



Title	Comparing Reading Techniques for Object-Oriented Design Inspection
Author(s)	Sabaliauskaite, Giedre; Kusumoto, Shinji; Inoue, Katsuro
Citation	Transactions on Information and Systems. 2004, E87-D(4), p. 976-984
Version Type	VoR
URL	https://hdl.handle.net/11094/50959
rights	copyright©2004 IEICE
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

PAPER

Comparing Reading Techniques for Object-Oriented Design Inspection

Giedre SABALIAUSKAITE^{†a)}, Student Member, Shinji KUSUMOTO^{††}, and Katsuro INOUE^{††}, Members

SUMMARY For more than twenty-five years software inspections have been considered an effective method for defect detection. Inspections have been investigated through controlled experiments in university environment and industry case studies. However, in most cases software inspections have been used for defect detection in documents of conventional structured development process. Therefore, there is a significant lack of information about how inspections should be applied to Object-Oriented artifacts, such as Object-Oriented code and design diagrams. In addition, extensive work is needed to determine whether some inspection techniques can be more beneficial than others. Most inspection experiments include inspection meetings after individual inspection is completed. However, several researchers suggested that inspection meetings may not be necessary since an insignificant number of new defects are found as a result of inspection meeting. Moreover, inspection meetings have been found to suffer from process loss. This paper presents the findings of a controlled experiment that was conducted to investigate the performance of individual inspectors as well as 3-person teams in Object-Oriented design document inspection. Documents were written using the notation of Unified Modelling Language. Two reading techniques, namely Checklist-based reading (CBR) and Perspective-based reading (PBR), were used during experiment. We found that both techniques are similar with respect to defect detection effectiveness during individual inspection as well as during inspection meetings. Investigating the usefulness of inspection meetings, we found out that the teams that used CBR technique exhibited significantly smaller meeting gains (number of new defect first found during team meeting) than meeting losses (number of defects first identified by an individual but never included into defect list by a team); meanwhile the meeting gains were similar to meeting losses of the teams that used PBR technique. Consequently, CBR 3-person team meetings turned out to be less beneficial than PBR 3-person team meetings.

key words: software inspection, reading technique, controlled experiment, unified modelling language

1. Introduction

For more than twenty-five years software inspections have been considered an effective and efficient method for defect detection [7], [8], [16], [26]. Inspections have been extensively investigated through controlled experiments in university environment and industry case studies. However, in most cases software inspections have been used for defect detection in documents of conventional structured development process, such as functional requirement documents or

code modules [1], [2], [6], [13], [15], [18], [20]. There is a significant lack of information about how inspections should be applied to Object-Oriented (subsequently denoted OO) artifacts, such as OO code and design diagrams, because inspections were developed when the structured development process was dominant. Since over the past decade OO development methods have replaced conventional structured methods, it is very important to adapt the existing inspection techniques and develop new ones for OO artifact inspection.

Different reading techniques, which provide guidelines on how to examine software artifacts and identify defects, are used during inspection. The most popular are *Ad hoc* and Checklist-based reading (subsequently denoted CBR) techniques [13]. The *Ad hoc* reading technique does not provide any instructions for the inspector on how to proceed during defect detection activity. CBR [5] provides the inspector with a checklist, which consists of procedural guidelines and “yes/no” questions. The inspector has to answer those questions while reading the software document. One more approach is the Scenario-based reading [1], [12], [14]. It provides inspector with a scenario, describing how to proceed and what to look for during inspection. Several variants of Scenario-based reading have been proposed: Defect-based reading [18], Reading technique based on Function Points [4] and Perspective-based reading (subsequently denoted PBR) [1], [12], [14], [21], [22]. The Defect-based reading concentrates on specific defect classes. The Function Point based reading technique focuses on specific function point elements. PBR focuses on the perspectives (points of view) of the users of software documents.

Most inspection experiments include inspection meetings after individual inspection is completed. During a meeting, inspectors gather in a team to discuss the inspection product and to detect as many defects as possible. The traditional thinking has been that inspection teams are especially effective as a result of their interactions [7]. However, several researchers suggested that inspection meetings may not be necessary [11], [19], [27] since an insignificant number of new defects are found as a result of inspection meeting (often referred to as meeting gains or “synergy”) [18]. Moreover, inspection meetings have been found to suffer from process loss – the phenomenon by which defects identified by individual inspectors are not included into the list during meeting [18].

Previously, we designed and conducted an experiment to test whether individual inspectors who use PBR technique outperform the ones who use CBR technique [22] in inspec-

Manuscript received July 10, 2003.

Manuscript revised September 18, 2003.

[†]The author is with the Graduate School of Engineering Science, Osaka University, Toyonaka-shi, 560-8531 Japan; and Information Systems Department, Kaunas University of Technology, Kaunas 3026, Lithuania.

^{††}The authors are with the Graduate School of Information Science and Technology, Osaka University, Toyonaka-shi, 560-8531 Japan.

a) E-mail: giedre@ics.es.osaka-u.ac.jp

tion of Object-Oriented design documents, written using the notation of Unified Modelling Language (subsequently denoted UML) [3]. We will subsequently refer to it as Experiment 1. The results of the Experiment 1 indicated that the subjects who used PBR had similar effectiveness and greater cost per defect as compared to the subjects who used CBR.

