Investigating Defect Detection in Object-Oriented Design and Cost-Effectiveness of Software Inspection

By

Giedre SABALIAUSKAITE

January 2004

Dissertation submitted in partial fulfilment of the requirements for the degree of Doctor of Engineering at the Faculty of Engineering Science of Osaka University

ABSTRACT

As the use of software grows in today's society, software quality is becoming an increasingly important issue, and the need for activities to control and improve it is increasing dramatically. Software inspection has been extensively used for over thirty years to ensure the quality of software, by finding and repairing defects in software products. However, the yet narrow scope of research has been centred on inspection of Object-Oriented artifacts. Moreover, few methods have been developed for evaluating inspections and deciding whether they are cost-effective, as compared to other quality assurance techniques, such as testing.

A typical inspection process consists of two stages critical for defect detection: individual inspection (preparation) followed by inspection meeting. Recently, the controversy of inspection meeting effectiveness has been raised in the literature, because the cost of organizing and conducting them is high, while there is no net meeting gain. Thus, further research is needed in order to allow evaluation of effectiveness and efficiency of these inspection stages.

This thesis addresses the issues of effective defect detection in Object-Oriented design, usefulness of inspection meetings and evaluation of cost-effectiveness of inspection. As a result, two inspection strategies (reading techniques), namely *Checklist-Based Reading* and *Perspective-Based Reading*, are developed and experimentally evaluated. Furthermore, four new metrics to allow more precise evaluation of inspection as compared to the conventional methods are proposed, and their usefulness is demonstrated using the data collected from an experimental investigation. Two of these metrics, namely *Preparation Losses* M_{l_IDV} and *Inspection Meeting Losses* M_{l_MEET}, are intended for evaluation of the cost wasted during preparation and inspection meeting stages of an inspection process due to false positives (erroneously identified defects). Another two, namely *Extended Cost Effectiveness of Preparation* M_{g_IDV} and *Extended Cost Effectiveness of Preparation and Inspection Meeting* M_{g_MEET}, are aimed at evaluating cost-effectiveness of preparation and inspection meeting stages.

The overall results indicate that the inspection techniques and metrics proposed in this thesis may facilitate the work of researchers and practitioners when utilizing and evaluating software inspection.

In this thesis, Chapter 1 briefly describes the overview and contribution of the thesis.

Chapter 2 presents a review of relevant literature, describing the main principles behind inspection, different reading techniques, and current methods to evaluate inspections.

Chapter 3 describes an experimental evaluation of two reading techniques, namely Checklist-based reading and Perspective-based reading. The goal of the experiment was the development and application of two different reading techniques in object-oriented design inspection, in order to compare the performance of individual inspectors using these techniques.

Chapter 4 presents the findings of a second controlled experiment that was conducted to investigate the performance of individual inspectors as well as 3-person inspection teams. The purpose of the experiment was twofold: to verify the results of the first experiment, and to investigate the effectiveness of inspection meetings.

Chapter 5 describes the results of a further investigation into inspection meetings using the data collected from the second experiment.

Chapter 6 investigates inspection-related costs and the methods to evaluate cost-effectiveness of inspections. As a result, four new metrics are proposed.

Chapter 7 presents an experimental evaluation of new metrics using the data collected from the second experiment.

Chapter 8 summarizes the main results of this thesis. It includes the main findings and provides directions for future research.

LIST OF MAJOR PUBLICATIONS

1. G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, K. Inoue, An Experimental Comparison of Checklist-Based Reading and Perspective-Based Reading for UML Design Document Inspection, Proceedings of the 2002 International Symposium on Empirical Software Engineering (ISESE2002) 148-157.

2. F. Matsukawa, G. Sabaliauskaite, S. Kusumoto, K. Inoue, Experimental Comparison of Checklist-Based Reading and Perspective-Based Reading for UML Design Documents, Proceedings of the 2002 Object Oriented Symposium (OO2002) 67-74 (In Japanese).

3. G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, K. Inoue, Further Investigations of Reading Techniques for Object-Oriented Design Inspection, Information and Software Technology 45 (9) (2003) 571-585.

4. G. Sabaliauskaite, S. Kusumoto, K. Inoue, Extended Metrics to Evaluate Cost Effectiveness of Software Inspections, IEICE Transactions on Information and Systems, E87-D (2) (2004) 475-480.

5. G. Sabaliauskaite, S. Kusumoto, K. Inoue, Comparing Reading Techniques for Object-Oriented Desing Inspection, IEICE Transactions on Information and Systems, (2004) (To appear).

6. G. Sabaliauskaite, S. Kusumoto, K. Inoue, Assessing Defect Detection Performance of Interacting Teams in Object-Oriented Design Inspection, Information and Software Technology (2004) (Conditional acceptance).

ACKNOWLEDGEMENTS

I am most indebted to my supervisor Professor Katsuro Inoue for his continuous support and supervision over the years. Without his help, experience and advice, this thesis would never have reached completion.

I am very grateful to Professor Tohru Kikuno and Professor Toshimitsu Masuzawa for their valuable comments and helpful criticism on this thesis.

I would like to express my gratitude to Associate Professor Shinji Kusumoto and Assistant Professor Makoto Matsushita. Their comments, criticism and advice have helped guide and shape the development of this thesis.

I would also like to express my gratitude to all members of the Department of Informatics and Mathematical Science for their guidance, especially Professors Toru Fujiwara, Kenichi Hagihara, Teruo Higashino, Masaharu Imai, Makoto Imase, Toshinobu Kashiwabara, Hideo Matsuda, Hideo Miyahara, Masayuki Murata and Kenichi Taniguchi.

Thanks are also due to many friends in the Department of Informatics and Mathematical Science, especially students in Inoue Laboratory.

I am extremely grateful to Ms. Yoshimi Katagiri for all the help during my stay in Japan. I also give my gratitude to my parents and friends, who have been a constant source of encouragement. Finally, I would like to say special thanks to Sergio and Gillean for their patience, their support, and for pushing me over the finishing line.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

1.1 Overview

Software quality is becoming increasingly important issue in today's society as the use of software grows. Inspection [Fagan 1976 and 1986; Ackerman et al. 1989] and testing [Hetzel 1998] are two widely recommended techniques for software quality improvement. While both of them are used for defect detection and removal in software products, testing cannot be conducted until software is implemented, whilst inspections can be applied in early stages of software development process and help to avoid costly rework.

"Inspections are now thirty years old and they continue to improve software quality and maintainability, reduce time to delivery, and lower development costs."

M. Fagan [Broy & Denert 2002, p.215]

Inspections may be used to find defects and remove them from all work products created during software development process: requirements specifications, design, code, test plans and test cases, and user documentation [Broy & Denert 2002]. It has been claimed that inspections can lead to detection and correction of 50% to 90% of total defects in a software document [Fagan 1986; Gilb & Graham 1993]. Although inspections have now been used for more than 30 years, software is still released with defects. Therefore, further research is needed to find more practical and effective inspection methods [Parnas & Lawford 2003].

Till now, inspections have been primarily used for defect detection in documents belonging to conventional structured development process, such as functional requirement documents, design and code. Yet, there exists a significant lack of information about how inspections should be applied to Object-Oriented (OO) artifacts, such as OO code and design [Dunsmore et al. 2001; Dunsmore 2002; Laitenberger et al. 2000; Sabaliauskaite et al. 2003].

Inspections typically comprise three to six participants, and consist of several stages [Fagan 1976; IEEE 1989]: (1) *planning*, where organizer prepares the materials, selects participants and meeting place for inspection; (2) *overview*, where participants are presented with a general overview of the scope and

purpose of inspection; (3) *preparation*, where participants analyze the work product individually with the goal of understanding it thoroughly; (4) *inspection meeting*, where participants gather in a team to discuss the inspection product and to detect as many defects as possible; (5) *rework*, where author revises the defects found and corrects the inspection product; (6) *follow-up*, where moderator verifies the quality of rework and decides whether a reinspection is required.

Recently, the focus of detecting defects has moved from inspection meeting stage to preparation stage of inspection. Therefore, there is a need for an adequate support for inspectors during preparation stage via reading techniques (a set of guidelines used by inspector during preparation). Furthermore, 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 [Johnson & Tjahjono 1998; Porter & Johnson 1997; Votta 1993].

In addition to true defects, false positives (erroneously identified defects that require no repair) are being reported during inspections. They do not add to the quality of software, because the rework of false positives is costly and may introduce new defects. Nevertheless, there is a lack of research in this area.

Another important issue is cost-effectiveness (the extent to which savings achieved are worth the costs) of software inspections. It has been claimed that software design inspections save on average 44% of the testing costs, while code inspections save on average 39% of those costs [Briand et al. 1998]. Since most companies spend between 50% to 80% of their development effort on testing [Collofello & Woodfield 1989], reducing testing cost is an important step towards increasing productivity and quality of software development, in which inspection plays a major role. However, are software inspections always worthwhile? What is the influence of false positives on inspection cost-effectiveness?

In order to evaluate the cost-effectiveness of inspections, several metrics have been previously proposed. Collofello & Woodfield have taken into account all the costs consumed and saved by inspections and proposed a metric, called *Cost Effectiveness* [Collofello & Woodfield 1989]. Kusumoto et al. proposed a metric M_k for evaluating cost-effectiveness of inspections, which is based on the degree to which costs to detect all faults from the software in a project during testing are reduced by inspections [Kusumoto et al. 1992]. Yet, none of those metrics includes the information about false positives introduced during inspection.

