Debunking the Geek Stereotype with Software Engineering Education

Laurie Williams
North Carolina State University
williams@csc.ncsu.edu

The Mountain Dew bicep curl is one of many helpful geek exercises.
Agenda

• The Geek Stereotype

• Educational Debunk-ments

• Summary

• Acknowledgement: This material is based upon work supported by the National Science Foundation under Grant No. 00305917.
Agenda

• The Geek Stereotype
  ➢ Insularity
  ➢ Long hours
  ➢ Women and others: Lack of relevance of field to society

• Educational Debunk-ments

• Summary
I want to be a doctor some day

I want to help people

Maybe I have to work hard by myself now, but someday I will help people.
I want to be a software engineer some day

Maybe I won’t really have to work long hours all alone

I guess it really is as lonely and hard as they say
Email Survey of Professional Software Engineering

- 359 responses
- 94 companies
- 21 countries
  - 270 responses from US

- 63% working alone
- 24% working with one other person
- 13% working with more than one other person
Perception                      Reality
Myers-Briggs Personality Type

- 153 junior/senior computer science students
  - NCSU, NC A&T, Meredith College
  - Fall 2004 & Spring 2005

- Introverts: 84 (55%)

- Extraverts: 69 (45%)
Stereotype: Long hours

In education . . .

Teacher survey

38 responses

- 31 said their class was “more” or “much more” work than other classes
- 5 said about the same as other classes
- 2 said less work than other classes
The Stereotype

Does the stereotype fit some?

Do they really have to work this long?
Agenda

• The Geek Stereotype

• Educational Debunk-ments
  • Insularity: Collaboration
  • All roads lead to a pair programming talk
  • Long Hours: Commitment-making
  • Relevance of Field to Society: Project Choice

• Summary
Collaboration: Teacher Survey

- SIGCSE survey
  - 50 responses
    - 17 from Colleges
    - 32 from Universities
    - 1 from pre-college

- 37 male teachers/professors

- 13 female teachers/professors
Benefits Cited

Student Morale
- Helps with anxiety level
- Increases camaraderie in class
- Students feel less alienated
- Reduces frustration
- More supportive study environment

Student Performance
- Common goal produces higher achievement
- Improved quality of submissions
- Greater productivity
- Learn better
- Peer pressure enhances effort
- Easier transition to workplace
- Students learn by explaining to others (and work out their errors, clarify concepts)
- Students see a variety of ways to solve a problem
- Develop teamwork skills

Teacher Impact
- Increases size of project that can be tackled
- Less grading
- Less time answering questions about minor issues (Groups can understand while individuals may not)
- Less cheating
**Concerns**

**Competency First**
- Want students to gain confidence in their own ability
- Want to make sure students know the fundamentals
- Weaker students do not develop individual skills
- Passive students do not learn

**Collaboration Management**
- Students get credit for work they may not have done
- Students partition the work and work alone anyway

**Compatibility**
- Stronger students don’t want to collaborate with weaker students
- Student backgrounds vary too much

**Teacher Workload**
- Added time to manage pairs
What about??

- Instead of Sequential . . .
  - Individual Competency
  - Realistic Setting

- Alternate

Many fewer women and minorities (and males!) by this point
Pair Programming in an Intro Course

- North Carolina State University
  - Fall 2001, Spring 2002, and Fall 2002 → AB-AB-AB
  - 660 engineering students
  - Southeastern US, very large public university
  - Large lecture sections
  - Closed lab
  - Pairs assigned, pair rotation

- University of California Santa Cruz
  - Fall 2000, Winter 2001, and Spring 2001 → A-B-A
  - 555 engineering students
  - Western US, large public university
  - Large lecture sections
  - Open lab
  - Pairs by student choice, same partner all semester
**Competency First: Success Rate**

An equal or higher percentage of students in paired labs will **complete the class with a grade of C or better compared to solo programmers.**

<table>
<thead>
<tr>
<th></th>
<th># Paired</th>
<th>% Pair passing</th>
<th># Solo</th>
<th>% Solo Passing</th>
<th>Stat. Sign.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCSU-Sub</td>
<td>171</td>
<td>70.76</td>
<td>255</td>
<td>60.00</td>
<td>Yes. $\chi^2=5.61$, p&lt; 0.023</td>
</tr>
<tr>
<td>NCSU-Total</td>
<td>379</td>
<td>64.37</td>
<td>281</td>
<td>59.78</td>
<td>No. $\chi^2=1.45$, p&lt; 0.228</td>
</tr>
<tr>
<td>UCSC</td>
<td>404</td>
<td>72.30</td>
<td>148</td>
<td>62.80</td>
<td>Yes. $\chi^2=4.57$, p&lt; 0.05</td>
</tr>
<tr>
<td>NCSU+UCSC</td>
<td>783</td>
<td>68.45</td>
<td>439</td>
<td>61.73</td>
<td>Yes. $\chi^2=5.67$, p&lt; 0.017</td>
</tr>
</tbody>
</table>
**Competency First: Exam Scores**

Students who work in pairs will earn exam scores equal to or higher than solo programming students.

