How to Rate Programming Skills in Programming Experiments? A Preliminary, Exploratory Study based on University Marks, Pretests, and Self-Estimation

Sebastian Kleinschmager, Stefan Hanenberg
University of Duisburg-Essen, Germany

PLATEAU Workshop @ SPLASH
2011-10-24 Portland, Oregon
Structure of this task

1. Motivation
2. Research question & results
3. Experiment
   - Experiment results
   - Threats to validity
4. Summary & conclusion
Motivation

• Different motivations
  • need for grouping criteria
  • scientific doubts that we currently have valid criteria
  • ...we are frequently asked to apply such criteria
Motivation - Need for grouping criteria

- We all believe that...
  - ...there are developers which differ with respect to development speed, coding quality, etc.
  - ...empirical studies are desirable that are performed (mainly) on good developers
  - ...education might be an indicator for the quality of developers

- But how to identify them?
Motivation – Scientific Need

- Empirical studies in software engineering are typically quite trivial in their experimental design.
- Blocked designs in SE hardly used – and hardly possible.
- Currently, empirical studies frequently do post-hoc grouping of subjects.
- Current dominating problem: large deviation.
- Internal validity of experiments often problematic.
  - Empirical comparisons require different groups.
  - Similarity of groups cannot be decided upfront.
- External validity unclear.
  - Criteria are missing to what extent subjects are representative.
Motivation – Personal Motivation

- We are frequently (always?) asked by reviewers to rate developers via questionnaires.
- We have no idea what those questionnaires should tell us.
  - Does „number of years in industry“ imply good programming skills?
  - Does „number of years in industry“ imply steep learning curve?
  - Does „100.000 JAVA-LOC“ imply better programming skills than someone who has done 5.000 LOC BASIC?
- We have doubts that...
  - ...there are (currently) scientific valid criteria that we can applied.
  - „often mentioned“ criteria reduce the problem of assigning subjects.
  - „often mentioned“ criteria increase the external validity.
Motivation – Personal Motivation

• But...
  • We have done such questionnaires in our experiments

• Idea
  • Test, whether such criteria reveal differences in development skills
Motivation – Personal Motivation

• We are frequently asked by reviewers to do pre-tests with subjects
• We have no idea how to design such pre-tests
  • Example: If the programming task is to implement a loop
    – is „implement a loop“ a valid pre-test?
    – can something completely different be used as a pre-test?

• Idea
  • Test, whether an assumed pre-test (taken from experiment tasks) would have revealed difference in programming skills
Initial Considerations

• We have two criteria that can be directly extracted from our questionnaires
  • University mark
  • Self-estimation
• Due to the nature of our experiments (multiple programming tasks), we can
  • extract one task
  • check whether the task's result is an indicator for the other tasks
Initial Considerations

• Remark
  • Our view was that self-estimation does not represent a valid criteria – we used it only for comparing the results „to something“
Research question

• Does one of the following criteria reveal significant differences in development skills?
  • University marks
  • Self-estimation
  • Pre-tests

• Result
  • No
Approach

- Two previous studies
  - static type systems [StuchlikHanenberg@DLS11], 20 students
  - aspect-oriented programming [KHJ@ESEM09] 21 students

- Both studies
  - measured development time until completion
  - measured Java development times + *something else*

- Our interpretation
  - Java development time as an indicator for programming skills
Approach

• Separating subjects (per experiment) into according to criteria into quartiles

• Detecting, whether there are significant difference between quartiles (development time)
Approach - Example Task

• Example (2\textsuperscript{nd} experiment, task 4)
  • Implement a method that stores a long ball between two players (which are either midfield players or forwards) of the same team

• Example solution

```java
void doLongBall(Player p1, Player p2) {
    if (p1.team != p2.team) throw new SoccerException("...");
    if (p1 instanceof Midfielder) {
        if (p2 instanceof Midfielder) {
            ((Midfielder) p1).longBall++;
            ((Midfielder) p2).receivedLongBall++;
        } else if (p2 instanceof Forward) {
            ((Midfielder) p1).longBall++;
            ((Forward) p2).receivedLongBall++;
        } else throw new SoccerException("...");
    } else throw new SoccerException("...");
}
```
Threats to validity

- Many, many, many
- ...
- Small sample size
- Post-hoc testing
- Multiple U-Tests
- ....
University Marks

- Quantiles 1&3 better than quantile 2!
- No difference between 1\textsuperscript{st} and 3\textsuperscript{rd}!
University Marks

- Quantile 3 better than quantile 2(!)
Pre-test (experiment 1)

<table>
<thead>
<tr>
<th>Task</th>
<th>P-value</th>
<th>dominating group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>0.3</td>
<td>-</td>
</tr>
<tr>
<td>Task 2</td>
<td>0.6</td>
<td>-</td>
</tr>
<tr>
<td>Task 3</td>
<td>0.82</td>
<td>-</td>
</tr>
<tr>
<td>Task 4</td>
<td>0.45</td>
<td>-</td>
</tr>
<tr>
<td>Task 5</td>
<td>0.88</td>
<td>-</td>
</tr>
<tr>
<td>Task 6</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>Task 7</td>
<td>0.45</td>
<td>-</td>
</tr>
<tr>
<td>Task 8</td>
<td>0.71</td>
<td>-</td>
</tr>
<tr>
<td>Task 9</td>
<td>0.26</td>
<td>-</td>
</tr>
</tbody>
</table>

• No significance anywhere...
Pre-test (experiment 2)

- Task 2 showed effect (5 “potential candidate“)

<table>
<thead>
<tr>
<th>Task</th>
<th>P-value</th>
<th>dominating group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>0.44</td>
<td>-</td>
</tr>
<tr>
<td>Task 2</td>
<td>0.02</td>
<td>best dev.</td>
</tr>
<tr>
<td>Task 3</td>
<td>0.26</td>
<td>-</td>
</tr>
<tr>
<td>Task 4</td>
<td>0.23</td>
<td>-</td>
</tr>
<tr>
<td>Task 5</td>
<td>0.12</td>
<td>-</td>
</tr>
</tbody>
</table>
Self-estimation

Development Time / Programming Skills

- 2\textsuperscript{nd} quantile worse than 1\textsuperscript{st}, 3\textsuperscript{rd} better than 2\textsuperscript{nd}
Self-estimation (2\textsuperscript{nd} Experiment)

- Group 5 better than group 1,2
Conclusion

• None of those criteria really showed a large effect

• We should continue with random experiment designs
How to Rate Programming Skills in Programming Experiments? A Preliminary, Exploratory Study based on University Marks, Pretests, and Self-Estimation

Sebastian Kleinschmager, Stefan Hanenberg
University of Duisburg-Essen, Germany

PLATEAU Workshop @ SPLASH
2011-10-24 Portland, Orgegon