This thesis develops and experimentally evaluates two reading techniques, namely *Checklist-Based Reading* and *Perspective-Based Reading* for OO design inspection. Besides, it investigates the usefulness of inspection meetings, and

the impact of false positives on inspection effectiveness as well as on software development costs. Finally, thesis proposes four new metrics for more precise evaluation of inspection as compared to the conventional metrics. Two of these metrics, namely *Preparation Losses* M_{l_IDV} and *Inspection Meeting Losses* M_{l_MEET}, are intended for evaluation of the cost wasted during preparation and inspection meeting stages due to false positives. Another two, namely *Extended Cost Effectiveness of Preparation* M_{g_IDV} and *Extended Cost Effectiveness of Preparation and Inspection Meeting* M_{g_MEET}, are the modifications of metric M_k, which include false positives and therefore are more precise than M_k. The validity and usefulness of proposed metrics is demonstrated using the data collected from a controlled experiment.

1.2 Contribution of Thesis

The work presented in this thesis makes the following contributions to the area of software inspection:

- The development of two reading techniques for inspection of object-oriented design – a checklist for inspection using *Checklist-Based Reading* technique, and three scenarios for inspection using *Perspective-Based Reading* technique – that can be used for inspection of design diagrams written using the notation of *Unified Modelling Language*.
- Two controlled experiments to investigate the effectiveness of two reading techniques developed specifically for object-oriented design.
- An extensive investigation of inspection meetings, focusing on the usefulness of inspection meeting stage as compared to preparation stage of the inspection process. It contributes to the further exploration of the impact of false positives on inspection effectiveness, and to the validation of a recently proposed *Behaviourally Motivated Inspection Theory* [Sauer et al. 2000].
- The investigation of the impact of false positives on the cost-effectiveness of inspection, and development of two cost models to graphically demonstrate this impact on preparation and inspection meeting stages.
- The development and experimental evaluation of four metrics to assess software inspections:
  - Two metrics to evaluate the cost wasted during preparation and

inspection meeting stages of inspection process due to false positives – M_{I_IDV} and M_{I_MEET}. Metric M_{I_IDV} for evaluation of *Preparation Losses*, and M_{I_MEET} for evaluation of *Inspection Meeting Losses*;

- Two metrics M_{g_IDV} and M_{g_MEET}, which are the modifications of Kusumoto's metric M_k, to evaluate the *Extended Cost Effectiveness of Preparation* and the *Extended Cost Effectiveness of Preparation and Inspection Meeting* respectively.

1.3 Thesis Outline

The remaining of the thesis is structured in the following way:

Chapter 2: Preliminaries

The thesis begins with a review of relevant literature, describing the main principles behind inspection, different reading techniques, methods to evaluate inspections and trends in the area of software inspection.

Chapter 3: Evaluation of Two Reading Techniques for Object-Oriented Design Inspection (Experiment 1)

This chapter describes an experimental evaluation (Experiment 1) of two reading techniques, namely Checklist-based reading and Perspective-based reading for object-oriented design document, written using the notation of Unified Modelling Language, inspection. The goal of the experiment was the development and application of two different reading techniques in object-oriented design inspection, in order to compare the performance of individual inspectors using these techniques.

Chapter 4: Investigating Individual and 3-Person Team Performance (Experiment 2)

This chapter presents the findings of a second controlled experiment (Experiment 2) conducted to investigate the performance of individual inspectors as well as 3-person interacting teams in object-oriented design inspection. The purpose of the experiment was twofold: to verify the results of Experiment 1, and to investigate the usefulness of inspection meetings.

Chapter 5: Assessing Inspection Meetings

This chapter describes the results of a further investigation into inspection meetings using the data collected from Experiment 2. It includes the investigation of the impact of false positives on inspection effectiveness, and

validation of a part of Behaviourally Motivated Inspection Theory [Sauer et al. 2000] that defines the behaviour of inspector teams.

Chapter 6: Extended Metrics to Evaluate Cost-Effectiveness of Software Inspections

This chapter investigates inspection-related costs and methods to evaluate cost-effectiveness of inspections. As the result it proposes two extended cost models: a model to describe the costs spent during preparation stage, and a model to describe the costs spent during preparation and inspection meeting stages. In addition it proposed four new metrics. Two metrics, M_{l_IDV} and M_{l_MEET}, are intended for evaluation of *Preparation Losses* and *Inspection Meeting Losses* respectively. Another two, M_{g_IDV} and M_{g_MEET}, are developed for evaluation of *Extended Cost Effectiveness of Preparation* and *Extended Cost Effectiveness of Preparation and Inspection Meeting* respectively. All those metrics enable more precise evaluation of software inspections as compared to the conventional metrics.

Chapter 7: Experimental Evaluation of the New Metrics

This chapter presents an experimental evaluation of the new metrics proposed in Chapter 6. The metrics are applied to the data collected from Experiment 2, and the resultant values obtained by different metrics are compared.

Chapter 8: Conclusions

The final chapter concludes the thesis by summarizing the main findings, and providing directions for future research.

CHAPTER 2

Preliminaries

Software inspections have been extensively investigated since their original description by Fagan in 1976. As a result, different modifications of the traditional inspection process have been proposed. Consequently, a number of defect detection strategies (reading techniques), which guide inspectors during inspection and help them to detect defects, have been developed.

This chapter provides a brief introduction to software inspection. It describes the inspection process, and shows how the focus of detecting defects has been moved from the inspection meeting to the preparation stage of an inspection process. Moreover, it describes the Behaviourally Motivated Inspection Theory [Sauer et al. 2000], and provides an overview of the various reading techniques currently available. Finally, it introduces to the evaluation of cost-effectiveness of software inspection by describing inspection costs and evaluation methods.

2.1 Software Inspection

2.1.1 Inspection Process

Software inspection, as a structured process, was firstly described by Fagan in 1976. It was designed to control and improve the software development process, resulting in improved productivity and product quality [Fagan 1976]. A typical software inspection consists of six stages [Fagan 1976 and 1986]:

1) Planning. The organizer ensures that the materials to be inspected meet inspection criteria, arranges the availability of the right participants, meeting place and time;

2) Overview. The author presents an overview of the scope and purpose of the work product to the inspection team;

3) Preparation (often called *Individual reading*). Inspectors analyze the work product individually with the goal of understanding it thoroughly;

4) Inspection meeting. The objective of the inspection meeting stage is to find defects. Therefore, the inspection team assembles to discuss the inspection product and to detect as many defects as possible;

5) Rework. The objective of the rework is for the author to perform rework and to correct the identified defects;

6) Follow-up. The moderator verifies whether all defects have been corrected and no secondary defects have been introduced during rework, and decides whether reinspection is required.

The next section explores and discusses the effects of the preparation and inspection meeting stages on the overall effectiveness of inspection process.

2.1.2 Investigating Preparation and Inspection Meetings

In Fagan's original inspection process [Fagan 1976], the preparation stage is used by inspectors to obtain a deep understanding of the inspection artifact, and the inspection meeting stage is used by the inspectors as a group to carry out defect detection. Although defects can be detected during the preparation as well, often it is assumed that meeting allows inspectors to detect more defects. The main reasons to use inspection meeting have been summarized by Votta [Votta 1993]:

- *Synergy* – the interaction of inspectors in the group creates a team that finds many faults during the meeting that were not discovered during individual reading;
- *Education* – less experienced inspectors learn from the experienced ones how the inspection process works, etc.;
- *Schedule deadline* – the inspection creates a planned event for people to work towards;
- *Competition* – the group setting provides an arena where people publicly contribute and earn the esteem of their peers, and hence strive to improve themselves;
- *Requirement* – the document inspection process specifies that a meeting be held to collect reviewer comments.

However, a series of empirical studies into the usefulness of inspection meetings question whether meetings are really necessary [Johnson & Tjahjono 1998; Porter et al. 1995; Votta 1993]. Votta (1993) suggests that inspection meetings are no longer required since the number of new defects detected at the meeting over those found in individual reading is relatively small (4% in average).

In addition, inspection meeting increases the cost and logistic complexity of software inspection due to the time delay in organizing and holding the meetings. 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 [Porter et al. 1995]. On the other hand, it was found that inspection meetings help reduce the number of false positives [Land et al. 1997b].

The following subsections describe the research done in the area of investigating the influence of false positives on inspection effectiveness, and a recent theory proposed by Sauer et al., which defines the behaviour of inspection groups during inspection meeting [Sauer et al. 2000].

Investigating False Positives

In addition to true defects, false positives are being detected during inspection [Cheng & Jeffery 1996; Johnson & Tjahjono 1998; Land 2000]. Although false positives do not improve software quality and their rework can introduce more defects [Sauer et al. 2000], the majority of researchers do not consider them to be of great importance [Votta 1993]. In some research, false positives are being either deliberately discarded from analysis [Fusaro et al. 1997] or not mentioned at all [McCarthy et al. 1996].

Varying levels of false positives have been detected in different studies: Porter et al. (1997) reported a large proportion of false positives (about 50%) of all the issues found, while Votta (1993) reported low level of false positives (about 1%). Land et al. reported that inspection meetings are especially effective in discriminating between true defects and false positives [Land et al. 1997b; Land 2000].

