An Empirical Study of Web-Based Inspection Meetings

Filippo Lanubile

Dipartimento di Informatica
University of Bari
Bari, Italy
lanubile@di.uniba.it

Teresa Mallardo

RCOST – Research Center on Software Technology
University of Sannio
Benevento, Italy
mallardo@unisannio.it

Abstract

Software inspections are a software engineering "best practice" for defect detection and rework reduction. In this paper, we describe an empirical evaluation with using a tool aiming to provide Internet groupware support for distributed software inspections. The tool is based on a restructured inspection process where inspection meetings have the only goal of removing false positives rather than finding additional defects. In place of face-to-face meetings, the tool provides web-based discussion forums and support for voting.

We present an empirical study of nine remote inspections which were held as part of a university course. We investigated whether all collected defects are worth to be discussed as a group. Results show that discussions for filtering out false positives (non true defects) might be restricted to defects which were discovered by only one inspector.

1. Introduction

Software inspection is an industry best practice for delivering high-quality software. The main benefit of software inspections derives from detecting defects early during software development and then reducing avoidable rework. Software inspections are distinguished from other types of peer reviews in that they rigorously define:

- a phased process to follow;
- roles performed by peers during review (e.g., moderator, author, recorder, reader, and reviewer[1]);
- a reading toolset to guide the review activity (e.g., defect taxonomies, product checklists, or scenario-based reading techniques);
- forms and report templates to collect product and process data.

[1] Some roles, such as reader and recorder, are defined specifically for the inspection meeting stage.

From the seminal work of Fagan [3] to its many variants [8], the software inspection process is essentially made up of six consecutive steps, as shown in Figure 1.

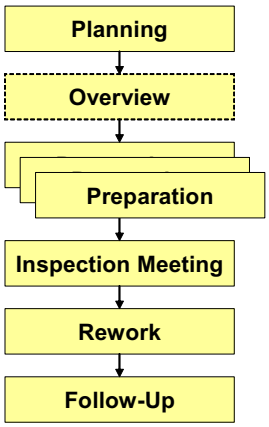

Figure 1. Conventional inspection process

During Planning, the moderator selects the inspection team, arranges the inspection material and sends it to the rest of the team, and makes a schedule for the next steps. During Overview, the moderator can optionally present process and product-related information for newcomers, if any. During Preparation, each inspector analyzes the document to become familiar with it and individually find potential defects. During the Inspection Meeting, all the inspectors encounter to collect and discuss the defects from the individual reviews and further review the document to find further defects. During Rework, the author revises the document to fix the defects. Finally, during Follow-Up the moderator verifies author's fixes, gives a final recommendation, and collects process and product data for quality improvement.

The main changes from the original Fagan's inspection have been a shift of primary goals for the Preparation and Inspection Meeting stages [8]. In order to make visible the quality of preparation prior to the meeting, the main

Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03)
0-7695-2002-2/03 $ 17.00 © 2003 IEEE

goal for Preparation has changed from pure understanding to defect detection, and so inspectors have to individually take notes of defects [5, 14]. Consequently, the main goal of the Inspection Meeting has been reduced from defect discovery, as a result of team analysis, to group consolidation of the defects individually found during Preparation.

In the attempt to shorten the overall cost and total time of the inspection process, the need for a meeting of the whole inspection team has been debated among researchers and practitioners. Parnas and Weiss first dropped the team meeting in their Active Design Reviews [15]. Then Votta [22] showed how defect collection meetings lengthened the elapsed time of software inspections at Lucent Technologies of almost one third, with defects discovered at the meeting (*meeting gains*) matched by defects not recorded at the meeting although found during preparation (*meeting losses*). Further studies [1, 2, 4, 9, 13, 17, 19] have also observed that the *net meetings improvement* (difference between meeting gains and meeting losses) was not positive, and then *nominal teams* (teams who do not interact in a face-to-face meeting) are at least equivalent to *real teams,* at a lower cost and time. However, meetings have been found useful for filtering out *false positives* (defects erroneously reported as such by inspectors), training novices, and increasing self-confidence [7, 9].

Based on the above empirical studies that argue the need for traditional meetings and on behavioral theory of group performance, Sauer et al. have proposed in [20] a reorganization of the inspection process to shorten the overall cost and total time of the inspection process. The alternative design for software inspections mainly consists of replacing the Preparation and Inspection Meeting phases of the classical inspection process with three new sequential phases: Discovery, Collection and Discrimination (see Figure 2).

