categorize users?
  – First-time or returning?
  – Expert or novice?
  – Broadband or dial-up?
  – Desktop or mobile?

Top Problems

• Contact information – address or phone number is buried
• Search function is not visible or unclear as to functionality
• No easy way to get back to critical points
• Pages that should load fast don’t (e.g. main page or key link page)
• “What’s new” is old
• Back button requires a repost of data
Usability Engineering

• Consists of 4 phases that are essentially parallel to the Web Engineering process

User-Centered vs. Usage-Centered

<table>
<thead>
<tr>
<th>Phase</th>
<th>Focal Points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User-Centered (Traditional)</td>
</tr>
<tr>
<td>Requirements</td>
<td>Meetings, interviews, focus groups</td>
</tr>
<tr>
<td>Design &amp; Implementation</td>
<td>User requirements</td>
</tr>
<tr>
<td></td>
<td>Direct user participation</td>
</tr>
<tr>
<td>Operation</td>
<td>Training, evaluation of help-desk logs</td>
</tr>
</tbody>
</table>
Requirements Analysis

• Systems Analysts & Usability Experts take the lead:
  – Competitive Analysis
  – Define qualitative/quantitative goals
    • Information, Entertainment, Exchange (Siegel)
    • Make them concrete and testable!
  – User-centered: build user profiles
  – Usage-centered
    • Task analysis
    • Ease-of-use or Ease-of-learning?

Interaction and Design

• Initially, the Interface Designer builds a conceptual model
  – Presentation: Storyboards & Paper mock-ups
  – Navigation: Card-sorting
  – Based on core use cases
  – Shows the basic structure

• Getting feedback from potential users

• Usability expert provides input after this first round.
Interaction and Design

• Designer and coders can then elaborate on the details

• Additional user testing:
  – Prototypes – exhibit some functionality
  – Usability Tests – real context, real tasks.

• Remote usability testing
  – Sample of representative users
  – Client-Logging software
  – Web-cams if possible
  – Better external validity & lower costs(?)

Coding and Post-Deployment

• Usability Expert assumes the role of the Quality Assurance manager.
  – Consistency?
  – Observed guidelines & standards?
  – Adhered to (current) requirements?

• Bring same users back in for testing, if possible.

• Document, document, document!
General Design Guidelines

- Design guidelines represent best practices
- OK for “general” users
  - Normal cognitive ability
  - Normal audiovisual abilities
- Some guidelines may be inappropriate for audience members with special needs.
  - Ex. Navigation elements for schizophrenics
- More rigorous usability engineering techniques (just discussed) should be employed

Human Information Processing

- Human cognition plays a critical role in user interface design.
  - Perception
    - Positioning, grouping, arranging
    - Perceiving shapes and relationships
  - Memory
    - Limitations of working memory
    - Chunking, 7 + 2 (Miller)
  - Attention
    - Focusing on one aspect
    - Movement, color schemes
Guidelines – Response Times

- As response times increase, user satisfaction decreases
  - Anything greater than 3 seconds, and the user becomes aware she’s waiting
  - After 10 seconds, user gives up
- Optimize, or minimize graphics
- Consider breaking up large pages.
- `<img>` - use “width” & “height” attributes
- Don’t forget your dial-up audience!
  - Home page size should be < 50Kb
  - Provide warnings (MPG – 2.5Mbs)

Guidelines – Efficiency

- Minimize distance between clickable elements (while keeping effective sizing)
- Avoid frequent changes between mouse and keyboard
- Tab-friendly for text-based browsers
- Minimize clicks to accomplish tasks (rule of thumb: no more than 4 clicks)
Guidelines – Colors

- Colors have different meaning depending on your audience
  - Cultural differences
  - Domain-specific meanings
  - Warm vs. cool colors

- Make sure all information conveyed by colors is also available without color.

- Minimize the number of colors

- Avoid extreme hues, highly saturated colors

- How does your site look on an LCD? CRT?

Guidelines – Text Layout

- Screen vs. Paper

- Consider different window sizes
  - Avoid fixed width layouts
  - Avoid multiple columns (typically)

- Readability
  - Sans-serif for screen, serif for print
  - Avoid patterns, low-contrast background
  - Short paragraphs

- Allow for user-selected font-sizes
Guidelines – Page Structure

- Display considerations
- Use relative positioning over absolute.
- Vertical scrolling is fine; horizontal scrolling is NOT.
- Important elements should ALWAYS be visible.
- Make page print-friendly or provide alternative style & print button.