This thesis further investigates the role of false positives in software inspection.

Behaviourally Motivated Inspection Theory

Sauer et al. (2000) have published a paper devoted to the construction of theory of group software inspections. In this paper, authors have applied the research from behavioural theory of group performance to software inspection meeting, and presented eleven research propositions derived from the theory.

For the purpose of the theory, two types of inspector groups are defined:

(1) *An interacting group* – a group of inspectors who interact face-to-face during

inspection meeting [Land 2000];

(2) *A nominal group* – an artificial group that consists of the same members as an interacting team, however in this case, inspectors do not interact with one another. The performance of a nominal group is generated from individual inspection performance [Biffl & Halling 2003; Land 2000].

The following are the propositions of the theory:

P1. Task expertise is the dominant determinant of group performance. If researchers manipulate the expertise variable, it is reasonable to expect that it will prove the dominant determinant of performance.

P2. Decision schemes (plurality effects) influence interacting group performance. A majority having discovered any issue is sufficient for the meeting to accept it as a true defect.

P3. In the absence of plurality, interacting group performance is a positive function of process skills. If there is no plurality to decide whether an issue is a true defect, the group's ability to make a correct discrimination is positively influenced by the quality of its group process.

P4. The interacting group meeting does not improve group performance over nominal group by discovering new defects. Group meetings do not discover significant number of new defects beyond the aggregation of those discovered by individuals (nominal group).

P5. Group performance is a positive function of task training. Providing group members with training gives significant benefits.

P6. The performance/size relationship is a function of task expertise. Group performance improves with increases in task expertise, and hence, it improves as the group size increases.

P7. Above the critical limit, performance declines with group size. There is a limit beyond which effectiveness will decline as a result of increased process loss outweighing the gains from increased expertise.

P8: The performance advantage of an interacting group over a nominal group is a function of the level of false positives discovered by individuals. The source of performance advantage of interacting groups is their ability to discriminate

between true defects and false positives; consequently interacting groups report fewer false positives than nominal groups.

P9. An expert pair performs the discrimination task as well as any larger group. Expert pairs can be expected to perform as well as larger sized groups in judgement task.

P10. Nominal groups outperform alternatives at the discovery task. While an inspection meeting may improve defect discrimination, it may cause some loss of performance by overlooking defects discovered by individual inspectors.

Some of those propositions have never been tested, while others have shown mixed evidence [Jeffery & Scott 2002]. This research verifies a part of the theory, and tests three propositions – P4, P8 and P10 (see Chapter 5).

2.1.3 Reading Techniques

The reading technique is a defect detection strategy that guides inspectors during the preparation stage of an inspection process. Several reading techniques proposed in the literature are discussed below.

Ad Hoc

This is the simplest reading technique that provides no support for inspectors, i.e. no guidelines or direction [Porter & Votta 1994]. Inspectors rely on their own intuition and experience to determine how to go about finding the defects in the document.

A strength of the *Ad hoc* technique is that more experienced inspectors have the freedom to use their knowledge in finding defects, while its main weakness is that with no support, less experienced inspectors may find it difficult to detect defects effectively.

Checklist-Based Reading

Checklist-Based Reading (CBR) is a broadly used technique in inspections since 1970's. It provides the inspectors with a list of issues which help them to know what kind of defects to look for in the software products. The items of the checklist can be either expressed as statements or as "yes/no" questions [Chernak 1996]. The inspector is requested to answer those questions while

reading software document.

Laitenberger & DeBaud (2000) identified several weak points of CBR:

a) The questions are often general and not sufficiently tailored to a particular development environment;

b) Concrete instructions on how to use the checklist are often missing, i.e. it can be unclear when and based on what information an inspector is to answer a particular checklist question;

c) The questions of the checklist are often limited to detection of defects that belong to particular defect types (inspectors may not focus on defect types not previously detected and, therefore, may miss whole classes of defects).

Scenario-Based Reading

The Scenario-based reading [Porter et al. 1995] was created to address the lack of effectiveness in using *Ad hoc* and CBR methods. Since *Ad hoc* and CBR are non-systematic techniques, they do not offer a set of concrete reading instructions, therefore inspector's experience has a significant impact on the number of defects found [Laitenberger & DeBaud 2000].

Scenario-based reading provides the scenario that gives guidance to inspectors on how to proceed and what to look for during inspection. Hence, several scenarios are developed with specific focus on particular viewpoint. Each inspector executes a single scenario, and a combination of scenarios should provide a broad coverage of a document [Porter & Votta 1994].

Several variants of scenario-based reading have been proposed: perspective-based reading, defect-based reading and traceability-based reading. Perspective-based reading has continued to be refined, and has been applied for inspection of requirement, design and code documents.

Perspective-Based Reading Technique

The main idea of the Perspective-Based Reading (PBR) technique is that a software product should be inspected from the perspective of different stakeholders [Basili et al. 1996a and 1996b; Laitenberger 2000; Regnell et al. 2000; Sabaliauskaite et al. 2002 and 2003; Thelin 2002]. The perspectives depend on the roles that people have within the software development and the maintenance process. This approach assumes that:

a) Inspector with specific focus performs better than an inspector who is trying to detect all the possible defects in a software product;

b) The union of perspective provides an extensive coverage of the document.

For examining a document from a particular perspective, PBR technique provides guidance for the inspector in the form of a PBR scenario. Such a scenario consists of three major sections [Laitenberger & Atkinson 1999]:

1. *Introduction* – describes the most relevant quality requirements to a particular perspective;
2. *Instructions* – describe what kind of documents to use, how to read, and how to extract the necessary information. The main objective of instructions is to gain a better defect detection coverage of a software artifact;
3. *Questions* – describe a set of questions which inspector has to answer during the inspection.

The benefits of PBR have been summarized by Shull et al. (2000):

- *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 as a definite procedure that does not depend on the inspector's experience, new inspectors can receive training in the procedure's steps while applying the technique.

Defect-Based Reading Technique

The main idea behind Defect-based reading is for different inspectors to focus on different defect classes [Porter et al. 1995]. For each defect class a separate scenario is developed. Porter et al. (1995) have developed scenarios for the following types of defects in the requirements documents: data type

inconsistencies, incorrect functionality, ambiguity or missing functionality.

Traceability-Based Reading Technique

The Traceability-based reading technique is aimed for inspection of OO design documents [Travassos et al. 1999a and 1999b]. The technique considers two types of reading:

1. *Horizontal*. The purpose of the horizontal reading is to check different types of design documents (e.g. class diagrams, sequence diagrams, component diagrams) against each other;
2. *Vertical*. The vertical reading checks whether the design documents correspond to the requirements. For each check (whether between two types of diagrams or one type of diagrams and requirements) a scenario is developed.

2.1.4 Comparing Reading Techniques

The majority of work in the area of software inspection concerns testing and comparing different reading techniques [Jeffery & Scott 2002]. The non-systematic techniques (*Ad Hoc*, CBR) are usually compared versus systematic techniques (PBR). The main findings of the experimental evaluations are the following:

- Basili et al. conducted an experiment on the inspection of requirements documents [Basili et al. 1996a]. Laitenberger et al. reported on the inspection experiments on OO design diagrams [Laitenberger et al. 2000] and code [Laitenberger et al. 2001]. In all these experiments, PBR was more effective than CBR;
- Wohlin et al. collected and analyzed data from the software inspection experiments [Wohlin et al. 2002]. In total, 21 data sets from the requirements phase and 10 data sets from code inspections were collected. The comparison of the effectiveness using different reading techniques (*Ad hoc*, CBR, PBR) showed that CBR was more effective than other reading techniques;
- Biffl & Halling (2002) investigated the influence of inspector capability factors with CBR and PBR techniques on inspection performance. The inspection

object was a requirement document. The results showed that CBR inspectors were the most effective and efficient on an individual level;

- Shull showed that under certain conditions, the use of PBR technique led to improvement over non-systematic techniques [Shull 1998].

As we can see from the results of the aforementioned experiments, there is no unified answer concerning effectiveness of reading techniques. Therefore, this research considers the further investigation of two reading techniques – CBR and PBR. Since there is a lack of information about how inspections should be applied to OO artifacts [Dunsmore 2002; Laitenberger et al. 2000], we decided to apply CBR and PBR for inspection of OO design and compared the performance individual inspectors as well as 3-person inspection teams using these techniques (see Chapters 3, 4, and 5).

2.2 Evaluation of Cost-Effectiveness of Software Inspection

Inspection and testing are two main activities used for defect detection and removal in software products. Software testing cannot be conducted until software is implemented, whilst inspections can be applied in early stages of software development process and help to avoid costly rework. Therefore, the goal of inspections is to detect defect before the testing stage begins.

In order to compare inspections and testing, industrial case studies [Conradi et al. 1999] as well as experiments [Andersson et al. 2003] have been conducted. The results showed that inspection is significantly more effective and efficient than testing alone [Andersson et al. 2003].

One more important issue is cost-effectiveness of software inspections. Cost effectiveness is defined as the extent to which savings achieved are worth the costs. It is important to assess cost-effectiveness of inspections in order to be able to monitor the factors that affect it, its variations across different projects, and then to be able to improve it [Briand et al. 1999].

