Research Presentation: Empirical Software Engineering

Gursimran S. Walia, North Dakota State University

April 22, 2013
Research Areas:

- Software engineering, particularly
  - Software quality improvement and measurement through software inspections and software errors
  - Software engineering education


- Multidisciplinary
  - Cognitive Psychology, Social Psychology, Statistics, and Educational Psychology
Major Current Research Topics

- **Integrating Software Engineering and Cognitive Error Models**
- **Using the Capture-Recapture to Measure Software Quality**
- **Other SE Research**
  - Using Gestalt Principles from Psychology in Software Engineering
  - Mental Model of Software Developers
  - Enhancing Creativity of Software Developers
  - Learning Styles of Computer Science Students
- **Software Engineering Education**
  - Pair Programming and Mental Model Evaluation of Computer Science Students at NDSU
  - Identifying and evaluating the *Knowledge* and *Skill-Deficiencies* among graduating CS students
  - Learning Styles of Computer Science Students at NDSU
Integrating Software Engineering and Cognitive Error Models

- My research employs the tried-and-true perspective of human error to address a serious problem in software engineering: defects made during software development.
- A medical analogy may help illuminate the differences between faults and errors.
- Application of human error research should also be relevant for reducing defects and improving the quality of software.
There is evidence....

- Our results provides evidence that use of structured error information during a requirements inspection allows inspectors to be more effective than use of only fault information.
- Other researchers employed our structured error information in a field-study to analyze and classify the errors found in space software requirement specifications.
Work done so far:

- Preliminary Classification of Requirement Errors
- Evaluation of the Usefulness of Structured Error Information for Improving Software Quality
- Fault Detection
- Fault Prevention
- Tool Developed for automating the error-based inspection
Research In progress

1. Develop Requirement Error Taxonomy
   - Requirement Error Classification
   - Interactive Workshop
   - Human Error Expert
   - Validate Error Taxonomy

2. Develop Error-Based Techniques
   - Prevention
   - Detection
   - Before Inspection
   - During Inspection
   - Validate Techniques

3. Develop Tools
   - Defect Prevention Tools
   - Defect Detection Tools
   - Validate Tools

Using the Capture-Recapture to Measure Software Quality

- My research validated the use of the Capture-Recapture method (originally developed by biologists) to support the defect size estimates of software artifacts.
Capture-Recapture Models

Model Assumptions:
1. Closed Population
2. Marks are not Lost
3. Equal Capture Probability

Inspections
1. Inspectors can have different abilities
2. Defects can have different difficulties
Research Results: 1

The *Effect of the Number of Inspectors* on the Defect Estimates Produced by Capture-Recapture Models

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Defect Type</th>
<th>Number of Inspectors</th>
<th>Inspector Type</th>
<th>Developer Type</th>
<th>First Inspection Defects</th>
<th>Total Number of Defects</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Seeded</td>
<td>73</td>
<td>Professional</td>
<td>Professional</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>8</td>
<td>Students</td>
<td></td>
<td>30</td>
<td>55</td>
</tr>
<tr>
<td>3</td>
<td>Natural Occurring</td>
<td>8</td>
<td></td>
<td>Students</td>
<td>46</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td>Defects</td>
<td></td>
<td></td>
<td></td>
<td>52</td>
<td>94</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>17</td>
<td></td>
<td>Professional</td>
<td>169</td>
<td>253</td>
</tr>
</tbody>
</table>
Evaluation Procedure:
1. Creating Virtual Teams

Create virtual teams of each size (1-73)

73 Inspectors
Results: Data set 1: *Median Estimates*

The graph shows the median relative error in the CR estimate as a function of inspection team size. The x-axis represents the inspection team size, while the y-axis shows the median relative error. The graph includes multiple lines, each representing a different dataset or condition, with error bars indicating variability. The horizontal dashed lines represent error thresholds, with the top line at 0% error and the bottom line at -100% error.
Adding more number of inspectors increases the cost and can impact the investment decision.
Metric to evaluate cost effectiveness of software Inspections using CR Methods

Kusumoto Metric ($M_k$) :

\[ M_k = \frac{\text{Reduction of total costs to detect all faults}}{\text{Virtual testing cost}} \]

\[ M_k = \frac{\Delta C_t - C_r}{C_t + \Delta C_t} \]
Calculation of parameters

- $C_r$ – Cost spent on Inspections
  
  $$C_r = \sum T_i$$

- $C_t$ – Testing Cost
  
  $$C_t = (D_{total} - D_r) \times c_t$$

- $\Delta C_t$ – Cost saved by inspections
  
  $$\Delta C_t = D_r \times c_t$$

- Virtual testing cost $C_{vt} = (C_t + \Delta C_t)$

- Reduction of the total costs $= (\Delta C_t - C_r)$
Data Set

73 Inspectors
Results: *Inspection Team Size vs. Cost-Effectiveness*

Cut off points for varying levels of Cost Savings

<table>
<thead>
<tr>
<th>Cost Savings</th>
<th>&gt; 40%</th>
<th>&gt;35%</th>
<th>&gt;30%</th>
<th>&gt;25%</th>
<th>&gt;20%</th>
<th>&gt;15%</th>
<th>&gt;10%</th>
<th>&gt;5%</th>
<th>&gt;0%</th>
<th>&lt; 0%</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Inspector Count</em></td>
<td>Never</td>
<td>9 to 17</td>
<td>7 to 22</td>
<td>5 to 23</td>
<td>5 to 26</td>
<td>5 to 29</td>
<td>3 to 31</td>
<td>1 to 33</td>
<td>1 to 35</td>
<td>37 to 73</td>
</tr>
</tbody>
</table>

