Observational Studies to Accelerate Process Experience in Classroom Studies: An Evaluation

Jeff Carver[1] Forrest Shull[2] Victor Basili[1,2]

[1]Department of Computer Science
University of Maryland
{carver,basili}@cs.umd.edu

[2] Fraunhofer Center for Experimental
Software Engineering, Maryland
fshull@fc-md.umd.edu

Abstract

Software Engineering studies run in classroom environments can and have made important contributions to empirical software engineering. Because the goal of such studies is to improve the state of the practice in industry, researchers must understand and account for the differences between university students and industrial professionals. One major difference identified is the amount of training and practice that students and professional may have when learning a new technique. We propose and test a method of allowing university subjects to cost-effectively gain experience to compensate for this difference. The results show that the proposed method for gaining experience provided subjects with enough experience to improve their effectiveness in some but not all cases. There was also an indication from the results that the proposed method allowed the subjects to become more comfortable with a new technique.

Keywords: Empirical study, requirements inspections, software process, experimental process, software quality

1. Introduction

Software engineering studies run in a classroom environment, using university students as subjects, are a useful and necessary method of advancing knowledge in the field. There are many practical reasons for using this readily available subject pool [7]. While there are also well known threats to validity (especially external validity) associated with such studies, analyses have been done showing that such threats are not as strong as sometimes imagined. For example, it has been shown that on certain tasks, such as assessing the impact of project factors on the lead-time of software development projects, Computer Science master's degree students do not perform significantly differently from professional software developers [9].

As an example of the contributions of classroom studies to research goals, we have used several classroom studies as part of a larger research program into software inspection techniques. Some studies have directly provided important insights into the nature of these techniques. For example, from a study run in a class with students with greatly varied backgrounds, ranging from very inexperienced at software development to returning professionals with many years of experience, we have found that the effectiveness of subjects when using a specified technique varied based on their level of prior experience [6,16].

Another situation where classroom studies have been useful is in the debugging of development technologies and experimental protocols for later use in industry. For example, we used a series of classroom studies to debug the steps of a design inspection technique [18] prior to its use in industry [11]. In related work, researchers at the Norwegian University of Science and Technology used a classroom study with undergraduates to identify problems related to subject motivation and accurate time estimates for experimental tasks [1]. Using a classroom study first, the researchers were able to understand and correct these problems before using expensive professional subjects. Those experiences later led to a successful industrial experiment [8].

Despite these successes, one important issue that has to be accounted for when running classroom studies is that the level of process training that subjects receive in a classroom setting is in most cases very dissimilar to that received in an industrial environment. Because of the constraints of a classroom environment, the researchers often do not have time for thorough training of the subjects. Often subjects are trained on a new technique and then have their performance measured almost immediately with little or no opportunity to practice the

new technique. This lack of process training introduces threats to validity of the results because the technology is not being measured as it was intended to be applied by professionals (who would typically have more chances to practice with the technology in their day-to-day work) but instead is being measured in the early stages of the learning curve. Students in such studies may need additional time (that is not always available in a classroom environment with a busy curriculum) to get comfortable with applying the techniques, to better understand how the technology should be applied, and to work through practical problems. Our goal in this work was to mitigate this problem of lack of process experience by experimenting with new methods for training students. This paper presents an evaluation of the results of one such method.

2. Objects of study

This section begins with a brief discussion of why it is important to study process experience. Then it proceeds to provide a brief overview of the steps in the proposed method for allowing subjects to gain process experience. Finally, the section concludes with an introduction to the technology that was used to evaluate the method.

2.1 Need for process experience

Researchers have suggested that the selection of inspectors based on their characteristics can impact the defects found during the inspection process [13,14,15]. The results of previous studies on process experience showed it to be an important factor influencing inspection effectiveness: Analysis of results from a series of studies indicated that different types of experience with inspections, such as comfort level and experience with reviewing requirements, and experience in the software inspection technique could be important factors in the effectiveness of an inspector [6].