It has been claimed that software design inspections save on average 44% of the testing costs, while code inspections save on average 39% of those costs [Briand et al. 1998]. However, are the software inspections always worthwhile? How cost-effective preparation and inspection meeting stages are? The results of an industrial case study [Conradi et al. 1999] show, that while software inspections are indeed cost-effective, the inspection meeting is not cost-effective, comparing to individual inspection, in finding defects. Thus, inspection

cost-effectiveness has to be further investigated.

In order to evaluate the effectiveness of the inspections with respect to software development cost, several metrics have been previously proposed. The following sections describe software inspection costs and metrics used for evaluation of inspections.

2.2.1 Software Inspection Costs

Project managers have to consider the costs and benefits of quality assurance activities, such as software inspection, for project planning. The following are the costs of inspection [Biffl et al. 2001]:

- *Indirect costs* – are caused by the deciding to conduct an inspection. They do not include the effort of the inspectors involved, but comprise the costs for project planning and the delay of the project due to inspection;
- *Opportunity costs* – may arise, since inspectors cannot do other work that eventually, would be more beneficial than their contribution to inspection;
- *Direct costs* – those costs come from actually performing a quality assurance activity, such as the effort of the inspection team during inspection. They depend on the number of persons involved and the effort they invest into inspection.

2.2.2 Metrics to Evaluate Inspections

In order to evaluate the effectiveness of the inspections, several metrics have been previously proposed:

- Myers has proposed a metric to evaluate the effectiveness of inspection. The value of the metric is equivalent to the number of defects detected by inspectors [Myers 1978];
- Fagan has proposed a metric called *Error Detection Efficiency* for measuring inspection efficiency [Fagan 1976]. It is defined as the number of defects found by inspectors over the total number of defects in the software product before inspection;
- Collofello & Woodfield proposed a metric called *Cost Effectiveness* [Collofello & Woodfield 1989]. The cost effectiveness of an error-detection process is

defined as the ratio of the “cost saved by the process” to the “cost consumed by the process”;

- Kusumoto et al. proposed a metric for evaluation the *Cost Effectiveness* of inspection in terms of reduction of cost to detect and remove all defects from software product [Kusumoto et al. 1992; Kusumoto 1993]. It is a ratio of the reduction of the total costs to detect and remove all defects from the software product to the virtual testing cost (testing cost if no inspection is executed).

However, none of those metrics includes the information about false positives introduced during inspection, although the rework of false positives is costly, can introduce new defects, and eventually, might influence the cost-effectiveness of inspection [Land et al. 1997b; Sauer et al. 2000]. Therefore, there is a need of new metrics to address this issue.

2.3 Summary

Inspection is an effective method to detect and remove defects in different documents generated during the lifetime of a software project. Recently, the focus of detecting defects has moved away from group activity (inspection meeting stage) towards preparation stage of inspection, where individual inspectors read the artifacts with the guidance of reading techniques. However, most of available reading techniques were developed when the structured development process was dominant, and they may not address effectively the features of object-oriented artifacts. Therefore, there is a need for adequate reading techniques to increase the effectiveness and efficiency of inspectors. Furthermore, it is required further investigation of pros and cons of preparation and meeting stages of inspection. In this context, Sauer et al. have developed the Behaviourally Motivated Inspection Theory [Sauer et al. 2000]. However, it has been poorly tested in practice.

Finally, it has been indicated that the cost-effectiveness of inspection is an important issue, since software projects have to be delivered on time and within the budget. Several metrics have been proposed to evaluate the effectiveness of the inspections with respect to software development cost; however they do not consider all the aspects of inspection. For example, they do include the information of false positives, although it has been found that the rework of false positives is costly, may introduce new defects, and eventually influence inspection cost-effectiveness [Land et al. 1997b; Sauer et al. 2000]. Hence, it has been

observed that new metrics are needed to enable more precise evaluation of software inspections.

CHAPTER 3

Evaluation of Two Reading Techniques for Object-Oriented Design Inspection (Experiment 1)

3.1 Introduction to Experiment I

To the best of our knowledge, little work has been done in the area of inspection of OO design document written in the notation of Unified Modelling Language (UML) [Booch et al. 1999]. One of few examples is a controlled experiment described by Laitenberger et al. (2000). In this work, the authors compared two reading techniques, PBR and CBR, in a controlled experiment with eighteen subjects and two software systems. The results showed that 3-person inspection teams, which used PBR, had 41% effectiveness improvement and 58% cost per defect improvement over CBR teams. It has been indicated, that it is necessary to conduct software inspection experiments in different environments, using different people, languages, documents, etc. in order to understand all the aspects of software inspections more completely.

We conducted a controlled experiment in Osaka University with 59 student subjects to compare CBR and PBR techniques for the UML diagram inspection. The experiment is not a direct replication of [Laitenberger et al. 2000]. Hence, different subjects, objects and different experiment design were used. Concerning data analysis, Laitenberger et al. (2000) analyzed 3-person real team data; however, in this experiment individual inspector performance was analyzed. The defect detection effectiveness, cost per defect, and time spent on inspection of individual inspectors using CBR vs those using PBR was compared.

3.2 Experiment Planning

In this section, the planning of the inspection experiment is described. It includes experimental variables, hypotheses and design of the experiment.

3.2.1 Variables

Two types of variables are defined for the purpose of the experiment: independent variables and dependent variables. Independent variables may include reading technique, simulated team size and composition, duration of experiment, experience of subjects, etc. However, only reading technique was considered as an independent variable. The dependent variables, considered in this experiment, are described in Table 3.1.

Table 3.1 Dependent variables

Dependent variable	Formula	Measurement units
Number of defects detected by a subject		Number of defects
Time spent on inspection		Minutes
Defect detection effectiveness	Effectiveness = (Number of defects found by inspector / Total number of seeded defects) * 100	Percent
Cost per defect	Cost per defect = Time spent on inspection / Number of defects found by inspector	Minutes
Average number of defects detected by subjects		Number of defects
Average time spent on inspection		Minutes
Average inspector effectiveness	Average inspector effectiveness = (Number of inspectors who have found defects / Total number of inspectors) * 100	Percent
Average defect detection effectiveness	Average defect detection effectiveness = (Average number of defects found by subjects / Total number of seeded defects) * 100	Percent

Dependent variables include the variables calculated for each subject such as the number of defects found, time spent on inspection, defect detection effectiveness and cost per defect. In addition, the average values of number of detected defects, time spent on inspection, inspector effectiveness and defect detection effectiveness for subjects who used CBR and those who used PBR inspection technique were measured. Beside actual defects, false positives have been detected. Only the actual defects were evaluated in this research. False positives are evaluated in Chapter 5.

3.2.2 Hypotheses

Three hypotheses have been stated before experiment. They were based on the following assumptions:

- We assumed (H_{01}) that the subjects who used PBR technique during inspection should spend less time on inspection than those who used CBR, because a PBR scenario covers only the UML documents related to a corresponding perspective and reviewer does not need to examine UML documents not related to his perspective. However, subjects who used CBR were supposed to examine all UML documents, and they needed to spend more time on the inspection.

- In addition, we assumed (H_{02}) that subjects who used PBR should have higher cost per defect, because they need not only to answer the questions, but also to perform various tasks before answering the questions. Cost per defect evaluates only the average time spent to detect one defect using different inspection techniques. In different papers authors often relate the costs to either the size of inspected product (for example, cost is the amount of time spent on code inspection per thousand lines of code) or the number of defects found (amount of time spent to detect one defect) [Laitenberger & DeBaud 2000]. We used the [Laitenberger et al. 2000] experiment as a reference in planning of our experiment, where authors evaluated cost per defect; therefore we decided to evaluate cost per defect as well.

- We did not know how different could the defect detection effectiveness of both methods be, so we only assumed (H_{03}) that PBR defect detection effectiveness should be different from CBR defect detection effectiveness.

The following null hypotheses were stated:

H_{01}: Subjects spend more time on inspection using PBR than using CBR;

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

H_{03}: There is no difference in defect detection effectiveness of subjects who use PBR inspection technique as compared to subjects who use CBR.

3.2.3 Experimental Design

Since we wanted to compare two inspection techniques CBR and PBR against each other, we chose design type of “one factor with two treatments” [Wohlin et al. 2000]. This is a simple experiment design for comparing two treatment means. The design of experiment is shown in Figure 3.1. 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). The number of subjects in each group is shown in Figure 3.1. Software systems used during experiment are described in the Section 3.4.2.

Each inspector group included students with the same mix of abilities based on the marks of Program Design course, in which students have been taught UML. In addition, we checked the groups to be similar according to the results, which students had shown during the training session.

	PBR (treatment 1)			CBR
	User	Designer	Implementer	(treatment 2)
Seminar system	7	6	6	11
Hospital system	7	6	6	10

Figure 3.1 Experimental design.

3.3 Two Inspection Reading Techniques

We developed the checklist for inspection using CBR technique and three scenarios for inspection using PBR – User’s, Designer’s and Implementer’s. Defect registration form for registration of defects and Comment form were developed as well.