The Discovery phase reflects the shift of goal for the Preparation phase that has changed from pure understanding to defect detection, and so inspectors are asked to individually take notes of defects.

The other two inspection phases are the result of separating the activities of defect collection (i.e., putting together defects found by individual reviewers) from defect discrimination (i.e., removing false positives), having removed the goal for team activities of finding further defects. The Collection phase is an individual task and requires either the moderator or the author himself. The Discrimination phase is the only phase where inspectors interact in a meeting. Sauer et al. suggest that the participation of the entire inspection team is not required; the number of discussants can be reduced to a minimal set, even a single expert reviewer paired with the author.

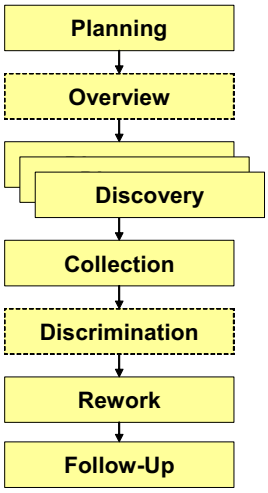

Figure 2. Reengineered inspection process

Another change for saving time and diminishing coordination overhead is introduced by skipping the Discrimination phase either entirely, passing all the collected defects directly to the author for rework, or partially, excluding from the discussion any potential defects (found by inspectors during the Discovery phase and merged in the Collection phase) that are considered to have high chances to be true defects. Sauer et al. suggest to select for the Discrimination phase only *unique defects*, that is defects which were found by only one inspector during the Discovery phase, while excluding *duplicates*, that is defects which were discovered by multiple inspectors and were merged during the Collection phase. To our knowledge, the entry criteria for the Discrimination phase have not been tested by means of empirical studies of software inspections but they are based on the behavioral theory of group performance and analogies with studies on audit reviews. Knowing which defects are worth of a discussion for discrimination purposes contributes to reduce inspection costs (because of less issues into the discussion agenda) without overwhelming the author with false positives that do not require rework.

In this paper we present an empirical study of software inspection aiming to assess the entry criteria for the Discrimination phase. Our empirical investigation is performed in the context of geographically distributed software inspections supported by an Internet-based tool specifically developed for the purpose.

The remainder of this paper is organized as follows. Section 2 presents the tool we used to support remote inspections. Section 3 describes the empirical study and Section 4 shows the results from data analysis. Finally, Section 5 summarizes findings and concludes with final remarks.

2. Tool Support for Distributed Inspection

The conventional process for software inspections hinders their applicability in the context of global software development, where software engineering activities are spread across multiple sites and even multiple countries [6]. In order to provide an Internet-based infrastructure for geographically distributed inspection teams, we developed a tool called the Internet-Based Inspection System (IBIS) [11, 12].

IBIS is mainly a web application to achieve the maximum of simplicity of use and deployment. All structured and persistent data are stored as XML files, programmatically accessed via the DOM API, and automatically manipulated by XSL transformations. All required groupware features are developed from dynamic web pages on the basis of scripts and server-side components. Event notification is achieved through automatic generation of emails.

Although Internet-based support makes it possible to reach skilled reviewers everywhere, it does not provide a process itself for the effective interaction of geographically dispersed inspection teams. Because of its roots on manual activities and face-to-face meetings, the conventional inspection process was considered inadequate to support distributed inspections, and thus we choose the reengineered inspection process [20], previously discussed, as the underlying model of software inspection.

In the following, we describe the use of the tool within the Collection and Discrimination phases. While the Discrimination phase is the object of interest for this paper, its entry criteria are defined in the Collection phase.

In the Collection stage, all the discovery logs from individual inspectors are collapsed into a unique defect inspection list (see Figure 3). The moderator can set identical defects from multiple inspectors as duplicates. Looking for duplicates is helped by the ability to sort the merged defect list with respect to location fields (e.g., document page number or requirement number) and reading support questions (i.e., which question in a checklist or a scenario was helpful for defect discovery). Collapsing duplicates from the collection of discovery logs is an iterative task (it can be performed over multiple sessions too).

Duplicates can be excluded from the Discrimination stage and let them go directly to the Rework stage. Looking at the inspection defects list the moderator may select which defects are worth to be discussed in the Discrimination stage and which inspectors will participate to the discussion. This decision can be supported by the display of inspectors' performance statistics, such as total number of reported defects and number of unique defects.