2.2 Method for accelerating process experience

The overall goal of this work was to test a proposed method for quickly and effectively increasing process experience that will allow the results from classroom studies to be more applicable to an industrial context. The proposed method for gaining process experience is to have subjects work in pairs and gain experience through observation.

Earlier research had identified some potential benefits of working in pairs to perform software engineering tasks. In an early study on software inspections, subjects who worked together as a pair had a significant increase in productivity, not necessarily efficiency, over subjects who were working alone [3]. In a user interface inspection study, researchers had some subjects work alone and some subjects work in pairs with one subject ensuring that the process steps were accurately followed. In this study the subjects working in pairs reported both that the process was easier to use and that they had higher process conformance [22]. In the world of eXtreme Programming, where two programmers work together at the same computer to write code, researchers have also noted the learning that takes place by the programmer who is observing [12,21].

In order to take advantage of the potential benefits of observation, our previous experience had led us to believe that *observational studies*, studies where an experimental subject who is doing a procedure is observed by someone else, would be an effective method for doing this observation. Because observers in a previous study [18] were able to gain useful insight into the execution of a technique, we believed that a subject could gain inspection experience by observing another subject performing an inspection. This approach not only allows the subjects to gain experience by observing, but through the notes taken during observation, it also helps the experimenters better understand the application of the process. These observational studies provide a level of detail about individual process steps and their usefulness that is difficult to collect using traditional post-experiment questionnaires [19]. Another goal of using the observational approach, similar to the interface inspection study discussed earlier, was for the observer to act as a "process guide" to ensure that the inspector was following the procedure and that any deviations from the process were conscious decisions by the inspector rather than simple oversight.

Based on the success of the observational studies, the methodology proposed below used the idea of observation to allow subjects to gain process experience and then evaluated the effect of that experience. The steps of the methodology are:

Step 1: Divide the subjects into two-person groups
This division can be done randomly, or it can be done in order to meet the external constraints of the study.

Step 2: Train the subjects in the new technology
Training typically occurs during a lecture period in which the subjects are provided with the necessary information in order to use the new technology.

Step 3: One person in each pair applies the technology while the other observes.
The subject applying the technology is referred to as the *executor* while the subject observing application of the technology is called the *observer*. The job of the observer is not to work together with the executor

to apply the technology; rather the observer is tasked with ensuring that the executor faithfully follows the steps of the technology. The observer also takes notes about the executor's use of the technology including places where difficulties are encountered.

Step 4: Subjects within the pairs switch roles.
This step allows the subject who observed in Step 3 to now perform the task using the technology.

Step 5: Measurement and analysis of the second treatment
The goal of this step is that the results are more accurate because more of the learning curve has been addressed.

2.3 Technology to which the method is applied

As a "testbed" for evaluating this new training method, we used our ongoing research on software reading techniques. Software reading techniques are procedurally-based approaches that aim to improve the effectiveness of defect detection in software inspections by providing tailored and focused techniques for inspectors to use during the individual review phase. One specific type of reading technique that has been developed for reviewing software requirements documents written in English is Perspective-Based Reading (PBR) [2]. The theory behind PBR is that because a requirements document is used by a series of stakeholders, it must satisfy the differing needs of each of those stakeholders. To verify this property, PBR provides a method for each inspector to follow to assume the perspective of one of those stakeholders. Each PBR perspective asks its user to create a model that represents an abstraction of the requirements. The model is chosen so that it is relevant to the stakeholder. For example, an inspector assuming the perspective of a tester would create test cases as his or her model, while someone assuming the perspective of a designer would create a high-level design as the model. The technique for each perspective provides a step-by-step procedure that instructs the inspector on how to create the model and then provides a series of questions for the inspector to answer to look for defects. More information about PBR can be found in [17].

3. High level goals and hypotheses

Based on the results of some previous studies [6], there were still a series of unanswered questions about process experience:

- What type of process experience is important?
- How can an inspector gain process experience?
- What other types of experience affect process experience?