3.3.1 Checklist

A diagram-specialized checklist was developed for the inspection experiment. It included 20 questions about Class, Activity, Sequence and Component diagrams. This is in line with the recommendations of Gilb and Graham [Gilb & Graham 1993] that a checklist should not be longer than a page (approximately 25 items). Inspectors had to answer “Yes” or “No” to each question. Negative answer to the question indicated that a defect was detected, and inspectors needed to fill in

information about defects into the Defect registration form. Some defects were considered possible to detect using several questions of the checklist. In case a defect had been already detected using another checklist question, student had to write this in the Comment form, and to proceed to the next question of the checklist. The checklist (translation from the Japanese language) is given in the Figure A1 in Appendix A.

3.3.2 Perspective-Based Reading

We used three perspectives in this experiment: User's, Designer's and Implementer's:

- *User's perspective*. The concern of the User is to ensure that the specification of the system operation at the end of analysis phase is complete, error free and that satisfies user requirements. It means that there must be no inconsistencies between the various analysis models, such as Requirements specification, Use-Case, Activity and Sequence diagrams.
- *Designer's perspective*. 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).
- *Implementer's perspective*. 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 were developed: User's, Designer's and Implementer's. Each scenario included introduction and instructions how to proceed during inspection. Inspection consisted of several steps. Each of them included the following information: diagrams to inspect, tasks to carry out, and questions to answer.

The User's scenario is given in Figure A2, the Designer's scenario – in Figure A3, and the Implementer's scenario – in Figure A4 in Appendix A.

3.3.3 Inspection Defect Forms

Each inspector was given two forms during inspection:

1) *Individual defect registration form* for registration of defects. It is given in Figure A5 in Appendix A. It this form, inspector had to fill in his/her name, exact time when he started reading, the data of each defect, and inspection ending time. The data of each defect consisted of: defect number; UML diagram in which defect was detected; Checklist/Scenario item which led to defect detection; exact defect detection time. After finishing reading, inspector had to answer questions, included at the end of the form;

2) *Comment form* for writing additional comments. It was a blank sheet of paper. Inspectors could write any kind of comments related to inspection in this form. Moreover, the subjects who used PBR technique had to use this form in order to perform tasks specified in their scenarios.

3.4 Experimental Subjects and Objects

This section discusses the subjects and objects used in the experiment. Subjects refer to the inspectors, and objects refer to the software artifacts inspected. A description of defects is given as well.

3.4.1 Experimental Subjects

Subjects were 59 participants in the 3rd year of the Software Development course of Osaka University. They have had previous classroom experience with the programming languages, Object-Oriented development, UML, software design activities and conventional software review.

The class was divided into two groups of 29 and 30 students, and each group included subjects with the same mix of abilities (based on marks from Program Design class). Each group then focused on inspection of one software system. Inside each group, subjects were divided into two subgroups, each of them focused on only one inspection technique (PBR or CBR).

After the experiment, we asked the students to fill in a feedback questionnaire. The aim of the questionnaire was to verify students understanding of UML diagrams and software systems used during the experiment. The results of student answers to those questions showed, that both students who used CBR and those who used PBR had similar level of understanding. Beside those questions, we asked students if this experiment was their first inspection

experiment. For most of the students, it was the first experiment. In addition, we have inquired if students followed the instructions of checklist and scenarios. The data of the students who did not conform to the process was eliminated from further analysis.

3.4.2 Experimental Objects

UML diagrams (paper-documents) of two software systems (Seminar system and Hospital system), borrowed from Itoh et al. [Itoh et al. 2001], were used as inspection objects. We have borrowed the Use-Case, Activity, Class and Sequence diagrams. In addition, we developed Component diagrams. The 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. The Hospital system included activities such as oral consultation, medical examination, treatment of patients, prescription of medicines, etc. The number of diagrams for each system is given in Table 3.2. The size of Seminar system documentation was 24 pages, and the size of Hospital system documentation was 18 pages.

Table 3.2 Number of Diagrams in Experiment I

UML diagram	Number of diagrams	
	Seminar system	Hospital system
Class	1	1
Activity	8	7
Sequence	12	7
Component	1	1

At the beginning of the project, we held a training session in order to improve student's understanding of the software systems used. Students received description of the requirements, Use-case diagram and a part of the Class diagram, and were asked to create Sequence and Component diagrams of those systems.

During the experiment, system requirements description and Use-case diagram were assumed to be defect-free. The rest of the diagrams might contain defects. At least three defects were inserted into each type of UML diagrams (Class, Activity, Sequence and Component).

Students who were using CBR needed to inspect all the diagrams of the

corresponding system. However, students who used PBR technique during inspection were inspecting only documents relevant to a specific perspective.

The assignment of UML documents to inspection perspectives is shown in Table 3.3. The assignment was based on a UML diagram development process, which students were learning during Software Design course. The main steps of this process are:

(1) Development of Use-case diagrams;

(2) Describing system activities in Activity diagrams;

(3) Defining static structure of the system in Class diagrams;

(4) Modelling dynamic aspects of the system in Sequence diagrams;

(5) Detailed description of object states in Statechart diagrams;

(6) Development of the Component diagrams.

The User's perspective in our experiment covered the second, and partially the third and the fourth steps of software development process; the Designer's perspective covered the third and the fourth steps; and the Implementer's perspective covered the sixth step, and partially the third and the fourth steps.

Table 3.3 Assignment of objects to checklist and scenarios

UML diagram type	Checklist (CBR technique)	Scenarios (PBR technique)		
		User	Designer	Implementer
Class	✓	✓	✓	✓
Activity	✓	✓		
Sequence	✓	✓	✓	✓
Component	✓			✓

3.4.3 Defects

In [Travassos et al. 1999a; ESEG] authors describe defect taxonomy for UML design diagrams that previously had been proven effective for requirement's defects [Basili et al. 1996a and 1996b]. This taxonomy classifies defects by identifying related sources of information, which are relevant for the software system being built. Authors defined five types of defect:

1) *Omission* – one or more design diagrams that should contain some concept

from the general requirements or from the requirements document do not contain a representation for that concept;

2) *Incorrect Fact* – a design diagram contains a misrepresentation of a concept described in the general requirements or requirements document;

3) *Inconsistency* – a representation of a concept in one design diagram disagrees with a representation of the same concept in either the same or another design diagram;

4) *Ambiguity* – a representation of a concept in the design is unclear, and could cause a user of the document to misinterpret or misunderstand the meaning of the concept;

5) *Extraneous Information* – the design includes information that, while perhaps true, does not apply to this domain and should not be included in the design.

We summarized defect taxonomy proposed by Travasos et al. [Travassos et al. 1999a; ESEG] into three types of defects:

a) *Syntactic*, which include omission and extraneous information defects;

b) *Semantic*, which include incorrect facts and ambiguity defects;

c) *Consistency*, which correspond to inconsistency defects.

Before the creation of defects, we have analysed the inspection objects, defined defect taxonomy and developed the checklist and scenarios. After that, we created defects by ourselves. Then we asked several colleagues to try to find those defects using checklist and scenarios. We received some important comments from colleagues, and considering them, we created the final list of defects. UML diagrams were created and defects were inserted using Rational Rose2001 Professional Java Edition software. In total fifteen defects were inserted into the software documents: 3 into the Class diagrams, 4 into the Activity diagrams, 5 into the Sequence diagrams, and 3 into the Component diagrams.

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 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 was different for different classes of defects. To minimize its impact to the setting of our research hypotheses, we inserted similar mix of defect classes into each type of UML diagrams.

3.5 Experiment Operation

Experiment was conducted in academic environment during a Software Development course in December 2001. The language of experiment was Japanese. The following timetable was used to arrange the experiment:

Week 1: Training session to improve students' understanding of the systems. The class was divided into two groups of 29 and 30 students. One of the groups received Requirement's description, Use-case diagram and part of Class diagram of a Seminar system. The other group received the above-mentioned documents of a Hospital system. Students were asked to create Sequence and Component diagrams of each system.

Week 2: 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 two groups: 38 (for PBR technique) and 21 (for CBR technique). Before the experiment students listened to the explanations, which lasted approximately 20 minutes. After explanations were given, the experiment was conducted. Experiment consisted of 120-minute (excluding explanations) individual inspection task. Students were inspecting the same software system they had analyzed during the training session.

Week 3: Feedback questionnaire to collect additional information from students. The results from the questionnaire showed that inspectors who used CBR and those who used PBR had similar level of difficulty to understand checklist and scenarios; however inspectors who used PBR had better understanding of software systems they inspected. Most of the students had no previous experience in software inspection experiments, and most of them stated that such experiments could be useful in practice.

3.6 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 [Wohlin et al. 2000]:

- *Threats to internal validity* are threats that can affect the independent variable with respect to causality without the researcher's knowledge. In our experiment, there are no threats to history, maturation or mortality, because subjects participated only in one treatment and it lasted no longer than 2.5 hours. 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. In addition, we checked the groups to be similar in aspect of the level of student's knowledge. 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. There was no risk for subjects to lack motivation, because students were told that the grading of the course would depend on their performance during inspection. To check the process conformance of the inspectors, we used the Defect registration forms, where student were asked to write which item of the Checklist or Scenario he used to detect each defect. After the experiment, we have checked if those items were in correct sequence, as it was defined in Checklist or Scenario. In addition, we have checked 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. Finally, the question about process conformance was included into the feedback questionnaire given to the inspectors after experiment. The data of the students who did not conform to the process has been eliminated from the further analysis.