In the Discrimination stage, discussion takes place asynchronously as in a discussion forum. Each defect in the discrimination list is mapped to a threaded discussion (see Figure 4). Invited inspectors may add their comments inside the threads. To support decision making, discussants can also vote by rating any potential defect as true defect or false positive (see Figure 5). When a consensus has been reached, the moderator can mark potential defects as false positives, thus removing them from the list that will go to the author for rework (potential defects marked as false positives appear strikethrough in Figure 4).

Figure 3. Merging discovery logs

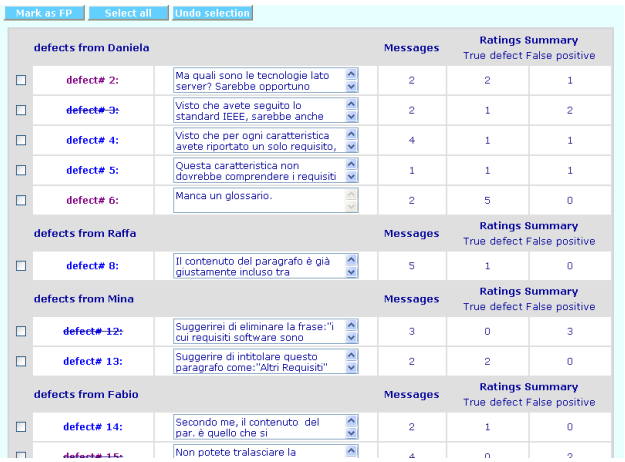

Figure 4. Defects included in the discrimination list

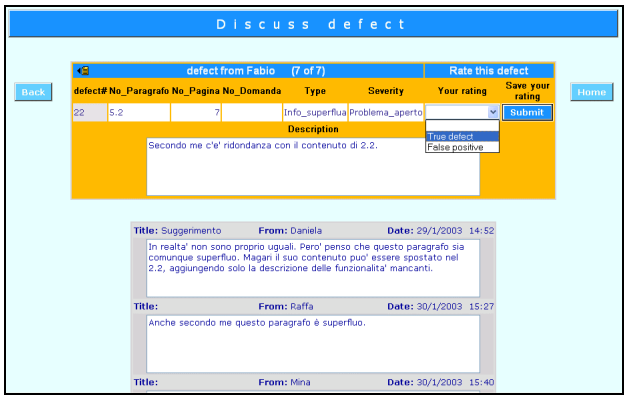

Figure 5. A discussion about a defect

3. The Empirical Study

We ran nine distributed inspections with participants interacting with the IBIS tool from university labs or home (no face-to-face meetings), thus reproducing the conditions of geographically dispersed teams.

Participants were 5th-year computer science students attending a web engineering course at the University of Bari. As a course assignment, students had to develop a web application, including documentation, working in groups of two or three people. The requirements documents of the nine student projects (ranging from 7 to 23 pages) were submitted for inspection and a member of the development team was selected to act as the author in the inspection. Because of the need to have a trained moderator, one of the researchers played the role of moderator for all the nine inspections. The rest of the inspection team was formed by two or three external (to the class) reviewers plus a student who was randomly selected from the class.

Table 1 summarizes the intermediate results of the nine inspections at the end of the Collection stage, when all the defects individually found have been merged in a single list, including duplicates. The Discrimination stage was planned to include all the collected defects (both unique and duplicated defects) and invite the entire inspection team to the discussion.

Focusing on the Discrimination stage, where collected potential defects are discussed with the main goal of discriminating true defects from false positives (to be removed), we looked for answers to the following questions.

3.1. Decision Making

Q1 Are there differences between unique defects and duplicates with respect to decision making?

Based on findings from previous studies [1, 10, 20], our hypothesis was that plurality effects apply: duplicates (defects found by more than one inspector) are more likely to be accepted as true defects than unique defects (defects found by only one inspector). Answering this question can lead to reduce the list of potential defects to be discussed as a group, thus saving cumulative team effort and shrinking elapsed time.