The goal of this study, therefore, was to apply the methodology from Section 2.2 in order to better understand the type and amount of experience with an inspection process that an inspector needed to possess to be effective. We wanted to determine if inspectors could gain this process experience in an inexpensive fashion. One main hypothesis in this study was:

> *Inspectors who observe an inspection before performing one will find more defects than inspectors who do not observe an inspection before performing one.*

We also hypothesized that the impact of this process experience would vary depending on the inspector's software development experience and their knowledge of the application domain of the requirements artifact.

4. The study

The remainder of this paper describes the study that was run to better understand whether process experience could be gained by observation. Complete details of the study, including the data and its complete analysis, can be found in [5].

4.1 Subjects

The subjects in this study were graduate students enrolled in a graduate level Software Engineering class at the University of Maryland in the Fall 2001 semester. The subjects worked in pairs to conduct two inspections, with one subject acting as the *executor* and the other as the *observer*, as defined in Section 2.1

Section 4.3.2 describes the external constraints that had to be met in grouping the 26 subjects into 13 pairs. For each relevant software development task in Table 1, the percentage of subjects who fell into each of the experience categories is shown. Where, *industrial* means that the subject has experience doing the task on at least one industrial project; *classroom* means the subject has learned about or has experience with the task in a classroom setting only; *none* means that the subject has not learned about or had experience with the task either in industry or in the classroom.

Table 1 – Experience levels of subjects

Experience / Dev. Tasks	Industrial	Classroom	None
Experience Writing Requirements	35%	39%	26%
Experience Writing Use Cases	19%	50%	31%
Experience Reviewing Requirements	38%	46%	16%
	High	**Low**	
Application Domain Knowledge	50%	50%	

4.2 Materials

The User perspective of PBR, which had the inspector create use cases as the abstraction of the requirements document, was applied to the requirements documents from two different systems: one for a Loan Arranger (LA) system, and one for an automated parking garage control system (PGCS). The LA system was responsible for organizing the loans held by a financial institution and bundling them for resale to investors. The PGCS was responsible for managing the open spaces in a parking garage and keeping track of the sales of reserved (monthly) tickets and non-reserved (daily) tickets. The LA requirements had 8 pages, 26 functional and 4 non-functional requirements, and 18 seeded defects. The PGCS requirements had 17 pages, 21 functional and 9 non-functional requirements, and 32 seeded defects.

4.3 Procedure

This section describes how each step of the methodology was implemented during the study.

4.3.1. Overview. Following the methodology described in Section 2.2, each pair of subjects performed two requirements inspections. During inspection 1, one team member gained process experience by observing his partner, before performing inspection 2. At the completion of the two inspections, each team wrote a report discussing the process they used to understand PBR, the feasibility of PBR and how the inspector's experience as an observer affected the second inspection. This report was the source of much of the qualitative data collected during the study.

A secondary goal of this study was to begin to understand the potential interaction between process experience and other types of experience. Specifically, we were interested in studying whether process experience had a different effect on subjects who were more experienced software developers vs. subjects who were less experienced software developers. Additionally, we wanted to study whether process experience had a different effect on subjects who were familiar with the domain of the requirements vs. subjects who were not familiar with the domain.

4.3.2. Step 1: Divide the subjects into two-person groups. In order to address the secondary goals, it was important to characterize the subjects based on their knowledge and make the pairings based on that characterization. Subjects were grouped so that the software development experience - process experience interaction and the application domain knowledge - process experience interactions could be measured.

First the subjects were blocked based on their software development experience into a group of highly experienced subjects (those that had industrial experience writing or reviewing requirements) and a group of low experienced subjects (those that had either no experience or only classroom experience writing or reviewing requirements). This blocking was done for two reasons. The first reason was so that the effect of process experience on subjects of different experience levels could be studied. The second reason was to eliminate a potential confounding variable that could arise if low and high experienced subjects worked together in the same pair. Specifically, there would have been the potential for the experienced subject, acting as the observer, to influence the performance of the inexperienced subject, acting as the executor.

In order to study the effects of application domain knowledge, it was assumed that the Loan Arranger (LA) domain was unfamiliar to the subject population and the Parking Garage Control System (PGCS) domain was much more familiar. Therefore, at least one member of each pair had to be knowledgeable in the PGCS domain and at least one member had to lack knowledge in the LA domain. Each member of the pair was assigned a document to review to satisfy this constraint.