This paper explains the second experiment (which we refer to as Experiment 2) conducted in July 2002 in Osaka University with 54 subject students. The goals of Experiment 2 were twofold: to verify the results of Experiment 1, and to investigate the usefulness of inspection meetings. Individual inspections, as well as inspection meetings were performed. The following elements were the same for both experiments: inspection techniques, inspection objects, defect types, defect registration forms.

The results of the individual inspection stage confirmed the results of the Experiment 1, i.e. cost per defect of PBR subjects was higher as compared to CBR subjects, and the effectiveness of CBR and PBR subjects was similar. Furthermore, there was no statistically significant difference between 3-person CBR and PBR teams with respect to defect detection effectiveness and cost per defect. Regarding the meeting gains of 3-person inspection teams, CBR teams exhibited significantly smaller meeting gains than meeting losses; meanwhile the meeting gains were similar to meeting losses of PBR teams. Consequently, CBR 3-person team meetings turned out to be less beneficial than PBR 3-person team meetings. The results of this research can be significant for other software inspection researchers and software developers, because to the best of our knowledge there is a lack of information how the inspections should be applied to UML diagrams.

The structure of the paper is as follows. Section 2 gives an explanation of inspection process and two reading techniques – CBR and PBR. Section 3 motivates the hypotheses stated for this experiment. Section 4 describes the design and operation of the experiment. Threats to validity are described in Sect. 5. Section 6 presents the analysis of experiment data. In Sect. 7, the results are discussed, and conclusions are given in Sect. 8.

2. Preliminary

This section describes inspection process, and two reading techniques, used during experiment.

2.1 Inspection Process

Software inspection, as a structured process, has first been described by Fagan [7]. It consists of five stages:

1. *Overview*. The author presents an overview of the scope and purpose of the work product.
2. *Preparation (or Individual inspection)*. Inspectors analyze the work product individually with the goal of understanding it thoroughly.
3. *Inspection meeting*. The inspection team assembles to

discuss the inspection product and to detect as many defects as possible under the direction of a moderator.

4. *Rework*. The author revises the work product.
5. *Follow-up*. The moderator verifies the quality of rework and decides if reinspection is required.

2.2 Checklist-Based Reading

CBR has been a commonly used technique in inspections since 1970's. Checklists are based on a set of specific questions that are intended to guide the inspector during inspection.

In a survey of software inspection [13] the authors discuss a list of weak points of CBR. First, the questions are often general and not sufficiently tailored to a particular development environment. Secondly, concrete instructions how to use the checklist are often missing, i.e. it is unclear when and based on what information an inspector is to answer a particular checklist question. Finally, the questions of the checklist are often limited to detection of particular defect types (inspectors may not focus on defect types not previously detected and, therefore, they may miss whole classes of defects).

We have developed a checklist for UML diagram inspection, based on the structure presented by Chernak [5] taking into consideration the weakness of CBR discussed in Ref. [13]. The checklist consists of two components: "Where to look" (a description where to search for defects) and "How to detect" (a list of questions that should help the inspector to detect defects). The checklist contained 20 questions, following with the recommendations of Gilb and Graham [8] (checklist should not be longer than a page, i.e. approximately 25 items). An excerpt from the checklist (translation from the Japanese language) is given in Fig. 1.

2.3 Perspective-Based Reading

The main idea of the PBR technique is that a software product should be inspected from the perspective of different stakeholders [1], [12], [14], [21], [22]. The perspectives depend on the roles that people have within the software development and the maintenance process. For examining a document from a particular perspective, PBR technique provides guidance for the inspector in the form of a PBR scenario.

The PBR scenario [12], [14] consists of three main sections: *introduction* (describes the quality requirements, which are most relevant to this perspective); *instructions* (describe what kind of documents to use, how to read, how to extract the necessary information), and *questions* (set of questions which inspector has to answer during the inspection). The main objective of using instructions for reading a document from different perspectives is to gain a better defect detection coverage of a software artifact.

The benefits of PBR have been summarized by Shull et al. [23]:

CHECKLIST		
Locate the following diagrams: Class Diagrams, Activity Diagrams, Sequence Diagrams and Component Diagrams. Answer the questions related to corresponding diagrams. When you detect a defect, mark it on the diagram and fill the necessary information in the Individual defect registration form. If you have any comments, write them in Comment form.		
CLASS DIAGRAMS		
1	Aren't there any inconsistencies between Class Diagrams and Requirements Specification?	Yes No
2	Are all the necessary Classes and Associations defined?	Yes No
3	Are there no redundant elements in Class Diagrams?	Yes No
4	Is the multiplicity of all Associations defined?	Yes No
5	Do Class Diagrams have enough information for code development?	Yes No
...		
SEQUENCE DIAGRAMS		
10	Aren't there any inconsistencies among Sequence Diagrams, Class Diagrams and Requirements Specification?	Yes No
11	Are all the necessary Objects and Messages shown?	Yes No
12	Is every Class from Class Diagrams included in any of Sequence Diagrams and vice versa?	Yes No
13	Is every Use Case from Use-Case Diagrams implemented in one of Sequence Diagrams?	Yes No
14	Are there no redundant elements in Sequence Diagrams?	Yes No
...		

Fig. 1 Excerpt from the checklist.

