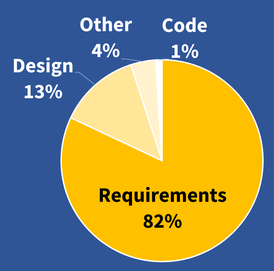
Evaluation of Human Error Abstraction Training for Professional Software Developers

Cameron Booth, Kent State University
Dr. Gursimran Walia, North Dakota State University

Introduction



Distribution of time to repair software defects

Goals

- Develop Human Error Abstraction Training
- Evaluate participants ability to perform the most basic error abstraction (Planning vs. Execution errors)
- Improve training for error abstraction

Study Design

Participants

26 Graduate-Level Computer Science Students from North Dakota State University

Study Steps

Step 1 (Training) – Video Lesson on Human Error Abstraction

Step 2 (Pre-test)– Abstract Errors from 5 Real Software Faults [with feedback]

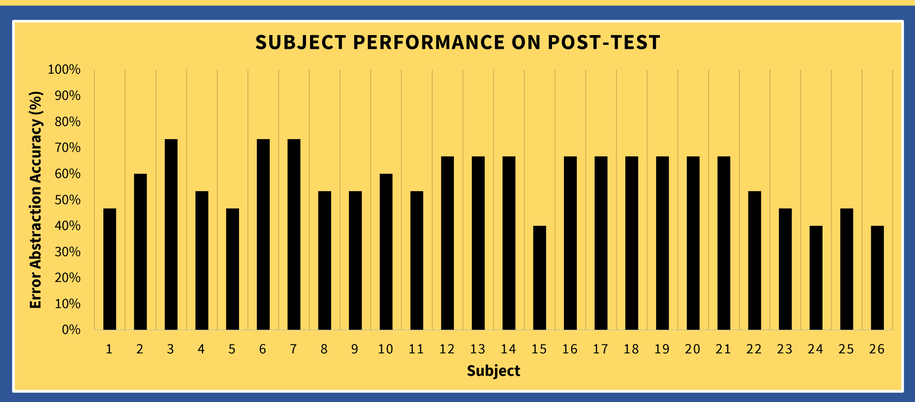
Step 3 (Post-test)– Abstract Errors from 15 Real Software Faults [no feedback]

Step 4 (Survey)– Feedback on Training and Error Abstraction Performance

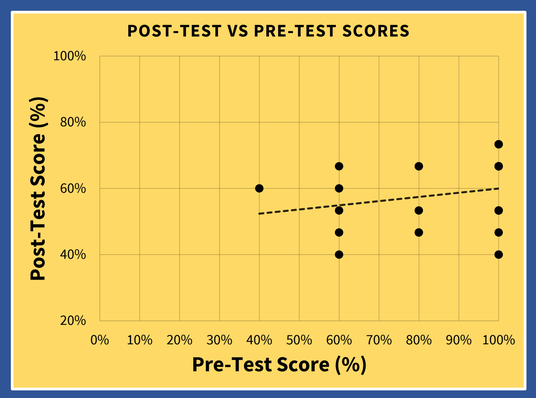
Data

- % of errors correctly abstracted
- % of errors correctly abstracted
- Responses on 7-point Likert Scale

Results



Average subject performance on post-test was 58%, with a Standard Deviation of 11%

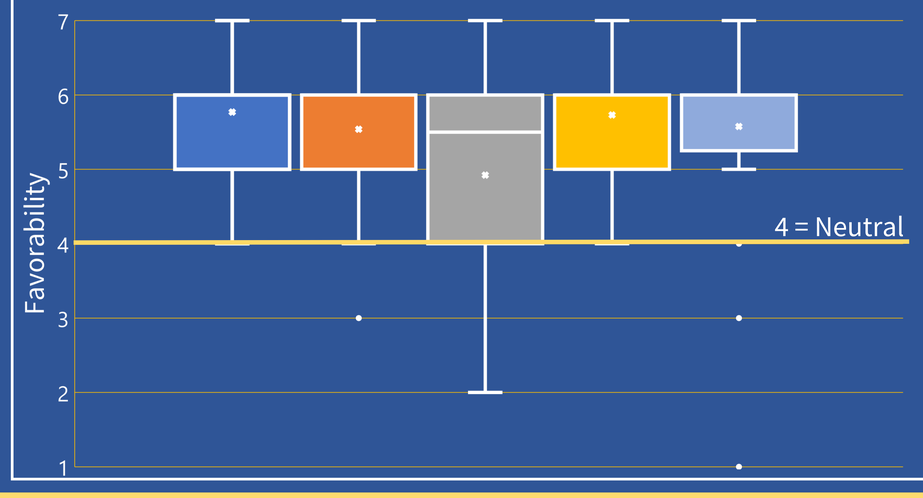


Students' performance on Pre-test is Positively Correlated (Coefficient of 0.201) with their performance during the post-test ($p = 0.162$).

Training Survey Responses

Note: All Responses are in a 7-point Likert Scale

- How would you rate the instructions provided in the error abstraction training video?
- How would you rate the practice questions during the error abstraction training?
- How would you rate your ability to abstract errors from software faults?
- How would you rate your understanding of human errors?
- How would rate your abstracted human errors in terms of actual problems that happen during the software development?



Discussion

- Pre-test performance though correlated, can not be used to predict their performance on post-test
- While subjects exhibited 58% accuracy (lower than expected) during the error abstraction, they rated the training instrument effective (Mean = 5.769, Std. Dev = 0.815)
- Planning errors were harder to identify when compared to the execution errors especially during retrospective analysis (after-the-fact)
- The highest frequency of suggestions to improve the training were to increase the number of examples in the training
- Participants also highly rated their understanding of human errors (Mean = 5.731, Std. Dev = 0.827)
- The accuracy between error types was 55% for planning errors; 64% for execution errors. This result is similar to findings in psychology literature[3]
- This study is an exploratory one, and further research should be done to explore different tangents of training that may have and effect
- We plan to add more examples and practice faults in the training video, as well as evaluate improvements in future studies

References

[1] Anu, V., Walia, G., & Bradshaw, G. (2017, March). Incorporating Human Error Education into Software Engineering Courses via Error-based Inspections. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 39-44). ACM

[2] Doggett, A. M. (2004). A statistical comparison of three root cause analysis tools. *Journal of Industrial Technology*, *20*(2), 2-9.

[3] Reason, J. (1990). *Human error*. Cambridge university press.

Acknowledgements

- Material based on work supported by NSF DUE 1560142 and DUE 1852045. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.
- Thank you to the CiDER REU Faculty and Cohort!