Does Task Training Really Affect Group Performance?

Yuk Kuen Wong
Faculty of Information Technology, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, Australia

David Wilson
Faculty of Information Technology, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, Australia

ABSTRACT

The purpose of this paper is to examine the important relationships between task training experience and software review performance. A total of 192 subjects voluntarily participated in a laboratory experimental research and were randomly assigned into 48 four-member groups. Subjects were required to find defects from a design document. The main finding shows that task training experience has no significant effect on software review performance.

1. INTRODUCTION

While software engineering can be considered a well-established discipline, software development projects are still prone to failure [16]. Even if a software project is not classified as a failure, the general level of software quality leaves room for much improvement [8, 14]. Boehm and Basili [1] stated that one of the most prevalent and costly mistakes made in software projects today is deferring the activity of detecting and correcting software problems until the end of the project. Hence the cost of rework in the later stages of a project can be greater than 100 times of the project costs [5, 6]. About 80 percent of avoidable rework comes from 20 percent of defects [1]. As a result, techniques such as software review for improving software quality are important.

Since Fagan [5, 6] introduced software review (inspection) as an important technique to assure the quality of software projects, researchers have investigated ways to improve software review performance. Researchers believe that task training can improve an inexperienced reviewers' performance [1, 5, 15]. However, there is no empirical evidence in the software review literature to show. As a result, this paper examines how task training experience and software review performance are connected the above-mentioned issue by an empirical experiment.

2. LITERATURE REVIEW

2.1 Task Training

The software review literature suggests that training can improve defect detection skills [7, 17]. Training outcomes have an effect on knowledge and skills and this can be evaluated in terms of both its immediate and its long-term effects. However, experience gained from training may not be the same for all individuals. This refers to an aptitude-treatment interaction (ATT) effect.

An aptitude is defined as any characteristic of a trainee that is a determinant of their ability to benefit from training, including knowledge, skills and previous experience [3]. Kirkpatrick's

[11, 12] typology suggested that four levels of training effectiveness which include trainee reactions to training or affective responses to training, training learning or cognitive responses to training, subsequent outcomes of trainee behaviour and organization results. As a result, we are particularly interested in the effect of task training experience on the software review performance.

2.3. Performance

Group performance is defined as how well the individual and group carry out the decisions they make but not the quality of decisions itself, even though decision quality is often used as an indicator of group performance [e.g., 4, 9, 10]. In the human performance theory, Campbell's theory [2] suggests that experience, knowledge, and motivation could affect task performance. In particular, Campbell [2] proposed that performance is a function of an individual's declarative knowledge, procedural knowledge and skill, and motivation. Declarative knowledge is defined as knowledge required to complete a task. Procedural knowledge refers to skill-based knowledge about how effectively a task is performed. Declarative knowledge and procedural knowledge are based on education, training, experience and motivation.

In the context of a software review, at the completion of defect detection, there are two types of quantitative outputs: the reviewed software artefact, and quantitative outcomes such as defect information recorded in defect forms (e.g. number of defects). There are four possible outcomes of defect detection including:

- hit (defect exists and is successfully detected),
- miss (defect exists but is not detected),
- false positive (defect does not exist but is wrongly identified), and
- correct rejection (defect does not exist and is not identified).

The probability of results in each of these categories is determined by the performance of individuals and the interaction between those individuals in a group [5].

2.3 Research hypotheses

Sauer et. al's theory [15] states that expertise is a key driver in a group performance. They claimed task training can improve individual performance and the literature suggests that task training can improve defect detection skills [7, 17]. However, there is no empirical evidence to support whether software review training experience has a significant effect on review performance. As a result, we formulate:

Hypothesis 1a: Task training (TT) will have a positive effect on individual performance.

Hypothesis 1b: Task training (TT) will have a positive effect on group performance.

3. METHODOLOGY

3.1 Experimental Settings

A total of 192 subjects voluntarily participated in the research. The subjects were undergraduates students enrolled in an information systems course at The University of New South Wales in Australia. All the subjects majored in Information Systems and enrolled in a three-year course. They were randomly assigned to 48 four-member groups. Age range of the subjects was between 19 to 42 years old (mean = 21). There are approximately 58% male and 41% female. About 18% of subjects received formal (documented) or informal (undocumented) task training. The means of task training experience is 2.5 hours.