We measured the following variables (measures have been normalized):

- % unique defects removed as FP: ratio of unique defects marked as "false positive" to total number of unique defects;
- % duplicates removed as FP: ratio of duplicates marked as "false positive" to total number of duplicates;

Table 1: Inspections before entering Discrimination stage

Inspection ID	Insp1	Insp2	Insp3	Insp4	Insp5	Insp6	Insp7	Insp8	Insp9
Inspection team size	6	6	6	6	5	5	5	5	4
Defects individually recorded (at Discovery)	53	60	39	76	35	36	52	24	29
Average discovery effort	54 min	1 h	48 min	2 h	35 min	1 h 7 min	1 h 30 min	1 h 22 min	1 h 33 min
Defects merged and selected for discrimination (at Collection)	33	37	28	52	28	22	41	15	19
Unique defects (found by only one inspector)	20	25	21	38	24	15	34	9	12
Duplicates (found by multiple inspectors)	13	12	7	14	4	7	7	6	7
Collection effort	2 h 9 min	2 h 25 min	1 h 30 min	2 h 15 min	1 h 15 min	2 h	1 h 30 min	1 h	1 h 30 min

3.2. Voting

Q2 Are there differences between unique defects and duplicates with respect to voting?

In the IBIS tool, it is the moderator to mark potential defects as false positives, which are then removed from the list of defects going to the Rework stage to be fixed. However, the moderator should take decisions with the consensus of the inspection team. An approach to assess the degree of consensus is looking at inspectors' votes, if any. A vote is a ballot between "true defect" (TD) and "false positive" (FP).

We measured the following variables (measures have been normalized):

- votes as FP per unique defect: ratio of votes on unique defects in favor of "false positive" to total number of unique defects;
- votes as FP per duplicate: ratio of votes on duplicates in favor of "false positive" to total number of duplicates.
- votes as TD per unique defect: ratio of votes on unique defects in favor of "true defect" to total number of unique defects;
- votes as TD per duplicate: ratio of votes on duplicates in favor of "true defect" to total number of duplicates;

3.3. Discussion intensity

Q3 Are there differences between unique defects and duplicates with respect to discussion?

Although voting is an unequivocal method for communicating intentions, votes without an explicit exchange of messages among discussants would not be helpful to support moderator's decisions, and will be ignored while taking a decision about a potential defect.

We measured the following variables (measures have been normalized):

- discussion intensity on unique defects: ratio of posted messages on unique defects to total number of unique defects;
- discussion intensity on duplicates: ratio of posted messages on duplicates to total number of duplicates;

3.4. Contribution to discussion

Q4 Are there differences between participants with respect to discussion?

No value would be gained by letting passive participants to affect moderator's decisions by means of "silent" votes. Answering this question can lead to identify the critical group size for the discrimination task, above which discussants do not actively contribute to decision making. This implies saving inspection costs by reducing the number of inspectors to be invited for discussion.

We measured the following variables:

- messages from moderator: number of messages with the moderator as sender;
- messages from author: number of messages with the author as sender;
- messages from the most active reviewer: number of messages sent by the reviewer (neither moderator or author) who was the most active discussant with respect to other reviewers (except moderator and author).

4. Data Analysis

In order to answer the first three research questions, we need to compare a couple of variables (the former for unique defects and the latter for duplicates) which are measured in the same sample of cases. Because our sample is very small (nine inspection teams) and we could not rely on the normality assumption, we used the Wilcoxon's matched pairs test as a nonparametric alternative to the *t*-test for dependent samples. The Wilcoxon's matched pairs test only assumes that the variables to be compared are on an ordinal scale and that the differences between the two variables can be rank ordered too. Analogously, to answer the fourth research question we need to compare three variables which are measured in the same sample of cases. In this case we used the Friedman ANOVA by ranks test as a nonparametric alternative to a one-way repeated measures analysis of variance. The Friedman ANOVA assumes that the variables are measured on at least an ordinal scale. The null hypothesis is that the variables contain samples drawn from the same population, and then identical medians.

We run a total of five tests. In order to lower the probability of getting a significant result purely by chance, we control the level of significance for a set of tests through the Dunn-Bonferroni procedure [23]. Briefly, an experimenter may obtain the significance level for a single test as $\alpha ind = \alpha expw / m$, where $\alpha expw$ is the desired level of significance for the entire empirical study and m is the number of tests in the study. In our case, if we set $\alpha expw$ to 0.05, we will need a p-value less than 0.01 ($\alpha ind = 0.05 / 5$) to conclude that a single test has found a significant difference.

For the first question (are there differences between unique defects and duplicates with respect to decision making?) we compared the percentage of unique defects removed as false positives with the percentage of duplicates removed as false positives. Figure 6 shows

multiple bar plots for the two variables as measured in the nine inspections. We found a significant difference between the two variables (p = 0.0077), that is the proportion of unique defects that were rejected as false positives was higher than for duplicates.