- *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 professional start in industry. There are more experiments reported in the

literature, where students were successfully used as subjects [Höst et al. 2000; Land et al. 1997b; Travassos et al. 1999a]. 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* concert 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.

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

3.7 Experiment Data Analysis

3.7.1 Experiment Data of Individual Inspectors

Two types of data were collected during the experiment, time data and defect data. Time data showed the time each subject spent during the inspection. The added defect data showed the number of defects, which were detected by the subject. Data gathered during experiment is shown in Table 3.4. For each software system (Seminar and Hospital), inspected during the experiment, the following information is given: number of inspectors who used corresponding checklist or scenario, defect data and time data. Data related to defects consists of a number of defects, which could be detected using corresponding scenario, and average, maximum and minimum values of detected defects. The time-related data consists of the average, maximum and minimum time spent on inspection.

The inspectors' effectiveness to detect each defect is shown in the Figure 3.2. Defects are depicted in the X-axis. Y-axis shows the percentage of inspectors who have detected the defect. The defect distribution according to the diagram type is shown in the bottom of the Figure 3.2.

Table 3.4 Experiment data of individual inspectors

Software system	Checklist and scenarios	Number of inspectors	Defects			Time spent on inspection (minutes)	
			No of seeded defects	Average No of detected defects	max/ min	Average time	Max /min
Seminar	User's	7	7	4.4	6/3	60.4	90/46
	Designer's	6	6	5.0	6/4	65.5	80/51
	Implementer's	6	9	6.5	9/5	76.7	95/40
	Checklist	11	15	10.6	13/8	74.6	90/62
Hospital	User's	7	7	4.4	6/3	48.3	70/25
	Designer's	6	6	3.8	5/3	59.2	73/30
	Implementer's	6	9	6.3	7/5	63.3	77/44
	Checklist	10	15	10.5	12/8	70.1	94/60

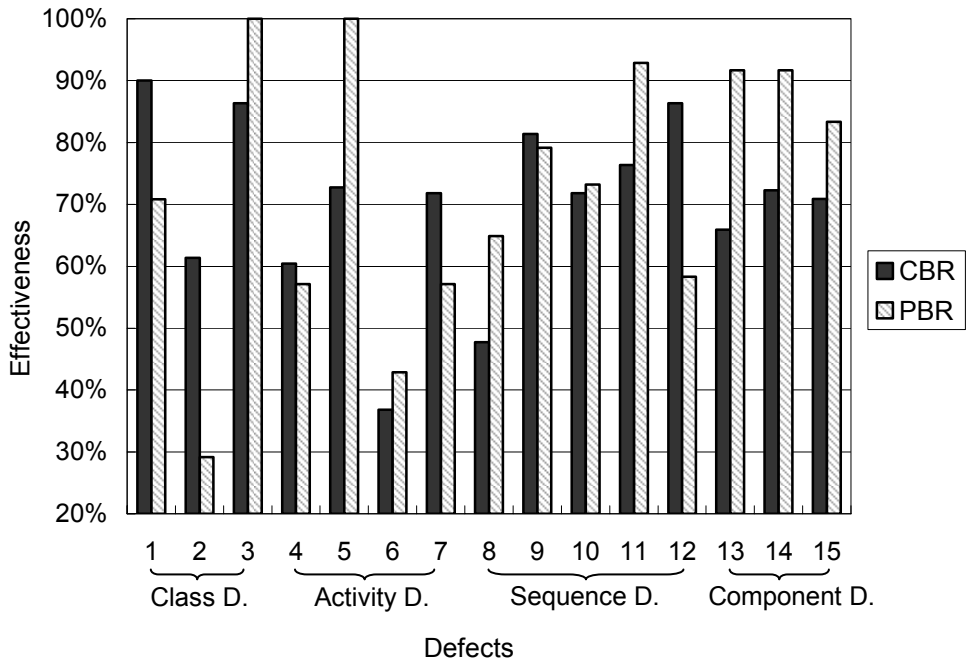


Figure 3.2 Inspector effectiveness for each defect.

3.7.2 Comparison of CBR and PBR Inspector Average Effectiveness Values

The average effectiveness of individual inspectors using CBR and PBR techniques has been compared while detecting:

1) Defects belonging to different PBR perspectives (User's, Designer's and

Implementer's);

2) Different types of defects (syntactic, semantic and consistency);

3) Defects in different UML diagrams (Class, Activity, Sequence and Component).

The following subsections describe the results of these comparisons.

Detecting defects belonging to different PBR perspective

We compared CBR and PBR with respect to percentage of inspectors who had found defects belonging to User's perspective, Designer's perspective, Implementer's perspective and all perspectives. Results are shown in Figure 3.3. We used independent samples t-test with significance interval of 95% ($p<0.05$) to compare inspector effectiveness between CBR and PBR for each group of defects. The results of comparison showed that there is no statistically significant difference between CBR and PBR inspectors.

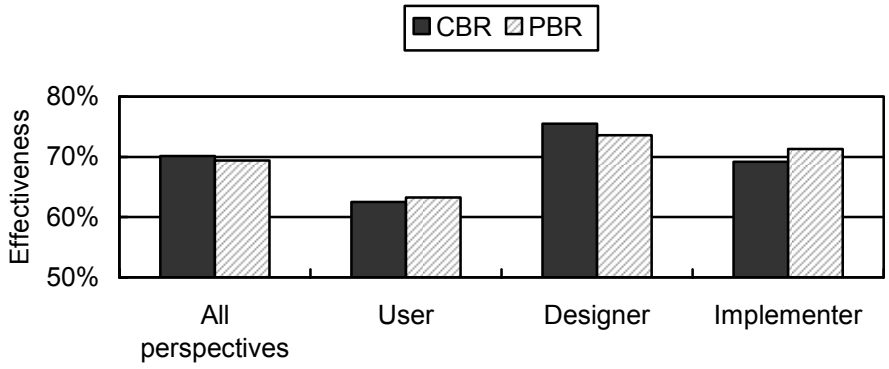


Figure 3.3 Inspector effectiveness according to defects belonging to different perspectives.

Detecting different types of defects

Three types of defects were inserted into software systems – syntactic, semantic and consistency defects. We compared CBR and PBR inspection techniques with respect to inspector effectiveness for those three types of defects. Results are given in Figure 3.4. We used independent samples t-test with significance interval of 95% ($p<0.05$) to compare effectiveness of CBR and PBR according to the different defect types. The results of this comparison showed that there is no

statistically significant difference between CBR and PBR techniques in syntactic and semantic defect detection, however there is statistically significant difference in consistency defect detection. In other words, PBR technique is more effective than CBR to detect consistency defects (PBR: 82%; CBR: 68%; p-value: 0.046). PBR provides inspectors with more guidance during inspection process, especially in detection of consistency defects among different types of diagrams, and this can lead to higher effectiveness in detection of consistency defects.

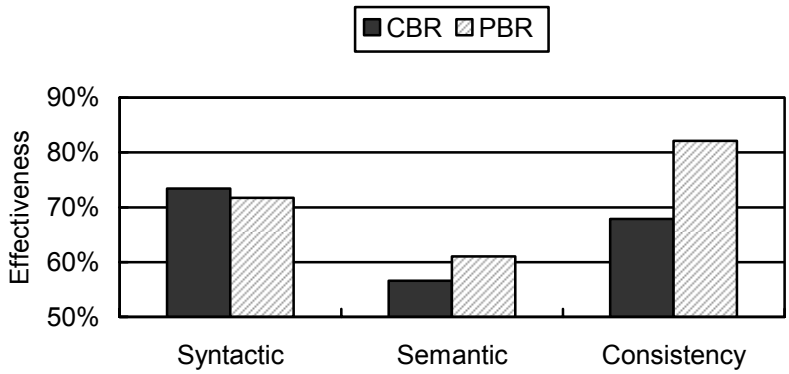


Figure 3.4 Inspector effectiveness according to defect types.

Detecting defects in different types of UML diagrams

Defects were seeded into the four types of UML diagrams – Class, Activity, Sequence and Component diagrams. We compared average inspector effectiveness for each type of diagrams. Results are shown in Figure 3.5.

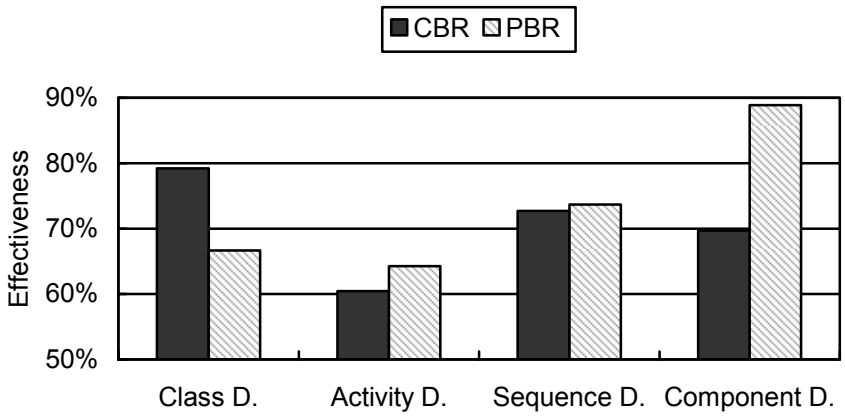


Figure 3.5 Inspector effectiveness according to UML diagram types.