- *Systematic*. PBR identifies the different uses of documents (perspectives), and a procedure (steps) for verifying whether those uses are achievable.
- *Focused*. PBR helps inspectors to concentrate more effectively on certain types of defects, rather than having to look for all possible types.
- *Goal-oriented and customizable*. Different perspectives can be used to reflect specific goals, and PBR can be easily customized to a specific project or organization.
- *Transferable via training*. Because PBR works from a definite procedure, and not the inspector's own experience with recognizing defects, new inspectors can receive training in the procedure's steps while applying the technique.

Contrarily, CBR does not help the reader focus on a particular aspect of the document. Inspectors have to find all types of defects in the entire document. Consequently, it is less systematic and focused, and therefore is expected to be less effective than PBR.

We used three perspectives in this experiment: User's, Designer's and Implementer's. The concern of the User is to ensure that the specification of the system operation at the end of analysis stage is complete, error-free and that satisfies

DESIGNER'S SCENARIO	
Assume that you are a Designer of the system. The concern of the Designer is to define the static structure of the system (Class diagrams) as well as to ensure that the required behaviour is achieved in terms of interactions between objects (Sequence diagrams). Please perform tasks following Step1 through Step3...	
...	
Step 3	Locate Class Diagrams and Sequence Diagrams
Sequence Diagrams show interactions between Objects in the system. Those interactions take place in a certain sequence at appropriate time. Please make a list of all Objects, included into Sequence Diagrams. Compare the list with Class Diagrams. Make sure, that all the Objects from Sequence Diagrams are defined in Class Diagrams. Examine the Messages between Objects in the Sequence Diagram to make sure that they are defined as Methods or Attributes of the corresponding Class in Class Diagrams as well. Examine the relationships between Objects in Sequence Diagrams. Make sure, that the relationship between two Objects, which exists in Sequence Diagram, exists between the same Object Classes in Class Diagrams as well. After that, answer the following questions.	
3.1	Are all the Objects of Sequence Diagrams defined in Class Diagrams?
3.2	Are all the Messages between objects in Sequence Diagram defined as Methods or Attributes of the corresponding Class in Class Diagrams?
3.3	Does the relationship between two Objects, which exists in Sequence Diagram, exists between the same Object Classes in Class Diagrams as well?
3.4	Are there no redundant or missing elements in Sequence diagrams?

Fig. 2 Excerpt from the Designer's scenario.

user's requirements. It means that there must be no inconsistencies between the various analysis models, such as Requirements specification, Use-Case, Activity and Sequence diagrams. The concern of the Designer is to define the static structure of the system (Class diagrams) as well as to ensure that the required behaviour is achieved in terms of interactions between objects (Sequence diagrams). The concern of the Implementer is to ensure that the system design is consistent, complete and ready for transferring from design into code. Implementation needs have to be completely satisfied in Class, Sequence and Component diagrams. Three scenarios have been developed: User's, Designer's and Implementer's. An excerpt from the Designer's scenario is given in Fig. 2.

3. Experimental Hypotheses

We stated two types of hypotheses before the experiment: hypotheses for individual inspectors, and hypotheses for 3-person inspection teams.

3.1 Hypotheses for Individual Inspectors

The results of Experiment 1 [22] indicate that subjects who use PBR inspection technique have similar effectiveness and greater cost per defect as compared to the subjects who use CBR. We decided to test the same null hypotheses related to defect detection effectiveness and cost per defect in Experiment 2, because: a) we wanted to compare the results of Experiment 1 and Experiment 2; b) some of the elements used in Experiment 2 were different from those used in Experiment 1 (subjects, inspection time), so the results might be different. Therefore, we considered that we should test the same assumptions (H_{01} and H_{02}). We assumed (H_{01}) that PBR defect detection effectiveness should be different from CBR defect detection effectiveness; and we assumed (H_{02}) that subjects who used PBR should have higher cost per defect. The following null hypotheses have been stated:

H_{01} : There is no difference in defect detection effectiveness of subjects who use PBR inspection technique as compared to subjects who use CBR;

H_{02} : Cost per defect of subjects who use PBR is lower than cost per defect of subjects who use CBR.

3.2 Hypotheses for 3-Person Inspection Teams

Performing various tasks during PBR is expected to result in better understanding of the document. Therefore, during the meeting, inspectors do not have to spend a lot of extra effort in explaining the defects that they found to their colleagues in the team. In order to test whether it has any influence on team defect detection effectiveness and cost per defect, we stated two hypotheses (H_{03} and H_{04}). We assumed that team defect detection effectiveness (H_{03}) and cost per defect (H_{04}) should be different for CBR and PBR 3-person inspection teams. The following null hypotheses have been stated:

H_{03} : There is no difference in defect detection effectiveness of 3-person inspection teams that used PBR technique during individual inspection as compared to the teams that used CBR technique;

H_{04} : There is no difference in cost per defect of 3-person inspection teams that used PBR technique during individual inspection as compared to the teams that used CBR technique.

The results of several experiments to investigate the usefulness of inspection meeting are contradictory. Fagan reported that the number of defects identified in meeting was much higher than in individual work [7]. However, Porter et al. [18] reported that inspection meeting produced no net improvement in the fault detection rate – meeting gains were offset by meeting losses. Based on those results, we stated two hypotheses (H_{05} and H_{06}) in order to evaluate if meeting gains are greater than meeting losses during CBR (H_{05}) and PBR (H_{06}) team meetings. A “meeting gain” occurs when a fault is found for the first time during team meeting. A “meeting loss” occurs when a fault is first found during individual inspection activity, but it is subsequently

not recorded during inspection meeting. The following null hypotheses have been stated:

H_{05} : There is no difference between meeting gains and meeting losses of CBR teams;

H_{06} : There is no difference between meeting gains and meeting losses of PBR teams.