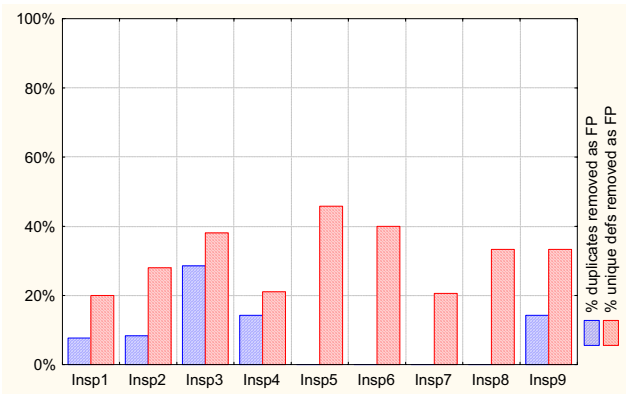

Figure 6. Duplicates and unique defects removed as false positives

For the second question (are there differences between unique defects and duplicates with respect to voting?) we performed two tests to compare the votes as "false positive" (or "true defect") per unique defect with the votes as "false positive" (or "true defect") per duplicate.

Figure 7 and Figure 8 show multiple bar plots for the two couples of variables. The first test (about votes as FP) showed a significant difference (p = 0.0077) between the two variables, with more votes as false positives per unique defect than per duplicate. On the contrary, the other test (about votes as TD) failed to reveal any significant difference between the two variables (p = 0,0663). This can be explained with discussants being more active in expressing affirmative votes (i.e., "this is a true defect") rather than negative votes (i.e., "this is not a true defect"), and with true defects being more than false positives both for unique defects and for duplicates.

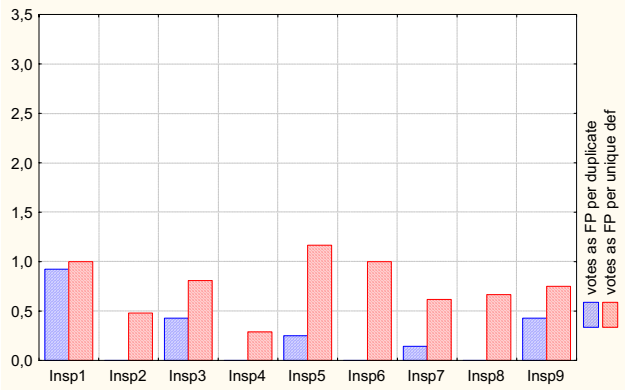

Figure 7. Votes as false positive per duplicate and unique defect

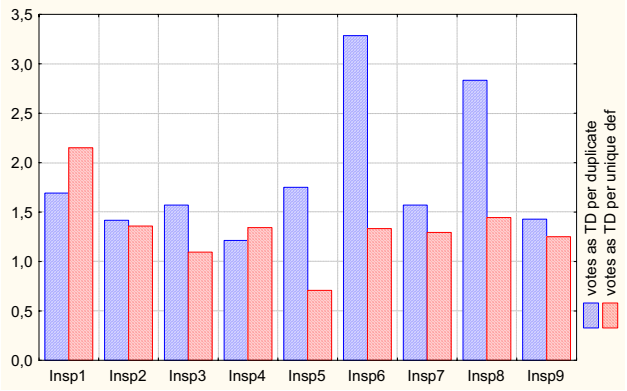

Figure 8. Votes as true defect per duplicate and unique defect

For the third question (are there differences between unique defects and duplicates with respect to discussion?), we compared the discussion intensity on duplicates with that on unique defects. Figure 9 shows multiple bar plots for the two variables. The test failed to reveal a significant difference between the two variables (p = 0.0506), that is messages per discussion thread did not differ between duplicates and unique defects.

For the fourth question (are there differences between participants with respect to discussion?) we analyzed posted messages with respect to the sender. Figure 10 shows multiple bar plots for three variables, respectively messages from moderator, author, and the most active reviewer. The test found a significant difference between the three variables (p = 0.0043), with messages from both moderator and author being more frequent than messages from the most active reviewer, and then from every other reviewer.