The software review task employed in this research was a design document in which subjects were required to find defects. The aim of the task was to allow groups to perform defect detection processes in both individual preparation and group review meeting.

3.2 Measurement Model

The measurement of individual (I_) and group (G_) performance include:

- True defects (TR) - defects that actually exist and have been successfully detected
- False positives (FA) - defects that do not exist but were wrongly identified
- Incomplete information (IN) - defects that were identified but lacking detailed description to indicate they are true defects.
- Net defects (NE) - true defects minus false positive and incomplete information.
- Total issues (TL) - true defects plus false positive and incomplete information.

Note that measurement of performance is based on the number of defects found. To assess the reliability of this measuring criterion, two lecturers evaluated all the defects reports. The interpreter agreement between the two lecturers was found to be .90, which indicated that the performance-measuring criterion has a reasonable degree of reliability. Measurement of task training refers to any form of training in software review. This could be formal (documented) or informal (undocumented).

3.3 Experimental Procedure

The experimental procedure comprises six major steps:

1. Briefing the purpose of the task – laboratory supervisor distributed the task instructions and went through the requirements.
2. Assigning group – supervisor randomly assigned four subjects to a group.
3. Performing one-hour individual preparation – individually studies the design document, as well as examines the document.
4. Performing one-hour group meeting – An interactive group meeting for collecting the defects from the design document. All group members must agree defects found from the individual preparation stage before recording them in the group record form.
5. Post meeting survey – A questionnaire survey was conducted. All subjects completed and returned the questionnaire to the supervisor.
6. Debriefing – feedbacks and comments.

4. RESULTS AND DISCUSSION

4.1 Hypotheses Test

All data analyses were carried out with a significant level of 0.05, two tailed. Pearson's correlation test was used to test the relationships between experience, task training and performance. Interesting findings show that there is no significant relationship between task training and individual performance (true defects: r = 0.21, p = n.s.; false positives: r = -0.27, p = n.s.; in-completed information: r = -0.4, p = n.s.; net defects: r = 0.28, p = n.s.; total issues: r = 0.2 p = n.s.).

Also, there is a weak relationship between training and group performances (true defects: r = -0.37, p = n.s.; fales positives: r= 0.5, p = n.s.; in-completed information: r = 0.3, p = n.s.; net defects: r = -0.46, p < 0.01; total issues: r = -0.4, p = n.s.) (See Table 1). The results do not support H1a and H1b.

Table 1: Results of correlation analysis on the relationships among experience, task training and performance

	I_TR	I_FA	I_IN	I_NE	I_TL	G_TR	G_FA	G_IN	G_NE	G_TL	TRA
TRA	.21	-.27	-.04	.28	-.02	-.37	.50	.03	-.46**	-.04	1.00

*$p < 0.05$.
**$p < 0.01$.

4.2 Implications

The main goal of this study was to validate the relationships between task training experience and software review performance. It is interesting that the results indicate that task training does not have a positive effect on either individual performance or group performance. However, the authors suggest that repeatable control experiments are required to validate the relationship.

The findings can have some implications for researchers and practitioners. For researchers, task training does not have a significant effect on performance in this study. This finding directly contradicts current software review literature. One of possible explanations is that subjects lacked appropriate training in software review. The authors believe that appropriate training such as task oriented reading techniques (e.g. perspective reading technique) [1] *might* have a beneficial effect on reviewers.

However, whether task training really can improve software review performance remains questionable. Given that most projects have tight schedules and budgets, it is necessary to ask the questions: does the effort and cost of software review training yield benefits? What are the costs and benefits of a training program? Is software review training cost effective? Studies show that even though U.S. organizations spend more than $50 billion on training annually [11], less then 50% of organizations evaluate the value returned from this budget expenditure [13]. We suggest that future research should evaluate the organizational cost of conducting training in software review.

Practitioners should pay more attention to the value of software review training. The results indicate that the value of current software review training programs (either in university or industry) has not been determined conclusively. Does current software review training really improve performance? Managers should be concerned with the evaluation of training programs.