4. The Experiment

This section describes the instruments and the procedure of the experiment. The following elements are the same as used during Experiment 1, and they are described in more detail in Ref. [22]: (a) inspection techniques (CBR checklist and PBR scenarios); (b) inspection objects; (c) defect types; (d) defect registration forms.

4.1 Variables

Two types of variables are defined for the purpose of the experiment: independent variables and dependent variables.

The independent variables include reading techniques, team size and composition, duration of experiment, experience of subjects, etc. In our experiment, we let only reading techniques change, while other independent variables were kept constant. Reading techniques (CBR and PBR) were independent variables in our experiment.

We measured two types of dependent variables: dependent variables for individual subjects, and dependent variables for 3-person inspection teams, which are described in Table 1 and Table 2 respectively.

Dependent variables for individual subjects were variables calculated for each subject, such as number of defects found, time spent on inspection, defect detection effectiveness and cost per defect (Table 1).

Dependent variables for 3-person inspection teams were the number of defects detected by a team, the time spent on team meeting, the total time spent on inspection by team members, defect detection effectiveness, cost per defect, meeting gains and meeting losses (Table 2).

Meeting losses were calculated by comparing Individual and Team defect registration forms. Individual defect registration form showed whether or not an inspector discovered a particular defect. Meanwhile, Team defect registration form showed whether or not a team discovered a particular defect. A meeting loss occurs when a defect, reported in Individual defect registration forms of team mem-

Table 1 Dependent variables for individual subjects.

Variable	Variable description	Measurement units
D	Number of defects detected by a subject	
T	Time spent on inspection	Minutes
E	Defect detection effectiveness $E = (D/\text{Total number of seeded defects}) * 100$	Percent
C	Cost per defect $C = T/D$	Minutes

Table 2 Dependent variables for 3-person inspection teams.

Variable	Variable description	Measurement units
TD	Number of defects detected by a team	
TM	Time spent on team meeting	Minutes
TIM	Total inspection time, spent by team members for individ. inspection and team meeting	Minutes
TE	Defect detection effectiveness $TE = (TD/\text{Total number of seeded defects}) \times 100$	Percent
TC	Cost per defect $TC = TIM/TD$	Minutes
MG	Meeting gains: number of new defects first detected during team meeting	
ML	Meeting losses: number of defects first detected by individ. inspectors but not reported by a team	

bers, is not reported in Team defect registration form. Alternatively, a meeting gain occurs when a defect, not reported in Individual defect registration forms of team members, is reported in Team defect registration form.

4.2 Experimental Subjects and Their Training

Subjects were 54 third-year Bachelor students of Software Design course at Osaka University. They had previous classroom experience with the programming languages, Object-Oriented development, UML, software design activities and conventional software review. A week before the experiment we conducted a training session to improve student's understanding of the systems which would be inspected during experiment (see Sect. 4.4). The class was divided into two groups. Students of each group received Requirement's description, Use-case diagram and part of Class diagram of one software systems (either Seminar or Hospital), and were asked to create Sequence and Component diagrams of the system.

4.3 Experimental Design

The design of the experiment is shown in Table 3. Subjects were randomly assigned to one of four inspection groups. Each group used one inspection technique (either CBR or PBR) and inspected one software system (either Seminar or Hospital). Each inspector group included students with similar mix of abilities based on the results students had shown during the training session. Software systems used during experiment are described in Sect. 4.4.

After the individual inspection stage was completed (maximum time for individual inspection was 60 minutes), subjects of each group were assigned into 3-person teams and performed inspection meetings (maximum time for inspection meetings was 30 min).

Table 3 Experimental design.

Groups	Individual inspection			No of teams during insp. meeting
	Reading technique	Software system	No of subjects	
Group 1	CBR	Seminar	15	5
Group 2	CBR	Hospital	12	4
Group 3	PBR	Seminar	12	4
Group 4	PBR	Hospital	15	5

Table 4 Experimental objects.

UML diagram type	Number of pages		Check-list	Scenarios		
	Seminar system	Hospital system		U	D	I
Use-Case	1	1	x	x	x	x
Class	1	1	x	x	x	x
Activity	3	1	x	x		
Sequence	9	10	x	x	x	x
Component	1	1	x			x

4.4 Experimental Objects, and Seeded Defects

The UML diagrams [3] (paper-documents) of two software systems (Seminar system and Hospital system), used in Experiment 1 [22], were used in this experiment. They have been borrowed from Ref. [10]. Seminar system was dealing with the activities such as arrangement of seminar schedules, seminar hall reservation, lecturer designation, audience subscription, report reception and grading, etc. Hospital system included activities such as oral consultation, medical examination, treatment of the patients, prescription of the medicines, etc.

We had less time for individual inspection in Experiment 2 (60 min) as compared to Experiment 1 (120 min). Therefore, the number of inspection objects used in Experiment 2 was smaller: the size of Seminar system documentation was 16 pages (24 in Experiment 1), and the size of Hospital system documentation was 15 pages (18 in Experiment 1). The number of UML diagrams and their assignment to inspection techniques is shown in Table 4 (in Table 4, "U" corresponds to User's scenario, "D" corresponds to Designer's scenario and "I" corresponds to Implementer's scenario).