After blocking the subjects, random pairings of the highly experienced subjects and random pairings of the low experienced subjects were made so that the domain knowledge assumption was ensured. Table 2 illustrates the study design that will be described in the following sections.

Table 2 – Study Design

Treatments	Group 1: 4 Low Experience Teams 3 High Experience Teams	Group 2: 3 Low Experience Teams 3 High Experience Teams
Review #1	**LA**	**PGCS**
	Switch Roles	*Switch Roles*
Review #2	**PGCS**	**LA**

4.3.3. Step 2: Train the subjects in the technology being studied. Subjects were trained in the PBR technique and the observational methods. The PBR training was done during a 60-minute class lecture that included the underlying PBR theory, the history and evolution of PBR and an explanation of the use of PBR along with some examples. The subjects were then given a chance, in class, to practice PBR and ask questions of the instructor.

The training in observational methods was done during a 30-minute class lecture consisting of an explanation of the specific responsibilities of roles of *process executor* and *process observer* and a short example. During the training, the subjects were instructed that when they performed their own inspections, the observer had to come up with his or her own set of questions to elicit information about the overall effectiveness of the PBR technique and the way in which it was applied (e.g. was the procedure too detailed? or was it missing key information?).

Finally, after the in-class training, each pair of subjects spent 45 minutes with one of the researchers. During this time the researcher watching the subjects use PBR to perform an inspection on a sample requirements document to ensure that each pair understood both the PBR technique as well as the roles of *observer* and *executor* by. Each subject spent part of this time as the observer and part of the time as the executor. The subjects were also given the opportunity to ask questions about PBR and the observer and executor roles.

4.3.4. Step 3: One person in each pair applies the technology while the second observes. A quasi-experimental, factorial design [4] with two treatments was used. In the first treatment, approximately half of the teams inspected the LA requirements and the other half inspected the PGCS requirements. During this inspection, one team member, the executor, followed the inspection technique to inspect the assigned requirements document while his or her partner, the observer, ensured the process was followed and took notes. To help ensure that the observer did not get involved in finding defects, he or she was given an opportunity at the conclusion of the inspection to add any defects to the list that he or she saw that the executor missed.

4.3.5. Step 4: Subjects within pairs switch roles. After performing the first inspection, the team members switched roles, i.e. the process observer in the first inspection became the process executor in the second inspection. For the second inspection, the teams were given the requirements document that they had not yet inspected (LA or PGCS).

4.3.6. Step 5: Measurement and analysis of the second treatment. Measurement was achieved by collection of both qualitative and quantitative data. The quantitative data included the amount of time required to perform the inspection using PBR, and the number and type of defects detected. The qualitative data, which was collected using the observational techniques and included in the report written by the subjects at the completion of the two treatments, included both direct observations made during the inspections as well as retrospective, or post-hoc, information. The observational data included:

- The subjective evaluation of the effectiveness of the technique.
- Any specific problems encountered with steps in the technique

The retrospective data included:

- The usefulness of the technique
- The practicality of the technique and whether it would be used again
- Any high-level problems with the technique

5. Results

The qualitative data from the study indicated that overall the subjects thought observation was beneficial, in terms of gaining understanding or confidence, or both:

- 11 of the 13 teams found the observation beneficial
  - Teams 1, 2, 4, 5, 10, 11, and 13 (7 of the 13 total teams) stated that observing the inspection process helped the second inspector better understand the overall process.
  - Teams 2, 3, 5, 6, 7, 8, and 13 (7 of the 13 teams) stated that observing the inspection process helped the second inspector better perform specific steps of building the use cases or detecting defects.
- 2 of the 13 teams (teams 9 and 12) did not comment at all on this issue.
- None of the teams indicated that the observation hurt their effectiveness.