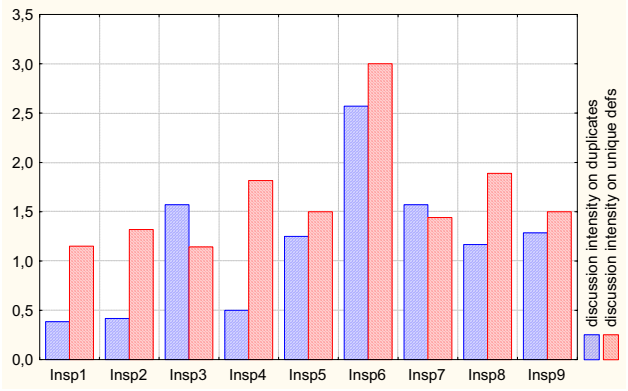

Figure 9. Discussion intensity on duplicates and unique defects

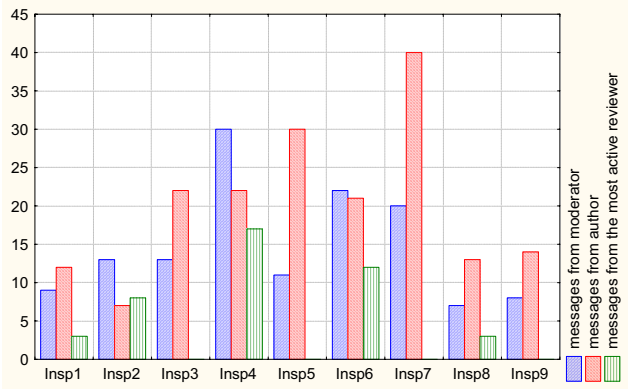

Figure 10. Posted messages per sender

5. Conclusions

In this paper we have investigated the entry criteria (which potential defects to select and which participants to invite) for web-based inspection meetings, where inspectors discriminate true defects from false positives. Tool support provides interacting groups with asynchronous electronic meetings in place of face-to-face meetings.

We specifically tested the hypothesis that defects individually found by multiple inspectors (duplicates) are accepted as true defects in a group discussion, and then can skip the Discrimination stage. So, we measured the performance of the Discrimination stage, as applied to potential defects collected from individual inspectors, and looked at differences between duplicates and unique defects (defects identified by only one reviewer).

We found that unique defects had higher chances than duplicates to be identified as false positives (conversely, duplicates had higher chances to be accepted as true defects). We found that decisions about false positives were actually supported by group consensus, as expressed by negative acknowledgments (votes as false positives) being proportionally higher for duplicates than for unique defects. On the other hand, we did not find significant differences between duplicates and unique defects with respect to discussion intensity.

Our findings are consistent with the hypothesis of considering duplicates unworthy a group meeting for the purpose of discrimination, thus limiting inspection team effort devoted to discussion. However, we cannot exclude that inspectors could benefit from discussing about all potential defects (including both duplicates and unique defects) for the purpose of joint learning.

We also found that most of the group discussion consisted of messages sent by either the moderator or the document's author. The other inspectors were less active in the discussion and mainly expressed their judgments by means of electronic votes without complementary messages. Then, our findings support the proposal in [20] of limiting discrimination meetings to a couple of discussants (in our case, the moderator and the author). Restricting the group size (other than the list of issues to discuss) for the discrimination task reduces the team effort that a project manager should allocate for running inspections.

For our empirical study, we can identify the following threats to external validity that limit the generalization of these findings to the industrial practice of software inspections.

Representative subjects. Since we involved students both as documents' authors and as reviewers, they may not be representative of the population of software professionals. Our fifth-year students can be considered equivalent to newcomers that are usually recruited in inspection teams for learning purposes. This threat is also mitigated by the presence of three skilled reviewers in each inspection team, who had been specifically trained on requirements engineering and inspection process, and had performed various inspections in the past.

Representative artifacts. The requirements documents inspected in this study may not be representative of industrial requirements documents. Our documents were requirements specifications for web applications while inspections are often conducted for dependable systems where quality and rework costs are perceived as critical.

Representative processes. Tool-supported distributed inspections in this study, based on a reengineered inspection process, may not be representative of industrial practice. Although software inspections are often identified with the Fagan's model [3], there are actually many variants of the inspection process which have been applied in industry and have been reported in the literature [8, 18]. Tool-supported inspections have also gained industrial adoption [16, 21].

How representative any of our findings are can only be determined by conducting further replications.