Although we used the same types of defects as in Experiment 1 [22] (syntactic, semantic and consistency), we modified some of defects. After creation of new defects, we asked several colleagues to try to find those defects using checklist and scenarios. As a result, we received some important comments and considering them, we created the final list of defects. UML diagrams were created and defects were inserted using Rational Rose 2001 Professional Java Edition software. The number of defects inserted in each software system is given in the Table 5.

The way to create defects could have some impact to the external validity, because our colleagues and we are not practitioners working with UML diagrams. However, we think that our knowledge was sufficient for this task. Defects were randomly distributed in UML diagrams, for this reason

Table 5 Seeded defects.

UML diagram type	Number of defects	
	Seminar system	Hospital system
Class	3	4
Activity	4	4
Sequence	3	3
Component	3	3
<i>Total:</i>	13	14

it might have a minor influence over the probability of a CBR/PBR inspector to find defects. Syntactic defects were the easiest to detect comparing to semantic and consistency defects. Therefore, the probability of inspectors to detect defects varied with defect classes. To minimize its impact to the setting of the research hypotheses, we inserted similar mix of defect classes into each type of UML diagrams.

4.5 Experiment Operation

The language of experiment was Japanese. The following timetable was used to arrange the experiment:

1. Explanations of the experiment activities and conduction of the inspection experiment. Two rooms were used, one for each inspection technique – PBR and CBR. Students were divided into four groups (see Table 3). Before the experiment students listened to the explanations, which lasted approximately 20 minutes.
2. After the explanations were given, individual inspection stage was conducted, which lasted 60 minutes. Students were inspecting the same software system they had analyzed during the training session.
3. After individual inspection stage has been completed, subjects were assigned into 3-person teams and performed 30-minute long inspection meetings. Each team was given a document with guidelines on how to perform inspection meeting. Teams had to register actual defects into Team defect registration form. The structure of this form was the same as of the form used during individual inspection.

5. Threats to Validity

There are four groups of threats to the validity of the experiment results: internal validity, external validity, conclusion validity and construct validity [28].

Threats to internal validity are those that can affect the independent variable with respect to causality without the researcher's knowledge. There might have been some threat to selection, because experiment was a mandatory part of the course. To minimize it, we have randomly assigned the subjects into groups which used only one of the reading techniques. Similar training was given to subjects to ensure they all start at the same level. The objects (UML diagrams), which we used, could also have influence to the internal validity – threat of instrumentation. We made sure for both software systems to be similar in size and complexity. To check the process conformance of the inspectors,

we used the defect registration forms, where student were asked to write down which item of the Checklist or Scenario he used to detect each defect. After experiment, we checked if those items were listed in correct sequence, as it was defined in Checklist or Scenario. In addition, we verified if it was possible to detect defects using corresponding items of the Checklist or Scenario. Students who used PBR had to perform various tasks as well. After the experiment, we have checked if those tasks have been performed. The data of the students who did not conform to the process has been eliminated from the further analysis. The transition between individual and team activities was as short as possible to minimize any interaction among individuals.

External validity concerns the ability to generalize the experiment results to industry practice. The biggest threat to the external validity is that students, instead of practitioners, were used as subjects. However, students were in the end of their third year of studies in software engineering, close to their start working in the industry. There are more experiments [9], [15], [24], [25] reported in the literature, where students were successfully used as subjects. The design documents were similar to those, which are used in practice, but the size of systems in industry is usually larger. However we think, that the amount of documents which subject were required to inspect was appropriate.

Threats to conclusion validity concern the issues that affect the ability to draw the correct conclusion about the relationship between dependent and independent variables. Threats regarding the random heterogeneity of subjects are limited, since students had similar knowledge and background.

Construct validity concerns the ability to generalize from the experiment results to the concept or theory behind the experiment. The subjects did not know what hypotheses were stated, and what the expected result of the experiment was. Consequently, those threats to validity are considered small.

Thus, it can be concluded that there were threats to internal and external validity, but they were not considered large in this experiment.

6. Data Analysis

This section describes the data collected during experiment and the statistical tests, which were used during data analysis. The section consists of two subsections: individual and team data analysis.

In hypotheses testing we assume that the null hypothesis is rejected if and only if it is rejected for both inspected systems (Seminar and Hospital).

6.1 Individual Data Analysis

Two types of data were collected during the experiment, time data and defect data. Time data showed how much time each subject spent during the inspection. The time spent for explanations before individual inspection (20 min) has been

Table 6 Means and standard deviation of dependent variables collected from the individual inspection stage.

System	Inspec. technique	Defects			Inspec. time	
		Seeded	Detected		Mean	Std. dev.
			Mean	Std. dev.		
Hospital	CBR	14	6.6	1.68	77.3	3.25
	U	7	4.2	0.84	72.8	6.06
	D	7	2.6	0.55	73.0	5.05
	I	8	3.6	1.14	76.8	2.39
Seminar	CBR	13	5.6	1.99	77.1	4.04
	U	7	4.3	2.06	75.8	0.50
	D	6	3.5	1.00	75.5	1.73
	I	7	3.0	1.41	75.3	3.77

Table 7 Results of hypotheses H_{01} and H_{02} testing.