We used independent samples t-test with significance interval of 95% ($p<0.05$) to compare effectiveness of CBR and PBR according to UML diagram types. The results of this comparison showed, that there is no statistically significant difference between CBR and PBR techniques in detection of Class, Activity and Sequence diagram defects, however there is a statistically significant difference in Component diagram effects. PBR technique is more effective than CBR to detect defects in Component diagrams (PBR: 89%; CBR: 70%; p-value: 0.027).

3.7.3 Statistical Comparison of CBR and PBR

We compared time spent on inspection, cost per defect (average time spent to detect one defect) and defect detection effectiveness (percentage of seeded defects which were detected) of subjects who used CBR and those who used PBR technique during inspection. The statistics of those variables are given in Table 3.5.

As we can see from Table 3.5, inspectors who used PBR inspection technique spent on the average 18% (11 min) less time on inspection than inspectors who used CBR. Cost per defect of inspectors who used CBR is 39% lower (4 min/defect) than of inspectors who used PBR. Inspectors who used PBR and those who used CBR exhibited similar defect detection effectiveness (about 70%). Parametric (Independent samples t-test) and non-parametric (Mann-Whitney) tests [Juristo & Moreno 2001; Norŭsis 1995; Wohlin et al. 2000] were used to test the hypotheses for the individual inspectors.

Table 3.5 Statistics for time spent on inspection, cost per defect and defect detection effectiveness

Variables	Reading technique	Number of subjects	Mean	SD	SE	95% CI of Mean	Median	IQR	95% CI of Median
Time spent on inspection	CBR	21	62.9	11.7	2.5	57.6 to 68.2	61.5	15.7	54.3 to 70.0
	PBR	38	51.3	15.1	2.5	46.3 to 56.3	50.0	17.1	44.2 to 59.2
Cost per defect	CBR	21	6.2	1.6	0.4	5.4 to 6.9	6.1	2.7	4.9 to 7.6
	PBR	38	10.2	3.5	0.6	9.0 to 11.4	9.9	4.1	8.5 to 11.6
Defect detection effectiveness	CBR	21	70.2	11.5	2.5	64.9 to 75.4	73.3	20.0	60.0 to 80.0
	PBR	38	69.1	15.3	2.5	64.0 to 74.1	69.1	22.0	66.7 to 77.8

The statistical results of testing hypotheses H_{01}, H_{02} and H_{03} are shown in Table 3.6.

Table 3.6 Statistics for t-test and Mann-Whitney tests

Statistics	Time spent on inspection (H_{01})	Cost per defect (H_{02})	Defect detection effectiveness (H_{03})
t-test P value	0.0036	<0.0001	0.78
t-test t value	3.04	4.97	0.28
Mann-Whitney test P value	0.0043	<0.0001	0.71
Mann-Whitney test U value	212	102	376

Hypotheses H_{01} and H_{02} can be rejected, but hypothesis H_{03} can not be rejected. In other words, it is statistically significant that subjects spend more time on inspection using CBR inspection technique than using PBR inspection technique. In addition, it is statistically significant that cost per defect of subject who used PBR inspection technique is higher than the cost per defect of those who use CBR. However, there is no statistical significant difference in defect detection effectiveness of individual reviewers between CBR and PBR inspection techniques.

3.8 Interpretation of Results

The following hypotheses show significant results:

- H_{01} – Subjects spend more time on inspection using PBR than using CBR. (t-test P = 0.0036; Mann-Whitney test P = 0.0043)
- H_{02} – Cost per defect of subjects who use PBR is lower than cost per defect of subjects who use CBR. (t-test P < 0.0001; Mann-Whitney test P < 0.0001)

The below hypothesis did not show significant results:

- H_{03} – There is no difference in defect detection effectiveness of subjects who use PBR inspection technique as compared to subjects who use CBR. (t-test P = 0.7769; Mann-Whitney test P = 0.7145)

The results of the individual data analysis show that the subjects who use PBR technique spend 18% less time on inspection than subjects who use CBR technique. This result might be influenced by the amount of objects assigned to

the two techniques: fewer objects were assigned to PBR as compared to CBR (see Table 3.3). Therefore, subjects who used PBR needed less time to check inspection objects.

In addition, the results show that cost per defect of PBR subjects is 39% higher. This result might be influenced by the procedural difference of the two techniques: subjects who used CBR were requested to answer questions during inspection, while subjects who used PBR were requested to perform various tasks along with answering the questions. Therefore, the subjects who used PBR consumed more time to detect defects.

Comparing individual inspector effectiveness, the inspectors who used PBR were more effective than those who used CBR in detection of consistency defects (PBR: 82%; CBR: 68%; p-value: 0.046) and Component diagram defects (PBR: 89%; CBR: 70%; p-value: 0.027). However, there was no statistically significant difference between CBR and PBR inspector effectiveness in detection of syntactic and semantic defects, and Class, Activity and Sequence diagram defects. Those results are not statistically significant due to high variability of the performance between the best and the worst performing inspectors for each inspection technique.

The results of this experiment are in line with the results of several experiments of requirement and code inspections. In [Wohlin et al. 2002] authors collected data from the software inspection experiments reported in literature. In total, 21 data sets from the requirements phase and 10 data sets from code inspections were collected. The comparison of the effectiveness in inspection using different reading techniques (*Ad hoc*, CBR, PBR) showed that CBR was more effective than other reading techniques. In [Dunsmore et al. 2002] authors reported on experiment of OO code inspection. The results of this experiment showed, that CBR emerged as the most effective approach. In [Biffl & Halling 2002] authors investigated influence of inspector capability factors with CBR and PBR techniques on inspection performance. The inspection object was a requirement document, which also contained UML diagrams. The results showed that CBR inspectors were the most effective and efficient at individual level.

The experiment included only preparation stage of inspection, while inspection meetings have not been performed. The following chapter presents the second experiment intended for further investigation of CBR and PBR techniques that includes both preparation and inspection meeting stages.

3.9 Conclusions

This chapter has presented an inspection experiment to evaluate two reading techniques, CBR and PBR. We have adapted these techniques for inspection of object-oriented design documents, written using the notation of Unified Modelling Language, and developed a checklist and three scenarios to guide individual inspectors during preparation stage of inspection.

The results of the experiment indicate that:

- CBR and PBR are effective techniques for inspection of Object-Oriented design, and lead to detection of on average 70% of defects;
- Time spent on inspection of subjects who use PBR is lower than of subjects who use CBR;
- Cost per defect of PBR subjects is higher as compared to CBR subjects, therefore CBR is more efficient;
- There is no difference in defect detection effectiveness of subjects who use PBR inspection technique as compared to subjects who use CBR.

CHAPTER 4

Investigating Individual and 3-Person Team Performance (Experiment 2)

4.1 Introduction to Experiment 2

This chapter describes the second experiment (Experiment 2) conducted in July 2002 in Osaka University with 54 subject students. The goals of Experiment 2 were twofold:

1) Verify the results of Experiment 1 (see Chapter 3);

2) 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, individual defect registration form.

4.2 Experimental Hypotheses

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

4.2.1 Hypotheses for Individual Inspectors

The results of Experiment 1 indicate that subjects who use PBR inspection technique have similar effectiveness and greater cost per defect as compared to the subjects who use CBR (see Chapter 3). 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.

4.2.2 Hypotheses for 3-Person Inspection Teams

Performing various tasks during PBR is expected to result in a better understanding of the document. Therefore, during the meeting, inspectors do not have to spend a lot of extra effort explaining to their colleagues in a team the defects that they found. 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 investigating the usefulness of inspection meeting are contradictory. Fagan [Fagan 1976] reported that the number of defects identified in meeting was much higher that in individual work. However, Porter et al. [Porter et al. 1995] 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. Since the final defect list is the list of defects recorded during inspection meeting, the defects that have been recorded during individual inspection but not included into defect list during meeting are "lost". The following null hypotheses were 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.

4.3 The Experiment

This section describes the instruments and procedure of the experiment. The following elements are the same as used during Experiment 1:

a) Inspection techniques (CBR checklist and PBR scenarios). Checklist is described in Section 3.3.1 and given in Figure A1 in Appendix A; scenarios are described in Section 3.3.2 and given in Figures A2-A4 in Appendix A;
b) Experimental objects (Seminar and Hospital system). They are described in Section 3.4.2;
c) Defect types (syntactic, semantic and consistency defects). The detail description is given in Section 3.4.3;
d) Individual defect registration form, used during preparation stage. It is described in Section 3.3.3 and given in Figure A5 in Appendix A.

4.3.1 Variables

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 4.1 and Table 4.2 respectively. Dependent variables for individual subjects were calculated for each subject, including the number of defects found, time spent on inspection, defect detection effectiveness and cost per defect (Table 4.1). 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, and cost per defect (Table 4.2). In addition, we calculated the meeting gains (number of new defects first detected during team meeting) and meeting losses (number of defects first detected by individual inspectors but not reported by a

team).