On the other hand, the quantitative data, which was analyzed by comparing the percentage of defects found by the second group of inspectors (who observed the use of PBR before using it themselves) to the percentage of defects found by the first group of inspectors (who used

PBR without observing its use first), only shows statistically significant support for this conclusion in some cases. The box plots in Figures 1, 2, and 3 show the percentage of defects found by the subjects, grouped by the artifact inspected and inspection number. Figure 1 shows that there was overall little difference between the groups. Figure 2 shows that the low experienced subjects found more defects on the LA during inspection 2. Figure 3 shows that for the high experienced subjects there was not much difference between the groups for the PGCS document, and that subjects in inspection 1 found more defects than those in inspection 2 for the LA document.

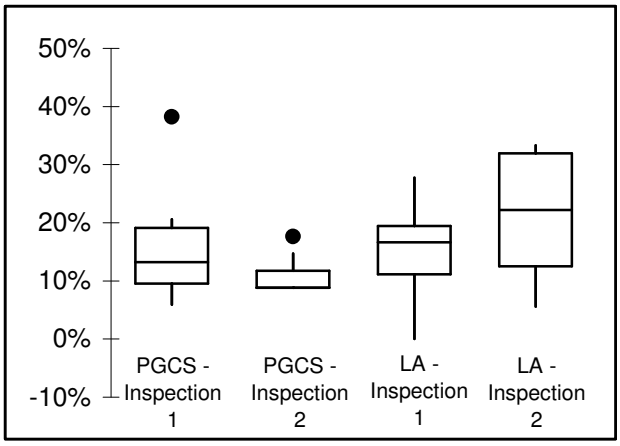

Figure 1 – Percentage of defects found (All inspectors)

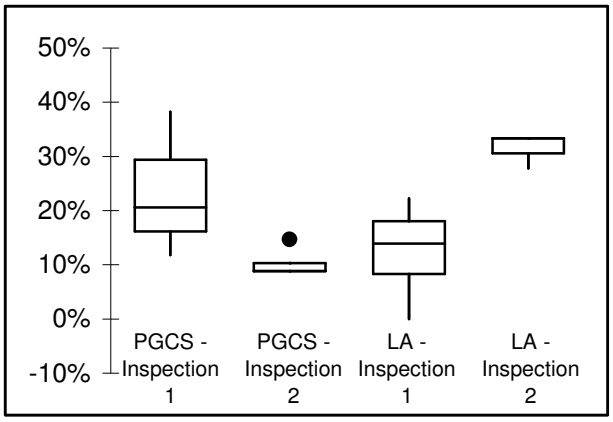

Figure 2 – Percentage of defects found (Low experience inspectors)

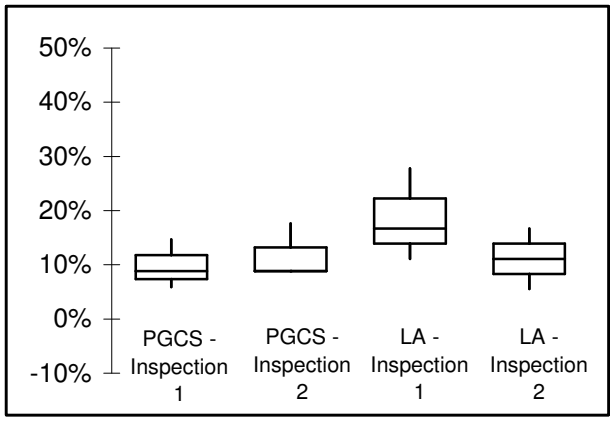

Figure 3 – Percentage of defects found (High experience inspectors)

The average percentages of defects found and the statistical analysis are presented in Table 3. Using a parametric t-test, there was a statistically significant (α = .01) improvement from inspection 1 to inspection 2 for subjects with low requirements experience who were inspecting the Loan Arranger (p = .01).