Acknowledgments

We gratefully acknowledge the collaboration of Fabio Calefato, Mina Di Bari, and Raffaella Massaro in the development of the tool for distributed inspection and the execution of the experiment. Thanks also to all the students who participated to the remote inspections.

References

[1] A. Bianchi, F. Lanubile, and G. Visaggio, "A controlled experiment to assess the effectiveness of inspection meetings", *Proc. of METRICS 2001*, London, United Kingdom, April 2001, pp.42-50.

[2] M. Ciolkowksi, C. Differding, O. Laitenberger, and J. Munch, "Empirical Investigation of Perspective-based Reading: A Replicated Experiment", ISERN Report 97-13, 1997.

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

[4] P. Fusaro, F. Lanubile, and G. Visaggio, "A Replicated Experiment to Assess Requirements Inspection Techniques", *Empirical Software Engineering*, 2: 39–57, 1997.

[5] T. Gilb, and D. Graham, *Software Inspection*, Addison-Wesley, Reading, MA, 1993.

[6] J. D. Herbsleb, and D. Moitra, "Global Software Development", *IEEE Software*, 18(2): 16-20, 2001.

[7] P. M. Johnson, and D. Tjahjono, "Does Every Inspection Really Need a Meeting?", *Empirical Software Engineering*, 3: 9-35, 1998.

[8] O. Laitenberger, and J.M. DeBaud, "An Encompassing Life Cycle Centric Survey of Software Inspection", *The Journal of Systems and Software*, 50: 5-31, 2000.

[9] L. P. W. Land, R. Jeffery, and C. Sauer, "Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews: Report of a Replicated Experiment", Caesar Technical Report 97/2, Univ. of New South Wales, 1997.

[10] L. P. W. Land, C. Sauer, R. Jeffery, "The Use of Procedural Roles in Code Inspections: An Experimental Study", *Empirical Software Engineering*, 5(1): 11-34, March 2000.

[11] F. Lanubile, and T. Mallardo, "Tool Support for Distributed Inspection", *Proc. of COMPSAC 2002*, Oxford, UK, 2002.

[12] F. Lanubile, and T. Mallardo, "Preliminary Evaluation of Tool-based Support for Distributed Inspection", *Proc. of the ICSE Int. Workshop on Global Software Development*, Orlando, FL, USA, 2002.

[13] J. Miller, M. Wood, and M. Roper, "Further Experiences with Scenarios and Checklists", *Empirical Software Engineering*, 3: 37–64, 1998.

[14] National Aeronautics and Space Administration, Software Formal Inspection Guidebook, Technical Report NASA-GB-A302, 1993. Available at http://satc.gsfc.nasa.gov/fi/fipage.html

[15] D. L. Parnas and D. M. Weiss, "Active Design Reviews: Principles and Practice", *Journal of Systems and Software*, 7: 259–265, 1987.

[16] J. M. Perpich, D. E. Perry, A. Porter, L. Votta and M. W. Wad, "Anywhere, anytime code inspections: using the Web to remove inspection bottlenecks in large-scale software development", *Proc. of the 19th ICSE*, Boston, USA May 1997, pp. 14-21.

[17] A. Porter, L. G. Votta, and V. R. Basili, "Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment", *IEEE Trans. on Software Engineering*, 21(6): 563–575, June 1995.

[18] A. Porter, H. Siy, A. Mockus, and L. Votta, "Understanding the sources of variation in software inspections", *ACM Trans. on Software Engineering and Methodology*, 7(1): 41-79, 1998.

[19] A. Porter, and L. Votta, "Comparing Detection Methods for Software Requirements Specification: A Replication Using Professional Subjects", *Empirical Software Engineering*, 3: 355-379, 1998.

[20] C. Sauer, D. R. Jeffery, L. Land, and P. Yetton, "The effectiveness of software development technical reviews: A behaviorally motivated program of research", *IEEE Trans. on Software Engineering*, 26(1): 1–14, 2000.

[21] M. van Genuchten, C. van Dijk, H. Scholten, and D. Vogel, "Using Group Support Systems for Software Inspections", *IEEE Software*, 18(3) : 60-65, 2001

[22] L. G. Votta, "Does Every Inspection Need a Meeting?", *ACM Software Engineering Notes*, 18(5): 107–114, December 1993.

[23] B. J. Winer, D. R. Brown, K. M. Michels, *Statistical Principles in Experimental Design*, third edition, McGraw-Hill, New York, 1991.