<table>
<thead>
<tr>
<th></th>
<th>Pair</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std. Dev.</td>
<td>N</td>
<td>Mean</td>
<td>Std. Dev.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>NCSU F01</td>
<td>74.1</td>
<td>16.5</td>
<td>44</td>
<td>67.2</td>
<td>18.4</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>NCSU S02a</td>
<td>70.6</td>
<td>28.8</td>
<td>82</td>
<td>73.2</td>
<td>27.4</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>NCSU S02b</td>
<td>71.9</td>
<td>26.7</td>
<td>198</td>
<td>74.9</td>
<td>28.5</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>NCSU-F02</td>
<td>75.1</td>
<td>15.7</td>
<td>55</td>
<td>67.5</td>
<td>35.6</td>
<td>110</td>
<td></td>
</tr>
<tr>
<td>UCSC</td>
<td>75.2</td>
<td>18.9</td>
<td>367</td>
<td>74.4</td>
<td>18.5</td>
<td>119</td>
<td></td>
</tr>
</tbody>
</table>
The use of pair programming in an introductory computer science course does not hamper student performance in future solo programming courses.

### Competency First: Future Success

<table>
<thead>
<tr>
<th>Semester</th>
<th>Paired (%)</th>
<th>Solo (%)</th>
<th>Statistical Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS1: Fall 2001 – CS2: Spring 2002</td>
<td>21.42 (6/28)</td>
<td>46.15 (12/26)</td>
<td>No. ( \chi^2 = 3.709, ) ( p &lt; 0.054 )</td>
</tr>
<tr>
<td>CS1: Spring 2002 – CS2: Fall 2002</td>
<td>26.37 (24/91)</td>
<td>29.50 (18/61)</td>
<td>No. ( \chi^2 = 0.179, ) ( p &lt; 0.672 )</td>
</tr>
</tbody>
</table>

NCSU: \% of students whose grades dropped by more than 1/3 of a grade

<table>
<thead>
<tr>
<th>Attempt Rates</th>
<th>Pass Rates (on 1(^{st}) attempt) of Attempters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pair</td>
<td>76.7%</td>
</tr>
<tr>
<td>Solo</td>
<td>62.2%</td>
</tr>
</tbody>
</table>

UCSC: Attempt and Pass Rates for Second CS Class
Aside: Persistence in Computer Science

Students participating in pair programming will be significantly more likely than solo programmers to pursue computer science-related majors one year later.

<table>
<thead>
<tr>
<th></th>
<th>Paired</th>
<th>Solo</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCSC</td>
<td>56.9%</td>
<td>33.8%</td>
<td>$\chi^2(1) = 12.18$, $p &lt; .001$</td>
</tr>
<tr>
<td>NCSU</td>
<td>25.6%</td>
<td>10.5%</td>
<td>$\chi^2(1) = 7.434$, $p &lt; .006$</td>
</tr>
</tbody>
</table>

Percentage of students declaring a Computer Science major 1 year after CS1
Collaboration Management: Pair Evaluation

Peer eval instrument developed by Rich Felder at NCSU.
# Teacher Report
*(names changed to protect the innocent)*

## Peer Evaluation Report

CSC 326 Section 201 Assignment 3 Review 1

<table>
<thead>
<tr>
<th>Assignment 2 #1</th>
<th>Assignment 3 #1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Partner</strong></td>
<td><strong>Partner</strong></td>
</tr>
<tr>
<td>Matt Amyot 201</td>
<td>Will Stuffelbeam 8 OK</td>
</tr>
<tr>
<td>Marie Boucher 201</td>
<td>Frank He 9 Y</td>
</tr>
<tr>
<td>Travis Breaux 201</td>
<td>Partner 0 Cty</td>
</tr>
<tr>
<td>Alex Cau 201</td>
<td>Aaron Pecore 9 Y</td>
</tr>
<tr>
<td>Michael Gagik 201</td>
<td>Lucas Layman 9 Y</td>
</tr>
<tr>
<td>Frank He 201</td>
<td>Marie Boucher 8 OK</td>
</tr>
</tbody>
</table>
Collaboration Management: Pair Rotation Research Methodology

- **Student Survey**
  - **CS1**
    - at end of course Spring 2003 \( \Rightarrow \) N=270
    - Four course sections
    - Four assignments, new partner after each

- **SE**
  - post hoc via email from course Fall 2002 \( \Rightarrow \) N=17
  - One course section
  - Four assignments, new one after each
  - Six-week team project (4-5 person teams)
Student Results