5. LIMITATIONS

Six limitations associated with internal and external validity in this study remain and will be incorporated in future research:

1. One of the limitations of the study was the training effect. Training effect is due to (1) subjects learning as the experiment proceeds. Subjects had six weeks intensive course training in the software package used and most subjects were familiar with the software itself. The software review task was conducted in the last week of the training course. The authors believe that that is the major reason why task training (i.e. software review training) had no impact on performance. (2) Subjects who have previous software review training experience (this could include formal training or informal training) may not benefit from this in performing the task.
2. The design of task instruments and performance measurements may not be representative of real problems.
3. The sample was relatively small (48 groups). In fact, small sample sizes are a common limitation affecting many group-based research studies. Although the small sample could have contributed to lack of support for some of our hypotheses, full support was found for the research hypotheses.
4. Laboratory based experimental studies are often limited by low external validity although internal validity is high. As a result, generalization of the research findings into real world contexts should be done cautiously.
5. One potential problem of internal validity is the selection effect. Selection effect is due to natural variations in human performance. For example, the random assignment of subjects may accidentally create an elite team. Therefore, differences in the group performance may in fact be differences in a groups' natural ability.

6. References

[1] B. W. Boehm and B. R. Basili, "Software Defect Reduction Top 10 List", *IEEE Computer,* 34 (1), January 2001.

[2] J. P. Campbell, "Modelling the Performance Prediction Problem in Industrial and Organizational Psychology", In M. D. Dunnette, L. M. and Hough (Eds.), *Handbook of Industrial and Organizational Psychology*, 2nd Edition, Consulting Psychologists Press Inc., Palo Alto, CA, 1990, pp.87-732.

[3] L. J. Cronbach, and R. E. Show, *Aptitudes and Instructional Methods: A Handbook for Research on Interactions*, Irvington, New York, 1977.

[4] S. J. Czaja, J. Sharit, and R. Ownby, "Examining Age Differences in Performance of A Complex Information Search and Retrieval Task", *Psychology and Aging,* 16 (4), 2001, pp.564-579.

[5] M. E. Fagan, "Design and Code Inspections to Reduce Errors in Program Development", IBM *System Journal*, 15(3), 1976, pp.182-211.

[6] M. E. Fagan, "Advances in Software Inspections", *IEEE Transaction on Software Engineering*, 12 (7), July1986.

[7] P. J. Fowler, 'In-process inspection software products at AT&T", *AT&T Technical Journal*, 65 (2), 1986, pp.744-751.

[8] R. L. Gless, 'Evolving a New Theory of Project Success". *Communications of the ACM*, 45 (11) 1999, p.7.

[9] J. R. Hollenbeck, D. R. Ilgen, J. A. LePine, J. A. Colquitt, and J. Hedlund, 'Extending The Multilevel Theory Of Team Decision Making: Effects Of Feedback And Experience In Hierarchical Teams", *Academy of Management Journal,* 41(3), 1998, pp. 269-282.

[10] P. H. Kim, 'When, What You Know Can Hurt You: A Study of Experiential Effects on Group Discussion and Performance", *Organizational Behavior and Human Decision Processes*, 69 (2), 1997, pp.165-177.

[11] D.L. Kirkpatrick, 'Evaluation of Training," In R.L. Craig & L.R. Bittel (eds.), *Training and Development Handbook*, McGraw-Hall, New York, 1967, pp. 87-112.

[12] D. Kirkatrick, 'Making It All Worker-Friendly", *Fortune* 128 (7), 1993, pp.44-53.

[13] M. London, 'Managing the Training Enterprise: Hight-Quality", *Cost-Effective Employee Training in Organizations,* Jossey-Bass, San Francisco, 1989.

[14] K. Lyytinen, & R. Hirschheim, 'Information Systems Failure: A Survey and Classification of the Empirical Literature", *Oxford Surveys in Information Technology*, 4, 1987, pp.257-309.

[15] C. Sauer, R. Jeffery, L. Land, P. Yetton, 'Understanding And Improving The Effectiveness Of Software Development Technical Reviews: A Behaviorally Motivated Programme Of Research", *IEEE Transactions on Software Engineering*, 26 (1), 2000, pp.1-14.

[16] I. Sommerville, *Software Engineering*, 5th Edition, Addison-Wesley, England, 1995, p.66.

[17] S. H. Strauss & R. G. Ebenau, *Software Inspection Process*, McGraw-Hill, 1994.