Empirical Software Engineering in Industry Short Courses

David Janzen, Clark Turner
California Polytechnic State University
San Luis Obispo, CA

Hossein Saiedian
University of Kansas
Lawrence, KS
Outline

• EBSE Background
• Pilot Experience
• Survey Results
• Challenges and Suggestions
• Late-Breaking News
• Conclusions
Evidence-Based Software Engineering (EBSE)

• Promises:
  – Demonstrate the efficacy of software practices, tools, and methods
  – Inform adoption decisions

• Tactics:
  – Controlled experiments
    • Laboratory (often with students)
    • Field (professionals in familiar/usual domain)
  – Case studies
  – Surveys (review existing studies and experiments)
EBSE Challenges

- Threats to validity with academic studies
  - Students less mature than professionals
  - Contrived application domains
  - Smaller project and (sometimes) team size

- Access to industry
  - Reluctance to participate in controlled experiments (IP/NDA, reputation, productivity)
  - Few want to try “bleeding edge” tools/practices
  - Ignorance (EBSE results, techniques, opportunities)
Opportunity

• Many professionals acquire new skills through Industry Short Courses

• Proposal:
  – Introduce EBSE techniques and results by conducting small experiments in Industry Short Courses
Expected Benefits

• Education on EBSE techniques and results
  – Goal is not to train researchers, but to inform practitioners of EBSE benefits and results
• Opportunities to conduct experiments
  – Lab experiments and surveys in training courses
  – Future field experiments in professional domain
• Teach analytical thinking
  – Consider alternatives (admits “no silver bullets”)
  – Compare EBSE results with personal experience
Pilot Experience

• Conducted quasi-controlled experiments in three industry short courses
  – On-site courses for two Fortune 500 companies

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Timeframe</th>
<th>Students</th>
<th>Assignment</th>
<th>Pairs/ Solo</th>
<th>Submissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD in Java</td>
<td>Fall 2005</td>
<td>15</td>
<td>Bowling</td>
<td>Both</td>
<td>3</td>
</tr>
<tr>
<td>C++ for C Programmers</td>
<td>Summer 2005</td>
<td>14</td>
<td>Bowling</td>
<td>Solo</td>
<td>2</td>
</tr>
<tr>
<td>TDD in Java</td>
<td>Fall 2006</td>
<td>14</td>
<td>Bowling</td>
<td>Pairs</td>
<td>2</td>
</tr>
<tr>
<td>TDD in Java</td>
<td>Fall 2006</td>
<td>14</td>
<td>ToDo List</td>
<td>Pairs</td>
<td>4</td>
</tr>
</tbody>
</table>

– TDD in Java (2 day)
– C++ for C Programmers (4 day with ½ day on day 4 devoted to TDD)
Context

• Series of leveled studies to compare test-first (TDD) and test-last approaches
• Hypothesis:
  – TDD improves internal software quality
    • Complexity, size, coupling, cohesion, testability
• Leveled:
  – CS1, CS2, Undergrad SE, Grad SE, Industry
This paper focuses on training experience
Ethical Considerations

• support the IEEE/ACM Software Engineering Code of Ethics section 7.02
  - “[a]ssist colleagues in professional development”

• Belmont Report states that “[a]pplications of the general principles to the conduct of research leads to consideration of the following requirements:
  - informed consent, risk/benefit assessment, and the selection of subjects of research.”
  - Get approval from management in writing beforehand, get verbal or written approval from participants, allow them to not participate
Results of Study

• Caveat:
  – Results are of limited value due to common threats to validity
    • Small sample size (14 or 15 per study)
    • Short project duration (2-3 hours)
    • Immaturity with concepts (learning TDD)
Test Coverage

- Test-first programmers tended to achieve higher test coverage
- Differences were larger in larger projects