Do you think it was a good idea to change partners after each assignment?
- CS1: 73% yes
- SE: 94% yes

Advantages:
- Exposure to more classmates
- Desire for a new partner

Disadvantages:
- Need to readjust
- Loss of a perfectly-good partner
Teaching Staff Qualitative Findings

- **Advantages**
  - Multiple forms of feedback
  - Natural handling of dysfunctional pairs

- **Disadvantages**
  - Reassigning pairs
  - Need for peer evaluation (all of pair programming)
Collaboration Management: Compatibility Study Design

- CS1 (Freshman, Spring 2003, 387 students)
  - Closed lab
  - Four projects
  - Assigned a new partner each project

- SE (Junior/Senior, Fall 2002, 140 students)
  - Closed lab
  - Four projects
  - Assigned a new partner each project

- OO (Graduate, Fall 2002, 37 pairing students)
  - No closed lab
  - Pairing optional
  - TA assigned partner
# Overall Compatibility Results

<table>
<thead>
<tr>
<th>Class</th>
<th>Very Compatible</th>
<th>OK</th>
<th>Not Compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS1</td>
<td>63% (633)</td>
<td>26% (264)</td>
<td>11% (106)</td>
</tr>
<tr>
<td>SE</td>
<td>65% (324)</td>
<td>27% (132)</td>
<td>8% (40)</td>
</tr>
<tr>
<td>OO</td>
<td>72% (46)</td>
<td>19% (12)</td>
<td>9% (6)</td>
</tr>
</tbody>
</table>
# Compatibility Summary

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>CS1</th>
<th>SE</th>
<th>OO</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-1... <strong>different</strong> personality type <strong>are grouped together</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>H-2... <strong>similar</strong> actual skill <strong>level are grouped together</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>H-3... <strong>similar</strong> perceived skill <strong>are grouped together</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>H-4... <strong>similar</strong> programming self-esteem <strong>are grouped together</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>H-5... <strong>same</strong> gender <strong>are grouped together</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>H-6... <strong>similar</strong> ethnicity <strong>are grouped together</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>H-7... <strong>similar</strong> work ethic <strong>are grouped together</strong></td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H-8... <strong>similar</strong> time management <strong>are grouped together</strong></td>
<td>No</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Concerns - Reprisal

Competency First
- Want students to gain confidence in their own ability
- Want to make sure students know the fundamentals
- Weaker students don’t develop individual skills
- Passive students do not learn

Seems OK

Collaboration Management
- Students get credit for work they may not have done
  - pair evaluation/pair rotation
- Students partition the work and work alone anyway
  - Is this worse (for the students) than solo?

Compatibility ➔ OK 90% of the time
- Stronger students don’t want to collaborate with weaker students
- Student backgrounds vary too much

Teacher Workload ➔ offset by less grading, less technical support
- Added time to manage pairs
Commitment-Making: Brooks

All programmers have a tendency perhaps this modern sorcery especially attracts those who believe in happy endings and fairy god-mothers. Perhaps the hundreds of nitty frustations drive away all but those who habitually focus on the end goal. Perhaps it is merely that computers are young, programmers are younger, and the young are always optimists. But however the selection process works, the result is indisputable: “This time it will surely run,” or “I just found the last bug.”

- Need “gutless estimating.”
- Stop false scheduling to meet the patron’s desired date
- Need quantitative methods, supported by data, productivity figures, estimating rules, etc.

Brooks, Mythical Man Month
Extreme Programming/SCRUM

- Developer makes estimates
- Estimates do not change as part of choosing stories for iteration
- Energized Work primary practice
Project Choice ➔ Computing with a Purpose

- Some “just” interested in the thrill of making computers “do things.”
- For others (esp. women), the study of computer science is made meaningful by its connections to other fields, working with human and social contexts
  - Use computing to study disease
  - Robot car that reduces number of accidents caused by human error
  - Not sports statistics, number crunching games
- “Call It Oceanography and They Will Come”
Grade my assignments for social relevance

- Acme Vending Machine
- Monopoly game
- Pie Throwing Simulation (Abstract Factory Pattern)
- Translate hexadecimal to text (Adapter Pattern)
- Computer Configurator (Factory Pattern)
- Report Generator (Factory Method)
- Football Scoreboard (Observer Pattern)
What’s the harm?

- Bioinformatics, oceanography
  - Thrill seekers ☺
  - Socially relevant ☺

- Aside: industry looking for more well-rounded students
Agenda

• The Geek Stereotype

• Educational Debunk-ments

• Summary
Summary

- The geek stereotype is . . . a stereotype.
- However, computer science education can affirm the stereotype.
- What can be done in education?
  - Add collaborative content
  - Teach commitment-making along with time management, and project management
  - Choose programming projects with social value