Proceedings of the 7th IASTED International Conference
SOFTWARE ENGINEERING AND APPLICATIONS
November 3-5, 2003, Marina Del Rey, CA, USA

DO INDIVIDUALS' EXPERIENCE AND TASK TRAINING REALLY AFFECT SOFTWARE REVIEW PERFORMANCE?

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

Abstract

Past research suggests that individuals' experience and task training are the factor to software review performance. However, there is no empirical evidence in the software review literature to show how individuals' experience, task training and performance are connected. As results, the aim of this paper is to presents the important relationships between individuals' experience, task training and software review performance. A laboratory experiment was conducted in autumn 2003 at The University of New South Wales in Australia. One hundred and ninety-two volunteer university students were employed. Subjects were required to detect defects from a design document. The main findings include (1) role experience has a positive effect on performance; (2) working experience in the software industry has a positive effect on performance; (3) task training has no significant effect on performance.

Key Words

Software Review, Experience, Training and Performance.

1. Introduction

Software review is considered to be one of the most cost effective techniques in finding and removing defects at various stages of the software development life cycle [1], [2], [3], [4]. The main goals of software review are to improve software quality of the product and increase software developers' productivities [5], [6].

Software review (inspection) was originally introduced by Fagan [2]. The review process essentially includes six major steps: planning, overview, individual preparation, group review meeting, rework and follow-up [2], [5].

1. Planning - organize and prepare the software review, typically for preparing the review materials and review procedure, forming review team and scheduling review meeting, selecting review participants and assigning roles [2], [7].
2. Overview - author explains overall scope and the purpose of the review.
3. Individual preparation - individual reviewers analyze and review the software artefact.
4. Group review meeting - find errors [2], sometimes also called "logging meeting" [9]. Review teams correct and the reader summarizes the work.
5. Rework - defect correction [2], which involves the author in resolving problems by reviewing, revising and correcting the identified defect [2] or by decreasing the existence of errors [9] of the software artefact.
6. Follow up - validate the correction quality and decide if re-inspection is required [2], [10].

Since Fagan [2] introduced software inspection (review) as an important technique to assure the quality of software projects, researchers have investigated ways to improve software review performance. Wong [11], [12] suggested that experience is critical to review performance. Researchers also believe that training can improve an inexperienced reviewers' performance [13]. However, there is no empirical evidence in the software review literature to show how individuals' experience, task training and performance are connected. As a result, this paper examines the above-mentioned issue by an empirical experiment.

2. Literature

2.1 Experience

Past research has shown that experience has a significant effect on performance [12]. Individual abilities and skills have the most important influence on task performance [12]. Abilities refer to the knowledge an individual has of a specific task and skills refer to the specific competencies required to complete the task.

According to the "EIIO" theory [11], [12], it is suggested that implicit inputs are one of the major factors of review performance and are difficult to articulate with formal language or notations [14]. Implicit inputs include reviewers' personal knowledge and expertise as well as behavioral aspects (norms, beliefs, values etc) [14] of reviewers. Individual performance is determined by two major types of implicit inputs that include task-relevant

abilities and skills (AS) (i.e. Can I do it?), and motivation and effort (ME) (i.e. Will I do it?) [15]. "When capacity, interest and effort are present; the best performance is likely to result" [16].

Expert theory suggests that it is important to employ expertise to achieve the highest performance. The achievement of experts is the result of acquiring many years of experience in their domain area. The scope of expert theory that consists of [17]:

- Age associated with peak performance - the age at which experts attain their highest level of performance is closely related to their domain of expertise.
- Ten years of necessary preparation – preparation is essential in most domains in order to achieve the highest performance. In other words, the ten years experience rule is necessary even for a talented individual.
- Role of deliberate practice – domain knowledge often draws from effective training and qualification with significant feedback and improvement in order to achieve the highest performance.

In fact, Sauer et al [13] theorize that expertise is a key driver of software review performance. Wong [12] found that experience (i.e. knowledge and skills) is the most significant input influencing performance. However, some studies have shown that individuals who have experience do not necessarily perform better than in-experience individuals [18].

In this research, we look at the effects of role experience and working experience in the software industry. Role experience is defined as the experience gained by performing a similar task (i.e. software review), as long as the experience is the relevant to the experimental task. Working experience is defined as any experience gained in the software industry.