Guidelines – Navigation

- Provide your user with a model of the site
  - Intuitive navigation elements
  - Site map
  - Breadcrumbs
- Dropdown menus
  - Pros: Efficient use of space
  - Cons: Key information is hidden
Guidelines – Multicultural

• Location is typically not a constraint on the Web.

• “Smallest common cultural denominator”:
  – Avoid over-expressive colors
  – Symbols
  – Language
  – Information representation (date/time formats)

• Present form elements consistently

Guidelines – Consistency

• Consistency keeps learning to a minimum; users don’t want to have to think!

• Identity can be set by consistent components
  – Header: home, logo, navigation, search, help
  – Footer: author, modification, contact

• Consistent design helps users avoid getting lost, especially when jumping to different sub-units of an organization.
More on Web Accessibility

- People with disabilities are adopting the Web in greater numbers.
- Tim Berners-Lee stressed universal access to the Web as essential.
- 20% of the world’s population have disabilities in at least one of the senses.
- Key takeaways:
  - Designing for special needs doesn’t necessarily require reinventing your application.
  - Doing so can also help “general” users

Web Accessibility Initiative (WAI)

- Web Content Accessibility Guidelines 2.0 (WCAG, 2008) published by the W3C's WAI
- 12 Guidelines
- 4 Groups
  1) Perceivable
  2) Operable
  3) Understandable
  4) Robust
- Defines Special Needs Groups
- Conformance Levels (A, AA, AAA)
That's almost all for day...

WRAP-UP

Things to keep in mind
(or summary)

- Testing and Usability are orthogonal issues
  - Testing covers functional requirements validation
  - Usability tends to cover “non-functional” requirements validation (this is not totally exact)

- Both are part of a good (Web) application design process

- They can be partially automated

- Key differences
  - Testing deals mainly with in house and (rarely) external developers and end users
  - Usability relies on expertise outside “technological world”
Bibliography

• Mandatory reading

• Other References
  – Kent Beck, eXtreme Programming Explained
  – http://www.usability.gov/

Testing Tools

• JUnit.org
  – http://www.junit.org
• Cactus
  – http://jakarta.apache.org/cactus
• Clover
  – http://www.thecortex.net/clover
• dbUnit
  – http://www.dbunit.org
• HttpUnit
  – http://www.httpunit.org
• Canoo WebTest
  – http://webtest.canoo.com
Usability Tools

- Development
  - Firefox Developer Toolbar (http://chrispederick.com/work/web-developer/)

- Testing
  - http://webusability.com/usability_tools.htm
  - http://wave.webaim.org/

- Comprehensive list of WAI tools
  - http://www.w3.org/WAI/ER/tools/complete

Next Lecture

<table>
<thead>
<tr>
<th>#</th>
<th>Date</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5th March</td>
<td>Web Engineering Introduction and Overview</td>
</tr>
<tr>
<td>2</td>
<td>12th March</td>
<td>Requirements Engineering for Web Applications</td>
</tr>
<tr>
<td>3</td>
<td>19th March</td>
<td>Web Application Modelling</td>
</tr>
<tr>
<td>4</td>
<td>9th April</td>
<td>Web Application Architectures</td>
</tr>
<tr>
<td>5</td>
<td>16th April</td>
<td>Developing Applications with WebML</td>
</tr>
<tr>
<td>6</td>
<td>23rd April</td>
<td>Testing and Usability of Web Applications</td>
</tr>
<tr>
<td>7</td>
<td>30th April</td>
<td>Maintenance and Performance of Web Applications</td>
</tr>
<tr>
<td>8</td>
<td>7th May</td>
<td>Web Technologies I</td>
</tr>
<tr>
<td>9</td>
<td>14th May</td>
<td>Web Technologies II</td>
</tr>
<tr>
<td>10</td>
<td>21st May</td>
<td>Web Application Development Process</td>
</tr>
<tr>
<td>11</td>
<td>28th May</td>
<td>Project Management for Web Applications</td>
</tr>
<tr>
<td>12</td>
<td>4th June</td>
<td>Web Application Security</td>
</tr>
<tr>
<td>13</td>
<td>11th June</td>
<td>Mobile Application Development I</td>
</tr>
<tr>
<td>14</td>
<td>18th June</td>
<td>Mobile Application Development II</td>
</tr>
<tr>
<td>14</td>
<td>25th June</td>
<td>Final Exam</td>
</tr>
</tbody>
</table>