System	Dependent variable	Inspec. techn.	Mean	Std. dev.	p-value (2-tailed)
Hospital	Effectiveness (H_{01})	CBR	47.0	11.97	0.946
		PBR	47.4	14.58	
	Cost per def. (H_{02})	CBR	12.7	4.83	0.000
		PBR	23.6	8.05	
Seminar	Effectiveness (H_{01})	CBR	43.1	15.33	0.164
		PBR	53.9	22.18	
	Cost per def. (H_{02})	CBR	15.6	5.87	0.008
		PBR	24.8	10.56	

added to the inspection time. The added defect data showed the number of defects, which were detected by the subject. Data gathered during individual inspection stage is shown in Table 6 (in Table 6, “U” corresponds to User’s scenario, “D” corresponds to Designer’s scenario and “I” corresponds to Implementer’s scenario of PBR inspection technique).

We used independent samples t-test [17], [28] with significance interval 95% ($p < 0.05$) to test the hypotheses for individual inspectors H_{01} and H_{02} . The statistical results of hypotheses testing are given in Table 7.

Hypothesis H_{02} can be rejected, and hypothesis H_{01} can not be rejected. In other words, there is no statistically significant difference in individual inspector effectiveness between CBR and PBR inspection techniques. However, it is statistically significant, that cost per defect of subjects who use PBR is higher as compared to subjects who use CBR. These results confirm the results of Experiment 1.

6.2 Team Data Analysis

The data collected during inspection meetings is given in the Table 8. For each inspection team we have collected the number of detected defects, inspection meeting time, total inspection time, new defects found during inspection meeting and defects lost during inspection meeting.

We used independent samples t-test [17], [28] with significance interval 95% ($p < 0.05$) to test the hypotheses H_{03} and H_{04} for inspection teams. The statistical results of hypotheses testing are given in Table 9.

Both hypotheses H_{03} and H_{04} can not be rejected. In other words, there is no statistically significant difference in defect detection effectiveness and cost per defect between

Table 8 Means and standard deviation of dependent variables collected from inspection meeting stage.

Variable	Statistics	Value of statistics			
		Hospital syst.		Seminar syst.	
		CBR	PBR	CBR	PBR
No of defects (TD)	Mean	8.75	8.80	6.8	6.25
	Std. dev.	1.26	1.64	0.84	0.5
Meeting time (TM)	Mean	72.8	58.2	85.8	64.5
	Std. dev.	6.65	16.01	14.17	5.19
Total insp. time (TIM)	Mean	302.3	280.8	317.0	291.0
	Std. dev.	7.04	21.88	15.5	2.58
Meeting gains (MG)	Mean	0.25	0.8	0	0
	Std. dev.	0.5	1.3	0	0
Meeting losses (ML)	Mean	2.25	1.2	2.4	1.75
	Std. dev.	1.5	0.84	1.34	1.26

Table 9 Results of hypotheses H_{03} and H_{04} testing.

System	Dependent variable	Inspec. techn.	Mean	Std. dev.	p-value (2-tailed)
Hospital	Effectiveness (H_{03})	CBR	62.5	8.99	0.96
		PBR	62.9	11.74	
	Cost per def. (H_{04})	CBR	35.2	6.23	0.56
		PBR	32.7	5.99	
Seminar	Effectiveness (H_{03})	CBR	52.3	6.44	0.26
		PBR	48.1	3.85	
	Cost per def. (H_{04})	CBR	47.2	6.62	0.91
		PBR	46.8	3.59	

Table 10 Results of hypotheses H_{05} and H_{06} testing.

System	Dependent variable comparison	Inspection technique	p-value (2-tailed)
Hospital	Meeting gains – meeting losses	CBR (H_{05})	0.041
		PBR (H_{06})	0.587
Seminar	Meeting gains – meeting losses	CBR (H_{05})	0.016
		PBR (H_{06})	0.069

CBR and PBR 3-person inspection teams.

In order to test the hypotheses H_{05} and H_{06} for inspection teams, a paired samples t-test [17], [28] with significance interval 95% ($p < 0.05$) has been used. The statistical results of hypotheses testing are given in Table 10.

As we can see from Table 10, hypothesis H_{05} can be rejected, and hypothesis H_{06} can not be rejected. It means that there is a statistically significant difference between meeting gains and meeting losses of CBR teams. CBR teams lose more defects (Hospital system: 2.25, Seminar system: 2.4) than the number of new defects they detect during inspection meetings (Hospital system: 0.25, Seminar system: 0). However, there is no statistically significant difference between meeting gains and meeting losses of PBR teams. This means, that PBR teams lose similar number of defects during inspection meeting to the number of new defects they find.

7. Discussion

In this section, an interpretation of the results of hypotheses testing is given.

Hypothesis H_{01} did not show significant results, it

means that there is no difference in defect detection effectiveness of subjects who use PBR inspection technique as compared to subjects who use CBR (for subject who inspected Hospital system $p = 0.95$; for subjects who inspected Seminar system $p = 0.16$).

Hypothesis H_{02} did show significant results, which means that cost per defect of subjects who use PBR is greater than the cost per defect of subjects who use CBR (for subject who inspected Hospital system $p = 0.000$; for subjects who inspected Seminar system $p = 0.008$).

The results of hypotheses for individual inspectors H_{01} and H_{02} confirm the results of Experiment 1, which indicated that there is no statistically significant difference in defect detection effectiveness between subjects who use CBR technique as compared to the subjects who use PBR technique. In addition, subjects who use PBR technique have higher cost per defect than the ones who use CBR (Hospital system: 46% higher, Seminar system: 37% higher).