2.2 Task Training

The software review literature suggests that training can improve defect detection skills [19], [20]. 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 [21]. Kirkpatrick's [22], [23] 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

In this paper, performance is defined as how well the individuals carry out the decisions they make but not the quality of decisions itself, even though decision quality is often used as an indicator of performance (e.g. [24]). In the human performance theory, Campbell's theory [25] suggests that experience, knowledge, and motivation could affect task performance (see Figure 1).

Performance = f (declarative knowledge, procedural knowledge and skills, motivation)

Figure 1: Determinants of task performance

In particular, Campbell [25] 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 as shown in Figure 2. These include:

- 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 cells is determined by the performance of individuals [2].

		Behaviour	
		Defect detected	Defect not detected
Defect	Defect exist	Hit	Missed
	Defect does not exist	False positive	Correct rejection

Figure 2: Possible Outcomes of defect Detection

3. Research Hypotheses

Studies have shown that task experience has a positive effect on performance under most circumstances (e.g. [12], [26], [27]). Korman [28] suggests that role experience also has an effect on performance. Further, we believe that work experience in the software industry provides an understanding of software practice that has a positive effect on review performance. The experiment we conducted explores the impact of role experience and work experience on software review and tests the hypotheses as follow:

Hypothesis 1a: Role experience (RE) will have a positive effect on individual performance.

Hypothesis 1b: Individuals who have working experience in the software industry (IND) will have a positive effect on individual performance.

uer et. al's theory [13] states that expertise is a key ..iver of software review performance. They claimed task training can improve individual performance and the literature suggests that task training can improve defect detection skills [13]. However, there is no empirical evidence to support whether software review training has a significant effect on review performance. As a result, we formulate:

Hypothesis 2: Task training (TT) will have a positive effect on individual performance.

Further, studies suggest that the level of task training needed is determined by the levels of individuals' experience [26], [29]. The more experience an individual has, the less task training they require. Thus we formulate:

Hypothesis 3a: Role experience will have an effect on task training.

Hypothesis 3b: working experience in software industry will have an effect on task training.

3. Methodology

3.1 Experimental Settings and Subjects

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. Age range of the subjects was between 19 to 42 years old (mean = 21) (see Table 1). There are approximately 58% male and 41% female (see Table 2). Table 3 shows that more than half subjects have software industry working experience (mean of months experience = 11 months) and about 25% had role experience in software review and about 18% of subjects received formal (documented) or informal (undocumented) task training. The means of role experience is 3 months; industry experience is 11 months and training is 2.5 hours.

Table 1: Age

Age	Frequency	Percent
19	8	4.1
20	54	27.8
21	58	29.9
22	40	19.6
23	13	6.7
24	11	5.7
25	5	2.6
27	1	.5
30	2	1.0
36	1	.5
42	1	.5
Total	192	100

Table 2: Gender

	Frequency	Percent
Male	112	58.3
Female	80	41.7

Table 3: Role Experience (RE), Software Industry Experience (IND) and Task Training (TT)

	RE	IND	TT
Percent	Y=24.48 N=75.52	Y=55.21 N=44.79	Y=18.75 N=81.25
Mean	3 months	11 months	2.5 hours

3.2 Task

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 individual defect detection processes.

3.3 Measurement Model

The measurement of individual (I_) performance include:

- True defects (TR) - defects that actually exist and have been successfully detected
- False positive (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 experience can be classified into role experience and software industry experience. Task training refers to any form of training in software review. This could be formal (documented) or informal (undocumented).

3.4 Experimental Procedure

The experimental procedure includes (1) briefing the purpose of the task, (2) performing one-hour individual defect detection, (3) post meeting survey and (4) debriefing. In the first stage, a laboratory supervisor distributed the task instructions and went through the requirements. In the second stage, all subjects individually examined the design document. Next, a questionnaire survey was conducted. All subjects completed and returned the questionnaire to the supervisor. Finally, feedbacks and comments were collected from the subjects in the debriefing stage.

4. Analyses and Results

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. Table 4 shows the results of the Pearson's correlation test. The results show that there is a positive relationship between software role experience and individual performance (true defects: $r = 0.18$, $p < 0.05$; in-completed information: $r = -0.26$, $p < 0.01$; net defects: $r = 0.16$, $p < 0.05$). These results indicate that H1a is supported. However, there is a weak relationship between people who have working experience in software industry and individual performance (true defects: $r = 0.05$, $p < 0.05$, in-completed information: $r = -0.18$, $p < 0.05$). Hence, H1b is weakly supported.