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Exercise</th>
<th>Approach</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD Summer 2006</td>
<td>Bowling</td>
<td>TF</td>
<td>81% 79%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>Bowling</td>
<td>TF</td>
<td>45% 0%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>Bowling</td>
<td>TL</td>
<td>91% 87%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>Bowling</td>
<td>TL</td>
<td>0% 0%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>ToDo</td>
<td>TF</td>
<td>100% 100%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>ToDo</td>
<td>TF</td>
<td>100% 100%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>ToDo</td>
<td>TF</td>
<td>88% 75%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>ToDo</td>
<td>TF</td>
<td>86% 76%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TF</td>
<td>50% 19%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TF</td>
<td>58% 55%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TF</td>
<td>49% 30%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TL</td>
<td>68% 63%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TL</td>
<td>73% 80%</td>
</tr>
<tr>
<td>TDD Fall 2005</td>
<td>Bowling</td>
<td>TL</td>
<td>6% 0%</td>
</tr>
<tr>
<td>Average TF</td>
<td></td>
<td></td>
<td>73% 59%</td>
</tr>
<tr>
<td>Average TL</td>
<td></td>
<td></td>
<td>60% 58%</td>
</tr>
</tbody>
</table>
Survey Results 1

• Difference from Pre-experiment survey to Post-experiment survey
  – Importance of unit testing (Attitude)
  – Timing of writing unit tests (Timing)
• Higher means they prefer earlier testing in process

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Direction</th>
<th>Attitude</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD Fall 2005</td>
<td>%Increasing</td>
<td>62%</td>
<td>50%</td>
</tr>
<tr>
<td></td>
<td>%Decreasing</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>C++ Summer 2005</td>
<td>%Increasing</td>
<td>0%</td>
<td>88%</td>
</tr>
<tr>
<td></td>
<td>%Decreasing</td>
<td>13%</td>
<td>0%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>%Increasing</td>
<td>9%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>%Decreasing</td>
<td>9%</td>
<td>0%</td>
</tr>
</tbody>
</table>
Survey Results 2

• Choice of test-first or test-last programming (Choice)

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Choice</th>
<th>%Pre</th>
<th>%Post</th>
<th>%Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD Fall 2005</td>
<td>Test-First</td>
<td>67%</td>
<td>83%</td>
<td>17%</td>
</tr>
<tr>
<td></td>
<td>Test-Last</td>
<td>33%</td>
<td>17%</td>
<td>-17%</td>
</tr>
<tr>
<td>C++ Summer 2005</td>
<td>Test-First</td>
<td>29%</td>
<td>33%</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>Test-Last</td>
<td>71%</td>
<td>67%</td>
<td>-5%</td>
</tr>
<tr>
<td>TDD Summer 2006</td>
<td>Test-First</td>
<td>60%</td>
<td>82%</td>
<td>22%</td>
</tr>
<tr>
<td></td>
<td>Test-Last</td>
<td>40%</td>
<td>18%</td>
<td>-22%</td>
</tr>
</tbody>
</table>

Reluctance to adopt test-first by C programmers in C++ class
Challenges 1

- Difficult to calculate results quickly for immediate feedback
  - Used results from previous studies
  - Training did include test coverage tools so students reported their own test coverage results

- Short time frame
  - Difficult to create short yet meaningful exercises
Challenges 2

- Student reluctance to submit projects
  - Desire to master new concepts before showing others
- Integration with course concepts
  - Companies are paying you for training, not to conduct research
Suggestions

• Get approval from management in writing beforehand
• Get verbal approval from participants
• Allow trainees to not participate
• Identify trainee submissions with numbers
• Minimize “extra” time for conducting study
• Allow everyone to apply both(all) approaches/tools/methods
Late-Breaking News 1

• Conducted empirical study in middle-quarter of three quarter (nine-month) software engineering capstone
• Assigned students to perform study analysis in third quarter
• Results: students lacked enthusiasm about calculating and analyzing metrics, but gained appreciation for empirical approach
Late-Breaking News 2

• Met with representatives from both companies two weeks ago
• Both indicated desire to collaborate
• One indicated interest in participating in a new, larger empirical study
Conclusions

• Conducting empirical studies in industry short courses may:
  – Increase awareness of EBSE techniques and results
  – Increase opportunities for field experiments
  – Yield evidence on attitudes and learning curves
  – Yield supporting evidence for larger studies
Questions?