Table 3 - Percentage of defects found

Subjects	Inspection Without Observation	Inspection With Observation	p-value
All			
PGCS	16.7 %	10.9 %	.146
LA	15.0 %	21.3 %	.159
PGCS & LA	15.8 %	15.7 %	.979
Low Requirements Experience			
PGCS	23.5%	10.3%	.118
LA	12.5%	31.5%	.010
PGCS & LA	17.2%	19.4%	.369
High Requirements Experience			
PGCS	9.8%	11.2%	.322
LA	18.5%	11.1%	.148
PGCS & LA	14.2%	11.4%	.242

Additionally, based on the design of the study, those subjects who participated in inspection 2 on the LA artifact were observers of inspection 1 on the PGCS artifact, and vice versa. The data in Table 3 shows that low experienced subjects who observed inspection 1 on the PGCS artifact found more defects in the LA artifact during inspection 2 than the subjects who inspected the LA artifact in inspection 1 (note that this group found the most defects of any group inspecting the LA, so we can hypothesize that observing an effective inspection will help the observer to perform an effective inspection). On the other hand, the low experienced subjects who observed inspection 1 on the LA artifact found fewer defects in the PGCS artifact during inspection 2 than the subjects who inspected the PGCS artifact during

inspection 1. We therefore hypothesize that for inexperienced developers, the observation of a requirements inspection will be helpful for training when the artifact analyzed in the observed inspection is from a domain where he or she has high knowledge. Although this hypothesis covers only one of the four potential cases analyzed here, we believe this hypothesis to be plausible because during the inspection of an artifact on which the observer has high domain knowledge, the observer does not need to spend as much effort on understanding the artifact, and can spend more effort on understanding the inspection procedure therefore increasing his or her process experience. This hypothesis is also supported by the qualitative data which indicated that the subjects believed the observations helped.

6. Conclusion

The goal of this study was to understand whether or not subjects of a study run in a classroom setting could cost-effectively gain process experience such that the results of the study would be more applicable to an industrial setting. Based on the qualitative results from this study, the subjects believed that process experience was important. Additionally, the quantitative results showed that process experience was gained by observing an inspection in one of the two cases, the observation of the inspection of an artifact from a familiar domain. However, due to the confounding variables this result has not been shown conclusively.

In addition to these results, some new hypotheses about the usefulness of observation as a method for gaining process experience are proposed based on the results of this study:

1) Observing an inspection of an artifact from a domain where the inspector has high knowledge can be of more benefit than observing an inspection of an artifact from a domain where the inspector has low domain knowledge
2) Observation is not an effective way to gain process experience in general but is effective under certain conditions:
   - o Inspectors must observe more than one inspection. This hypothesis arises from the fact that the subjects indicated the observation was helpful, but the data did not support this observation.
   - o Inspectors must observe an expert in either the inspection process or the specific technology used or the application domain. This hypothesis arises from the fact that the low experienced subjects who observed an inspection on the PGCS, which was both a familiar domain and, based on the effectiveness, a well done inspection, found more defects when inspecting the LA than any other group. This argument is also made in the context of the learning that takes place in eXtreme programming [20].
   - o Inspectors must take a more "active" role during their observation. This hypothesis is similar to the argument made in the Active Design Review literature that for a design reviewer to fully understand the design, he must do something "active", such as construct a model [10].

In order to continue to understand this issue of the applicability of classroom studies to industrial environments, further studies should be run to test these new hypotheses. It is important to continue investigating methods that allow subjects of classroom studies to gain sufficient process experience in order to allow researchers to have more confidence in the applicability of their results to industrial settings.

Acknowledgements

We would like to thank the students of CMSC735 in the Fall Semester of 2001 at the University of Maryland for their participation in this study. We would also like to thank the reviewers for their helpful and insightful comments. We acknowledge support from the NSF Reader's Project (CCR-9900307) and from the NSF CeBASE Project (CCR-0088078).

References

[1] Arif, T., and Hegde, L.C. "Inspection of Object Oriented Construction: A study of Reading Techniques Tailored for Inspection of Design Models expressed in UML." Prediploma Thesis, Norwegian University of Science and Technology, Nov. 2001.

[2] Basili, V.R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sorumgard, S., and Zelkowitz, M.V. "The Empirical Investigation of Perspective Based Reading." *Empirical Software Engineering — An International Journal*, 1(2): 133-164, 1996.