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.). The results do not support H2. In Addition, it was found that there is positive relationship between role experience and software industry working experience ($r = .36$, $p < 0.01$); and a negative relationship between training and software industry working experience ($r = -.66$, $p < 0.01$); plus no significant relationship between role experience and task training ($r = -.3$, p = n.s.).

Table 4: Results of correlation analysis on the relationships between experience, task training and performance

	I_TR	I_FA	I_IN	I_NE	I_TL	RE	TRA	IND
RE	.18*	-.04	-.26**	.16*	.14	1.00		
TRA	.21	-.27	-.04	.28	-.02	-.30	1.00	
IND	.05*	-.03	-.18*	.06	.04	.36**	-.66**	1.00

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

Although correlation analysis demonstrated the positive relationship between experience and performance, a regression analysis is necessary in order to test the cause-and-effect relationship. Also, because correlation analysis showed both role experience and current working experience are positively related to performance, we want to know which variable is the most significant determinant of individuals' performance. Hence, we carried out a set of regression analyses: performance on role experience, performance on working experience. Table 5 shows the regression results. Hypotheses 1a and 1b are supported because experience is a significant determinant of individual performance. However, results also indicate that experience does not determine the net defects. Figure 4 shows the revised model.

Table 5: Research Hypotheses (individual performance)

	TR	FA	IN	NE	TL
RE	R=.21* R^2=.043	R=.38* R^2=.01	R=.26** R^2=.065	R=.16 R^2=.026	R=.14** R^2=.019
IND	R=.54* R^2=.03	R=.26* R^2=.01	R=.18** R^2=.032	R=.56 R^2=.03	R=.035** R^2=.001

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

5. Discussions and Conclusion

The main goal of this study was to validate the relationships between experience, task training and performance. Two main findings of this study are summarized and discussed below.

As expected, the experienced individuals have a positive effect on individual performance (H1a and H1b were

supported). Although this finding is interesting and may help researchers to explore why the experience and performance relationship is always controversial in many other research studies, our research indicates that understanding the key attributes and characteristics of reviewers is important in selecting the people who will perform the software review task. For example, do different types of experience and/ or number of years of experience affect performance? It is suggested that the selection of reviewers is an important area of research in future empirical studies of software review.

It is interesting that the results indicate that task training does not have a positive effect on individual performance. The findings demonstrate that task training is not determined by role experience but by working experience. The results suggest that the more work experience an individual has in software review industry, the less task training they require. However, role experience did not impact on the task training required. The authors suggest that further research should investigate reasons behind these relationships. Repeatable control experiments are required to validate the revised model.

5.1 Implications

These findings can have some implications for researchers and practitioners. For researchers; the findings indicate that there is a positive relationship between experience and individual performance. It is suggested that further research should investigate what types of role experience and software industry experience could have significant effect on software review performance. Though Fagan [2], [5] proposed that role assignment would improve the performance, it would be interesting for researchers to find out how many years of experience and what types of role experience would be optimal in individual performance. Investigations should consider the effect of different kinds of individual experience on performance, for example the characteristics (computer technical skills vs. business skills) of experience should be considered when carrying out experience-performance related research.

Task training does not have a significant effect on performance in this study. This finding 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 [15], less then 50% of organizations evaluate the value returned from this budget expenditure [19]. We suggest that future research should evaluate the organizational cost of conducting training in software review.

Practitioners should pay more attention to the experience of the reviewers. The authors validated experience as a critical factor in software review performance. Role experience and software industry experience should be considered important attributes when seeking a software reviewer. 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. These research findings on experience-performance, experience-training and training-performance relationships can also help human resources managers to make a better selection of potential employees.

5.2 Limitations

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

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. Another limitation is experience effect. The subjects may not be representative of software developers. The average mean of software industry experience is only 11 months and role experience is only 3 months. Plus, the design of task instruments and performance measurements may not be representative of real problems. Further, the sample was relatively small (192 subjects). In fact, small sample sizes are a common limitation affecting many laboratory 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. Finally, 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.

6. References

[1] B. W. Boehm & Basili, B. R. Software defect reduction top 10 list, IEEE Computer, (34), 2001.