The hypotheses for 3-person inspection teams H_{03} and H_{04} did not show the significant results. It means (H_{03}) that there is no statistically significant difference in defect detection effectiveness between CBR and PBR teams (for teams that inspected Hospital system $p = 0.96$; for teams that inspected Seminar system $p = 0.26$). In addition, there is no statistically significant difference in cost per defect (H_{04}) between CBR and PBR teams (for teams that inspected Hospital system $p = 0.56$; for teams that inspected Seminar system $p = 0.91$).

The final cost per defect of PBR inspection teams is not higher than of CBR inspection teams, although PBR inspectors have higher cost per defect during individual inspection stage than CBR inspectors.

From the hypotheses H_{05} and H_{06} we get to know that only H_{05} showed significant results. This means that (H_{05}) CBR exhibited greater meeting losses than meeting gains (for CBR teams that inspected Hospital system $p = 0.041$; for CBR teams that inspected Seminar system $p = 0.016$); however (H_{06}) PBR exhibited similar meeting losses and meeting gains (for PBR teams that inspected Hospital system $p = 0.587$; for PBR teams that inspected Seminar system $p = 0.069$).

One of the reasons why hypotheses H_{01} , H_{02} , H_{03} and H_{04} did not show the significant result might be limited inspection time (the maximum time for individual inspection was 60 minutes; and the maximum time for inspection meeting was 30 minutes). Due to the time limits, some individual inspectors and inspection teams were unable to complete all inspection activities. Therefore, in order to verify the results of the experiment, a replication of the experiment should be conducted, letting individual inspectors and inspection teams to use as much time as they need.

The results of the experiment are in line with the results of several other experimental investigations [11], [18], [27], which have reported that inspection teams detect on average less than 10% of all defects during team meeting.

In Ref. [18] authors compared the performance of inspection teams which used *Ad hoc*, CBR and Scenario-based

reading techniques. The results of comparison did not reveal any difference among these techniques with respect to meeting gains and meeting losses: meeting gains were offset by meeting losses for the teams that used each technique. However, the results of our experiment shows a difference between CBR and PBR techniques with respect to these variables: for PBR teams, meeting gains are similar to meeting losses; for CBR teams, meeting gains are smaller than meeting losses. In other words, PBR technique outperforms CBR technique with respect to the difference between meeting gains and meeting losses. Consequently, CBR 3-person team meetings are less beneficial than PBR 3-person team meetings.

In addition to tangible benefits of inspection meetings, there exist intangible benefits such as dissemination of product information, development experiences, or enhancement of team spirit [13]. Therefore, if practitioners would decide to perform inspection which includes both individual and team meeting activities, we recommend using PBR inspection technique, since it is similar to CBR with respect to cost and effectiveness; however PBR team meetings are more beneficial than CBR team meetings.

8. Conclusion

The experiment presented in this paper focuses on comparison of two reading techniques, CBR and PBR for UML design document inspection. In addition, it investigates the usefulness of inspection meetings.

The results of the individual inspection stage of experiment confirmed the results of the Experiment 1, i.e. cost per defect of PBR subjects was higher as compared to CBR subjects, and the effectiveness of CBR and PBR subjects was similar. Moreover, there was no statistically significant difference between 3-person CBR and PBR teams with respect to defect detection effectiveness and cost per defect. In order to evaluate the usefulness of inspection meetings, we measured if there is a difference between meeting gains and meeting losses of CBR and PBR teams separately. The results showed, that CBR teams exhibit greater meeting losses than meeting gains; meanwhile PBR teams exhibit similar meeting losses and meeting gains. Thus, CBR 3-person team meetings are less beneficial than PBR 3-person team meetings.

The results of this work have important hints for software practitioners. The indications are that inspection meeting performance can be improved when individual inspectors use PBR inspection technique.

Further studies are needed to investigate these issues in an industrial setting.

References

- [1] V.R. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Sorumgard, and M.V. Zelkowitz, "The empirical investigation of perspective-based reading," *Empir. Softw. Eng.*, vol. 1, no. 2, pp. 133–164, 1996.
- [2] S. Biffl and M. Halling, "Investigating the influence of inspector ca-

