1.264 Lecture 1

Course introduction
Software process
Course outline

• Lecture Tuesday-Thursday 1-2:30pm
• Recitation Tuesday 4-5pm or Wednesday 3-4pm
• Staff:
  George Kocur
  Loai Naamani
  Darris Hupp
• Office hours
  George Kocur Thu 2:30-3:30pm or by appt
  TAs See MIT Server; may vary by week
• No prerequisite; familiarity with PCs, Windows, Word assumed
• Grading:
  10 homework sets (40%)
  Midterm (22%)
  Final exam (25%)
  Recitation participation/attendance (5%)
  Active learning (8%)
• Bring your laptop to class starting with lecture 5: exercises.
  Work with partner or by yourself
Topics

• **Software development**: rapid development methods
  Develop, configure or manage software
• **Data modeling and databases**
  Ensure correctness of data; allow sharing, flexibility
• **Web development**
  HTTP, XHTML, CSS, XML, Java, .NET, security. Integration
• **Middleware**
  System interfaces, connections between databases and applications: XML, EDI, WSDL, SOAP, …
• **Security**
  Encryption, certificates, SSL, implementation
• **Communications networks and hardware**
  Technologies, protocols, standards: data, video, voice
Homework (project)

• Work in teams of two (1 and 3 allowed by exception)
  – Choose your partner this week. Ask TA to match you if you don’t find a partner.
• Build a set of systems for a chemical distributor
  – First cycle of ‘spiral model’ of software development
  – Take 3 months to specify, design, prototype and assess
    And learn about all these technologies
  – After this first cycle, you could then build an operational system
    Even your prototype would almost be ok for a very small scale operation
1. Software process case studies
2. Requirements narrative
3. UML models (Visual Paradigm)
4. Data model (Visual Paradigm)
5. Database (MS SQL Server)
6. Web initial pages (Dreamweaver)
7. XML and Web catalog pages, XML (Dreamweaver, SQL Svr)
8. Web order entry (Dreamweaver, SQL Svr)
9. Security and Web login (Dreamweaver, SQL Svr)
10. Communications; software process retrospective
Readings

• This class is half reading and half homework sets.
  McConnell *Rapid Development*
  Fowler *UML Distilled*
  Murach *SQL for SQL Server* or Bowman *SQL Handbook*
  McFarland *Dreamweaver 8*
  Anderson *Security Engineering*
  Green *Handbook of Telecom*

• Share books with your teammate

• Successful projects involving software development or management rely on making no major mistakes
  You don’t have to do anything perfectly or optimally but you can’t make any major mistakes.
  We cover many topics, to make sure you’ve seen each topic at least once
  If you wind up working with software in your career (and most of you will), these books are standard texts and references on other topics you may need…
Computer systems

• Use your own laptop or desktop computer
  All software available for download
  Either open source or under MIT license
  Recitation 1 will help with installation and initial usage
  If you don’t have a laptop, sit next to someone who does at class
• 1.264 site on Stellar
  Lecture notes, posted after lecture (PowerPoint slides)
  Homework and online readings
  Announcements
• Web and database servers will be used for the Web sites
  IIS and SQL Server
A quick quiz

What percentage of large projects have excess schedule pressure?
- 25%
- 50%
- 75%
- 100%

What percentage of small projects have excess schedule pressure?
- 25%
- 50%
- 75%
- 100%

What percentage of large projects deliver on time and on budget?
- 25%
- 50%
- 75%
- 100%

What percentage of large projects are cancelled or fail to deliver at all?
- 25%
- 50%
- 75%
- 100%

What staff increase is necessary to speed up a schedule by 25%?
- 25%
- 50%
- 75%
- 100%

How much are resource needs reduced by cutting project scope in half?
- 25%
- 50%
- 75%
- 100%

How much have companies reduced time to market through better software practices in the last 10 years?
- 25%
- 50%
- 75%
- 100%
Answers to a quick quiz

What percentage of large projects have excess schedule pressure?
- 25%
- 50%
- 75%
- 100%

What percentage of small projects have excess schedule pressure?
- 25%
- 50%
- 75%
- 100%

What percentage of large projects deliver on time and on budget?
- 25%
- 50%
- 75%
- 100%

What percentage of large projects are cancelled or fail to deliver at all?
- 25%
- 50%
- 75%
- 100%

What staff increase is necessary to speed up a schedule by 25%?
- 25%
- 50%
- 75%
- 100%

How much are resource needs reduced by cutting project scope in half?
- 25%
- 50%
- 75%
- 100%

How much have companies reduced time to market through better software practices in the last 10 years?
- 25%
- 50%
- 75%
- 100%
Software development process

- Software development is often more demanding than consulting or analysis
  - Software process has applications and lessons for project management more generally
  - Software cannot be built the night before, like (bad) reports
  - Software can’t be downscoped at the last minute, with chapters or analyses simply left out, or done very simply
- Case studies in McConnell (Rapid Development) will be our focus
  - Software development process is otherwise a very boring topic
What are the four dimensions of development speed?

• Key factors that determine how well and how quickly you will develop, configure, implement or manage a project with a software component:
  1. ______________________
  2. ______________________
  3. ______________________
  4. ______________________
Dimensions of development speed

- **People**
  - Matter most: ability and motivation

- **Process**
  - Customer focus
  - Fundamentals, QA, risk management, lifecycle planning
  - “Code like hell” and chaos are still the most common approaches

- **Product**
  - Size and characteristics, phasing

- **Technology**
  - Product or software development environment
  - Tools
Use of good development, implementation or management practices

Figure by MIT OCW.

People, process, product, technology
<table>
<thead>
<tr>
<th>People-Related</th>
<th>Process-Related</th>
<th>Product-Related</th>
<th>Technology-Related</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heroics</td>
<td>Contractor failure</td>
<td>Feature creep</td>
<td>Silver-bullet syndrome</td>
</tr>
<tr>
<td>Weak personnel</td>
<td>Inadequate design</td>
<td>Developer gold-plating</td>
<td>Lack of automated source-code</td>
</tr>
<tr>
<td>Wishful thinking</td>
<td>Insufficient planning</td>
<td>Requirements gold-plating</td>
<td>Switching tools in the middle of a project</td>
</tr>
<tr>
<td>Lack of user input</td>
<td>Planning to catch up later</td>
<td>Push-me, pull-me negotiation</td>
<td>Overestimated savings from new tools or methods</td>
</tr>
<tr>
<td>Undermined motivation</td>
<td>Overly optimistic schedules</td>
<td>Research-oriented development</td>
<td></td>
</tr>
<tr>
<td>Noisy, crowded offices</td>
<td>Code-like-hell programming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unrealistic expectations</td>
<td>Insufficient risk management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lack of stakeholder buy-in</td>
<td>Abandonment of planning under pressure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Politics placed over substance</td>
<td>Shortchanged upstream activities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adding people to a late project</td>
<td>Insufficient management controls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lack of effective project sponsorship</td>
<td>Wasted time during the fuzzy front end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Friction between developers and customers</td>
<td>Premature or overly frequent convergence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncontrolled problem employees</td>
<td>Omitting necessary tasks from estimates</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shortchanged quality assurance</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure by MIT OCW.
Technical fundamentals

Spiral model as basis for development

Figure by MIT OCW.
For next class

• Read Mc Connell chapters 1-5
• Read case studies posted on course Web site
• Be prepared to discuss them in class
• Upload your case study answers to the course Web site next Tuesday
  – Preferably before class, but no later than 5pm
  – Graded only on whether you did it, not on right/wrong