[2] M. E. Fagan, Design and code inspections to reduce errors in program development. IBM System Journal, 15(3), 1976, 182-211.

[3] R. L. Gless, Evolving a new theory of project success. Communications of the ACM, 45(11), 1999, 7.

[4] K. Lyytinen, & R. Hirschheim, Information systems failure: a survey and classification of the empirical literature, Oxford Surveys in Information Technology, 4, 1987, 257-309.

[5] M. E. Fagan, Advances in software inspections. IEEE Transaction on Software Engineering, 12(7), 1986.

[6] I. Sommerville, Software engineering, 5th Edition. (England: Addison-Wesley, 1995), 66.

[7] F. A. Ackerman, L.S. Buchwald, F. H. Lewski, Software inspection: an Effective verification process, *IEEE Software*, 1989, 31-36.

[8] T. Gilb, & D. Graham, Software inspection. (Addison Wesley Publishing Company, 1993)

[9] G. C. Shirey, How inspections fail, Proceedings of 9th International Conference on Testing Computer Software, 1992, 151-159.

[10] Doolan, E.P. (1992). Experience with Fagan's Inspection method. Software-Practice Experience. 22 (3), 173-182.

[11] Y. K. Wong, The impact of inspection inputs on software inspection: an empirical investigation. Proceedings of Doctoral Workshop of International Conference on Software Engineering. Use of software inspection in practice, Proceedings of International Conference on Software Engineering, 2002.

[12] Y. K. Wong, An exploratory study of software reviews in practice, Proceedings of Portland International Conference on Management of Engineering and Technology, 2003.

[13] C. Sauer, Jeffery, R., 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, 1-14.

[14] I. Nonaka & H. Takeuchi, The knowledge-creating company: how Japanese companies create the dynamics of innovation. (Oxford: Oxford University Press, 1995).

[15] M. E. Hacker & B. M. Kleiner, A study of the effects of procedural structure and anonymity on process improvement work groups. Engineering Management Journal, 11(4), 1999, 26-30.

[16] R. P. Vecchio, G. Hearn & G. Southey Organizational behaviour: lift at work in Australia, First Australian edition, (Sydney: HBJ, 1992).

[17] K. A. Ericsson, R.T. Krampe and C. Tesch-Romer, The role of deliberate practice in the acquisition of expert performance, Psychology Review, 100(3), 363-406, 1993

[18] S. Biffl & M. Halling, Investigating the defect detection effectiveness and cost benefit of nominal inspection teams, IEEE Symposium on Software Metrics, 2003, 385-397.

[19] P. J. Fowler, In-process inspection software products at AT&T. AT&T Technical Journal, 65(2), 1986. 744-751.

[20] S. H. Strauss & R. G. Ebenau, Software inspection process. (McGraw-Hill, Inc, 1994).

[21] L. J. Cronbach, & R. E. Show, aptitudes and instructional methods: a handbook for research on interactions, (New York: Irvington, 1977)

[22] D.L. Kirkpatrick, Evaluation of training, in Training and development handbook, R.L. Craig & L.R. Bittel (eds), (New York: McGraw-Hall, 1967), 87-112.

[23] D. Kirkatrick, Making it all worker-friendly. Fortune 128(7), 1993, 44-53.

[24] S. J. Czaja, J. Sharit, & R. Ownby, Examining age differences in performance of a complex information search and retrieval task, Psychology and Aging, 16 (4), 2001, 564-579.

[25] J. P. Campbell, Modeling the performance prediction problem in industrial and organizational psychology. In M. D. Dunnette, and L. M. Hough, (Eds.), Handbook of industrial and organizational psychology, 2nd Edition, (Palo Alto, CA: Consulting Psychologists Press Inc., 1990), 687-732.

[26] G. Littlepage, W. Robison & K. Reddington, Effects of task experience and group experience on group performance, member ability, and recognition of expertise. Organizational behavior and human decision processes, 69(2), 1997, 133-147.

[27] S. J. Motowidlo & J. R. Van Scotter, Evidence that task performance should be distinguished from contextual performance. Journal of Applied Psychology 79(4), 1994, 475-480.

[28] A .K. Korman, Organizational behavior. (Englewood Cliffs, NJ: Prentice hall, 1977).

[29] M. London, Managing the training enterprise: hight-quality", cost-effective, Employee training in organizations, (San Francisco: Jossey-Bass, 1989)