[3] Bisant, D. and Lyle, J. "A Two-Person Inspection Method to Improve Programmer Productivity." *IEEE Transactions on Software Engineering* 15(10): 1294-1304, 1989.

[4] Campbell, D., and Stanley, J. *Experimental and Quasi-Experimental Designs for Research*. Houghton Mifflin Company, Boston, 1963.

[5] Carver, J., Shull, F., and Basili, V.R. "Investigating the Effects of Process Experience on Inspection Effectiveness." University of Maryland Technical Report CS-TR-4442, March 2003.

[6] Carver, J. *The Impact of Background and Experience on Software Inspections*. PhD Thesis, Computer Science Department, University of Maryland, 2003.

[7] Carver, J., Jaccheri, L., Moraso, S. and Shull, F. "Issues Using Students in Empirical Studies in Software Engineering Education." To appear in *Proceedings of 2003 International Symposium on Software Metrics (METRICS 2003).*

[8] Conradi, R. Parastoo Mohagheghi, Tayyaba Arif, Lars Christian Hegde, Geir Arne Bunde, and Anders Pedersen: "Object-Oriented Reading Techniques for Inspection of UML Models -- An Industrial Experiment", In *Proceedings of European Conference on Object-Oriented Programming (ECOOP'03).*

[9] Höst, M.; Regnell, B. and Wohlin, C. Using Students as Subjects: A Comparative Study of Students and Professionals in Lead-Time Impact Assessment. *Empirical Software Engineering – An International Journal*, 5(3): 201-214, 2000.

[10] Knight, J., Myers, E.A. "An Improved Inspection Technique". *Communications of the ACM*, 36(11): 51-61, 1993.

[11] Melo, W., Shull, F., and Travassos, G.H. "Software Review Guidelines." COPPE/UFRJ Systems Engineering and Computer Science Program Technical Report *ES-556/01*. September, 2001.

[12] Muller, M., and Tichy, W. "Case Study: Extreme Programming in a University Environment." In *Proceedings of 23rd International Conference on Software Engineering (ICSE 01)*, p. 537-544.

[13] Parnas, D.L., and Weiss, D.M. "Active Design Reviews: Principles and Practice." In *Proceedings of 8th International Conference on Software Engineering*, 1985, p. 132-136.

[14] Porter, A.A., and Johnson, P.M. "Assessing Software Review Meetings: Results of a Comparative Analysis of Two Experimental Studies." *IEEE Transactions on Software Engineering*, 23(3): 129-145, 1997.

[15] Sauer, C., Jeffery, D.R., Land, L., and Yetton, P. "The Effectiveness of Software Development Technical Reviews: A Behaviorally Motivated Program of Research." *IEEE Transactions on Software Engineering*, 26(1): 1-14.

[16] Shull, F. *Developing Techniques for Using Software Documents: A Series of Empirical Studies.* PhD Thesis, Computer Science Department, University of Maryland, 1998.

[17] Shull, F., Rus, I., and Basili, V.R. "How Perspective-Based Reading Can Improve Requirements Inspections." *IEEE Computer*, 33(7): 73-79, 2000.

[18] Shull, F., Carver, J., and Travassos, G.H. "An Empirical Methodology for Introducing Software Processes." In *Proceedings of the Joint 8th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-9)*, 2001. p. 288-296.

[19] Singer, J. and Lethbridge, T.C., "Methods for Studying Maintenance Activities." In *Proceedings of the 1st International Workshop on Empirical Studies of Software Maintenance (WESS 1996)*, p. 105-110.

[20] Spinellis, D. "Fear of Coding, and How to Reduce It." *IEEE Computer* 34(8): 100,98-99, 2001.

[21] Williams, L. "Integrating Pair Programming into a Software Development Process." In *Proceedings of the 14th Conference on Software Engineering Education and Training (CSEET 01)*, p. 27-36.

[22] Zhang, Zhijun. "The Design and Empirical Studies of Perspective-Based Usability Inspection." Ph..D. Thesis, University of Maryland, 1999.