- pability factors with four inspection techniques on inspection performance," Proc. 8th IEEE Symposium on Software Metrics, pp.107-117, Ottawa, Canada, June 2002.
- [3] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language user guide*, Addison Wesley Longman, 1999.
 - [4] B. Cheng and R. Jeffery, "Comparing inspection strategies for software requirements specifications," Proc. Australian Software Engineering Conference, pp.203-211, Melbourne, Australia, July 1996.
 - [5] Y. Chernak, "A statistical approach to the inspection checklist formal synthesis and improvement," IEEE Trans. Softw. Eng., vol.22, no.12, pp.866-874, 1996.
 - [6] A. Dunsmore, M. Roper, and M. Wood, "Further investigations into the development and evaluation of reading techniques for object-oriented code inspection," Proc. 24th International Conference on Software Engineering, pp.47-57, Florida, USA, May 2002.
 - [7] M. Fagan, "Design and code inspections to reduce errors in program development," IBM Syst. J., vol.15, no.3, pp.182-211, 1976.
 - [8] T. Gilb and D. Graham, *Software inspection*, Addison-Wesley, 1993.
 - [9] M. Host, B. Regnell, and C. Wohlin, "Using students as subjects — A comparative study of students and professionals in lead-time impact assessment," Empir. Softw. Eng., vol.5, no.3, pp.201-214, 2000.
 - [10] K. Itoh, T. Hirota, T. Fuji, S. Kumagai, and R. Kawabata, *Software engineering exercises*, Ohmsha, 2001.
 - [11] P.M. Johnson and D. Tjahjono, "Does every inspection really need a meeting?," Empir. Softw. Eng., vol.3, no.1, pp.9-35, 1998.
 - [12] O. Laitenberger, C. Atkinson, M. Schlich, and K. El Emam, "An experimental comparison of reading techniques for defect detection in UML design documents," J. Syst. Softw., vol.53, no.2, pp.183-204, 2000.
 - [13] O. Laitenberger and J.M. DeBaud, "An encompassing life cycle centric survey of software inspection," J. Syst. Softw., vol.50, no.1, pp.5-31, 2000.
 - [14] O. Laitenberger and C. Atkinson, "Generalizing perspective-based inspection to handle object-oriented development artifacts," Proc. 21st International Conference on Software Engineering, pp.494-503, Los Angeles, CA, USA, May 1999.
 - [15] L.P.W. Land, R. Jeffery, and C. Sauer, "Validating the defect detection performance advantage of group designs for software reviews: Report of replicated experiment," Proc. Australian Software Engineering Conference, pp.17-26, Sydney, Australia, Oct. 1997.
 - [16] O. Mizuno and T. Kikuno, "Empirical evaluation of review process improvement activities with respect to post-release failure," Proc. Empirical Studies on Software Development Engineering Workshop, pp.50-53, Los Angeles, CA, USA, May 1999.
 - [17] M.J. Norusis, *SPSS: SPSS 6.1 guide to data analysis*, Prentice Hall, Englewood, NJ, 1995.
 - [18] A.A. Porter, L.G. Votta, and V. Basili, "Comparing detection methods for software requirements inspections: A replicated experiment," IEEE Trans. Softw. Eng., vol.21, no.6, pp.563-575, 1995.
 - [19] A.A. Porter and P.M. Johnson, "Assessing software review meetings: Results of a comparative analysis of two experimental studies," IEEE Trans. Softw. Eng., vol.23, no.3, pp.129-145, March 1997.
 - [20] B. Regnell, P. Runeson, and T. Thelin, "Are the perspectives really different? Further experimentation on scenario-based reading of requirements," Empir. Softw. Eng., vol.5, no.4, pp.331-356, 2000.
 - [21] G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, and K. Inoue, "An experimental comparison of checklist-based reading and perspective-based reading for UML design document inspection," Proc. International Symposium on Empirical Software Engineering, pp.148-157, Nara, Japan, Oct. 2002.
 - [22] G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, and K. Inoue, "Further investigations of reading techniques for object-oriented design inspection," Inf. Softw. Technol., vol.45, no.9, pp.571-585, 2003.
 - [23] F. Shull, I. Rus, and V. Basili, "How perspective-based reading can improve requirements inspections," Computer, vol.33, no.7, pp.73-79, 2000.
 - [24] W. Tichy, "Hints for reviewing empirical work in software engineering," Empir. Softw. Eng., vol.5, no.4, pp.309-312, 2000.
 - [25] G. Travassos, F. Shull, M. Fredericks, and V. Basili, "Detecting defects in object oriented designs: Using reading techniques to increase software quality," Proc. ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages & Applications, pp.47-56, Denver, CO, USA, Nov. 1999.
 - [26] S. Uchida, A. Monden, H. Iida, K. Matsumoto, and H. Kudoh, "Analysis of program reading process in software debugging based on multiple-view analysis model," Proc. 6th Joint Workshop on System Development, CD-ROM Edition, Nagoya, Japan, April 2003.
 - [27] L.G. Votta, Jr., "Does every inspection need a meeting?," Proc. ACM SIGSOFT Symposium on Foundations of Software Engineering, ACM Software Engineering Notes, vol.18, no.5, pp.107-114, Dec. 1993.
 - [28] C. Wohlin, P. Runeson, M. Host, M.C. Ohlsson, B. Regnell, and A. Wesslen, *Experimentation in software engineering: An introduction*, Kluwer Academic Publishers, 2000.



Giedre Sabaliauskaite received the B.Sc. and M.Sc. degrees in informatics from Kaunas University of Technology, Lithuania in 1996 and 1998, respectively. She is currently a PhD candidate in the Software Engineering Laboratory at Osaka University. Her research interests include software inspection and software metrics.



Shinji Kusumoto received the B.E., M.E. and D.E. degrees in information and computer sciences from Osaka University in 1988, 1990 and 1993, respectively. He is currently an associate professor at Osaka University. His research interests are software metrics, software quality assurance technique. He is a member of the IEEE, IPSJ and JFPUG.



Katsuro Inoue received the B.E., M.E. and D.E. degrees in information and computer sciences from Osaka University in 1979, 1981 and 1984, respectively. He was an associate professor at the University of Hawaii at Manoa from 1984 to 1986. After becoming an assistant professor at Osaka University in 1989 and an associate professor in 1991, he has been a professor there since 1995. He is engaged in the study of software engineering. He is a member of the IEEE, ACM, IPSJ and JSSST.