Maximum Cost savings

No savings
Results:

Relative Error (R.E) percentage in the median $M_k$ values produced by each CR estimator
Discussion of Results

- Early Inspection saves cost
- Positive savings with even one inspection
- The increase in cost savings is positively correlated with the number of inspectors
- Results provide information about the minimum number of inspectors to be used for achieving accurate cost-effectiveness results.
Software Engineering Education Research

- Validated the use of Pair Programming with CS 160 and CS 161 courses at NDSU.
- Investigating Student-Instructor Interactions When Using Pair Programming.
- Evaluating the GAP BETWEEN THE SKILLS AND ABILITIES OF GRADUATING COMPUTER SCIENCE STUDENTS AND THE EXPECTATION OF INDUSTRY.
- Evaluating Mental Model Consistency of Introductory Computer Science Programming Students at NDSU.
EVALUATING THE GAP BETWEEN THE SKILLS AND ABILITIES OF GRADUATING COMPUTER SCIENCE STUDENTS AND THE EXPECTATION OF INDUSTRY

• **Motivation**
  - Graduating students not adequately prepared
  - Others have also identified issues

• **Knowledge Deficiency**
  - “Any skill, ability, or knowledge of concept which a recently graduated computer science student lacks based on employer expectations.”

• **Examples**
  - Object Orientation
  - Software Design
  - Test Coverage Tool
  - Collaboration Ability
Research Framework

- **Main Goal**

- **Three Main Aspects**
  - Systematic Literature Review
    - Establish a baseline for comparison
    - Provide understanding of existing research methods
  - Interviews with Industry Managers
    - Best source of information for knowledge deficiencies
    - Provide an updated look at industry needs
  - Student Surveys
    - Gain a better student perspective
    - Provide further support for knowledge deficiencies

“Identify and classify knowledge deficiencies in graduating computer science students for the purpose of better preparing students for their future careers.”
Systematic Literature Review

- Over 11,000 papers reviewed, 28 Chosen
- 30 Knowledge Deficiency Categories Identified
  - 11 in at least four publications

[Bar chart showing the distribution of knowledge deficiency categories, with Testing and Programming being the most frequently identified, followed by Teamwork and Oral Communication. The chart ends with a total of 23 deficiencies.]
Research Approach

- Three Main Aspects
  - Systematic Literature Review
  - Interviews with Industry Managers
  - Student Surveys
Manager Interviews

• Goals and Research Questions
  • How do knowledge deficiencies identified during the interviews with student applicants differ from those identified after a student begins his or her new job?
  • What knowledge deficiencies prevent recently graduated students from being hired for jobs?

• Participants
  • 14 managers and hiring personnel
  • Midwest U.S. companies that have worked with NDSU
## Results from Manager Interviews

<table>
<thead>
<tr>
<th>Knowledge Deficiency</th>
<th>Total</th>
<th>Interview</th>
<th>On the Job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oral Communication</td>
<td>9</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Project Experience</td>
<td>9</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>Problem Solving</td>
<td>8</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Config. Management</td>
<td>8</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>See &quot;The Big Picture&quot;</td>
<td>6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Teamwork</td>
<td>6</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Testing</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Written Communication</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Working with Customers</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>
## Overall Discussion of Results

<table>
<thead>
<tr>
<th>Knowledge Deficiency</th>
<th>Prior Research (SLR)</th>
<th>Manager Interviews</th>
<th>Student Surveys</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Testing</td>
<td>Strong</td>
<td>Moderate</td>
<td>Strong</td>
</tr>
<tr>
<td>Oral Communication</td>
<td>Strong</td>
<td>Strong</td>
<td>Weak</td>
</tr>
<tr>
<td>Software Tools</td>
<td>Moderate</td>
<td>Strong</td>
<td>Moderate</td>
</tr>
<tr>
<td>Programming</td>
<td>Strong</td>
<td>Weak</td>
<td>Moderate</td>
</tr>
<tr>
<td>Problem Solving</td>
<td>Moderate</td>
<td>Strong</td>
<td>Weak</td>
</tr>
<tr>
<td>Project Experience</td>
<td>Weak</td>
<td>Strong</td>
<td>Moderate</td>
</tr>
<tr>
<td>Teamwork</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Weak</td>
</tr>
<tr>
<td>Written Communication</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Weak</td>
</tr>
</tbody>
</table>
Discussion

• Comparison with Curricula Recommendations
  • Testing is not required to a large degree
  • Later courses may not have enough programming
  • Class projects may not be large enough
  • Does not specify or require technical writing or communication courses
  • Software tools not given much mention
1. Read the following statements and tick the correct answer in the front column.

```c
int a = 10;
int b = 20;
a = b;
```

The new values of `a` and `b` are:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
</tr>
</tbody>
</table>
The new values of a and b are:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
</tr>
</tbody>
</table>

1. Value moved right to left
2. Value copied right to left
3. Value moved left to right
4. Value copied left to right
5. Right value added to left
6. Right value moved and added to left
7. Left value added to right
8. Left value moved and added to right
9. Nothing happens
10. A Test of equality
11. Variables swap values
Assigning Student Programming Pairs Based on their Mental Model

Week 0
Always Pairs
Homogenous Consistent Homogenous Inconsistent Heterogeneous
Work in Pairs

Week 4

Week 12
Results

- Subjects who had exhibited mental model consistency on initial test performed *significantly* better in terms of course grade than the subjects who exhibited inconsistency.
- Subjects who *migrated to mental model consistency* before the re-test scored higher on the midterm exam than subjects who did not undergo migration.
- Additional Future work needed.
Am so easy a caveman can do it!