Table 4.1 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/Total number of seeded defects)*100	Percent
C	Cost per defect C=T/D	Minutes

Table 4.2 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 individual inspection and team meeting	Minutes
TE	Defect detection effectiveness TE=(TD/Total number of seeded defects)*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 individual inspectors but not reported by a team	

Meeting losses were calculated by comparing Individual and Team defect registration forms. Individual defect registration form (see Figure A5 in Appendix A) showed whether or not an inspector discovered a particular defect. Meanwhile, Team defect registration form (see Figure C1 in the Appendix C) 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 members, 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.

4.3.2 Experimental Subjects and Their Training

Subjects were 54 third year Bachelor students of Software Design course at Osaka University with previous classroom experience on programming languages, object-oriented development, UML, software design activities, and conventional software review. One week before the experiment a training session to improve students' understanding of the systems that would be inspected during experiment was conducted.

4.3.3 Experimental Design

The design of the experiment is shown in Table 4.3. 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.

Table 4.3 Experimental design

Groups	Individual inspection			Number of teams during inspection meeting
	Reading technique	Software system	Number 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

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).

4.3.4 Experimental Objects and Seeded Defects

The UML diagrams [Booch et al. 1999] of two software systems (Seminar system and Hospital system), used in Experiment 1, were used in this experiment.

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 4.4.

Table 4.4 Experimental objects

UML diagram type	Number of pages		CBR checklist	PBR scenarios		
	Seminar system	Hospital system		User's	Designer's	Implementer's
Use-Case	1	1	✓	✓	✓	✓
Class	1	1	✓	✓	✓	✓
Activity	3	1	✓	✓		
Sequence	9	10	✓	✓	✓	✓
Component	1	1	✓			✓

Although we used the same types of defects as in Experiment 1 (syntactic, semantic and consistency), we modified some of defects. The number of defects inserted in each software system is given in the Table 4.5.

Table 4.5 Seeded defects

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

4.3.5 Experimental 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 were given the explanations for about 20 minutes.

2) After the explanations, an individual inspection stage was conducted, which lasted 60 minutes. Students inspected the same software system they had analyzed during the training session.

3) After the individual inspection stage was completed, the subjects were assigned into 3-person teams to perform 30-minute inspection meetings. Each team was given a document with guidelines on how to perform the inspection meeting. Teams were requested to register actual defects into a Team defect registration form (see figure C1 in the Appendix C).

4.4 Data Analysis

This section describes the data collected during experiment and those statistical tests used during data analysis. Section includes 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).

4.4.1 Individual Data Analysis

Two types of data were collected during the experiment, time data and defect data. Time data shows the time spent by each subject during inspection. The time spent on explanations before individual inspection (20 min) has been added to the inspection time. The added defect data showed the number of defects, which were detected by the subject. The data gathered during the individual inspection stage is shown in Table 4.6 (in Table 4.6, “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 [Norŭsis 1995; Wohlin et al. 2000] 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 4.7. 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.

Table 4.6 Means and standard deviation of dependent variables collected from the individual inspection stage

System	Inspection technique	Defects			Inspection time	
		Seeded	Detected		Mean	Std. deviation
			Mean	Std. deviation		
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	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	0.5
	D	6	3.5	1	75.5	1.73
	I	7	3	1.41	75.3	3.77

Table 4.7 Results of hypotheses H_{01} and H_{02} testing

System	Dependent variable	Inspection technique	Mean	Std. deviation	p value (2-tailed)
Hospital	Effectiveness (H_{01})	CBR	47.0	11.97	0.946
		PBR	47.4	14.58	
	Cost per defect (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 defect (H_{02})	CBR	15.6	5.87	0.008
		PBR	24.8	10.56	

4.4.2 Team Data Analysis

The data collected during inspection meetings is given in the Table 4.8. 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 [Norŭsis 1995; Wohlin et al. 2000] 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 4.9.

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 CBR and PBR 3-person inspection teams.

Table 4.8 Means and standard deviation of dependent variables collected from inspection meeting stage

Variable	Statistics	Values of statistics			
		Hospital system		Seminar system	
		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	291
	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 4.9 Results of hypotheses H_{03} and H_{04} testing

System	Dependent variable	Inspection technique	Mean	Std. deviation	p value (2-tailed)
Hospital	Effectiveness (H_{03})	CBR	62.5	8.99	0.96
		PBR	62.86	11.74	
	Cost per defect (H_{04})	CBR	35.2	6.23	0.56
		PBR	32.7	5.99	
Seminar	Effectiveness (H_{03})	CBR	52.31	6.44	0.26
		PBR	48.08	3.85	
	Cost per defect (H_{04})	CBR	47.2	6.62	0.91
		PBR	46.8	3.59	

In order to test the hypotheses H_{05} and H_{06} for inspection teams, a paired samples t-test [Norŭsis 1995; Wohlin et al. 2000] with significance interval 95% ($p<0.05$) has been used. The statistical results of hypotheses testing are given in Table 4.10.

As we can see from Table 4.10, 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.

Table 4.10 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}) PBR (H_{06})	0.041 0.587
Seminar	Meeting gains – meeting losses	CBR (H_{05}) PBR (H_{06})	0.016 0.069

4.5 Interpretation of Results

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 a significant result might be the limited inspection time (the maximum time for individual inspection was 60 minutes; and the maximum time for inspection meeting was 30 minutes). Due to this limitation, 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 [Johnson & Tjahjono 1998; Porter et al. 1995; Votta 1993], which have reported that inspection teams detect on average less than 10% of all defects during team meeting.

Porter et al. (1995) compared the performance of inspection teams which used *Ad hoc*, CBR and Scenario-based reading techniques. The results of comparison among techniques did not reveal any difference 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.

The following chapter presents further investigation of inspection meetings using the data of Experiment 2. It compares CBR and PBR teams with respect to the number of false positives, number of overlapping defects and their ability to detect different defect types. Furthermore, it tests several propositions from the behaviourally motivated inspection theory.

4.6 Conclusions

This chapter presented an inspection experiment focused on the investigation of two stages of inspection process: preparation and inspection meeting.

The results of the individual inspection stage of experiment confirmed the results of 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.

CHAPTER 5

Assessing Inspection Meetings

5.1 Introduction

This chapter presents a further investigation on inspection meetings. The research is made using the data collected from Experiment 2 described in Chapter 4. The goals of the research were twofold:

1) Investigate the performance of interacting inspection teams using CBR vs. those using PBR. In Chapter 4, we compared CBR and PBR with respect to defect detection effectiveness, cost per defect and difference between meeting gains and meeting losses. In this research, we compare the following variables of two reading techniques: (a) number of false positives, (b) number of overlaps, and (c) the ability to detect different defect types;

2) Verity a part of the theory of group software inspections proposed by Sauer et al. (2000) (see Section 2.1.2) by testing several of its propositions. We decided to test a part of this theory, because it is the first theory on inspections, however it has scarcely been tested, and because "*empirical studies should be used to confirm that what works in theory can actually be used (and useful) in practice*" [Parnas 2003]. We tested the following propositions:

   P4. The interacting group meeting does not improve group performance over nominal group by discovering new defects. Group meetings do not discover a significant number of new defects beyond the aggregation of those discovered by individuals (often referred to as the nominal group). This proposition is supported by Votta (1993), Porter et al. (1995), Lanubile & Visaggio (1996), Land et al. (1997a).

   P8: The performance advantage of an interacting group over a nominal group is a function of the level of false positives discovered by individuals. Several authors discovered that the source of performance advantage of interacting groups is their ability to discriminate between true defects and false positives; consequently interacting groups report fewer false positives than nominal groups [Johnson & Tjahjono 1998; Land et al.

1997b].

P10. Nominal groups outperform alternatives at the discovery task. Lanubile & Visaggio (1996) and Land et al. (1997b and 2000) have reported that nominal groups discover more defects as compared to the interacting groups.

5.2 Goals and Hypotheses

Table 5.1 shows the relationship between goals and hypotheses presented in this research.

Using a Goal/Question/Metric (GQM) template [Basili et al. 1994], the first goal of the experiment can be stated as:

G1. **Analyze** the inspection techniques for UML diagram inspections **for the purpose of** assessment **with respect to** number of false positives, number of defect overlaps, and ability to detect different defect types of interacting team **from the point of view of** the researcher.

Beside number of false positives and ability to detect different defect types, we compared inspection techniques with respect to number of defect overlaps, because overlaps indicate the amount of time wasted during inspection to detect the same defect: the higher number of overlaps is the greater amount of time is wasted.

Based on the goal G1, the following hypotheses have been formulated:

H1.1. There is a difference in false positives for 3-person interacting teams using CBR vs. those using PBR.

H1.2. There is a difference in defect overlaps for 3-person interacting teams using CBR vs. those using PBR.

H1.3. There is a difference in effectiveness detecting different defect types of 3-person interacting teams using CBR vs. those using PBR.

Based on the propositions P4, P8 and P10 from the behaviourally motivated inspection theory [Sauer et al. 2000], the second goal of the experiment was stated:

G2. **Analyze** the performance of interacting and nominal inspection teams during

UML diagram inspections **for the purpose of** testing propositions of a theory of group software inspections **with respect to** inspection meeting synergy, number of false positives, and defect detection effectiveness **from the point of view of** the researcher.

The following hypotheses have been stated based on goal G2:

H2.1. Interacting teams improve group performance by discovering new defects.

H2.2. The number of false positives reported by the interacting team is smaller than that reported by the nominal team.

H2.3. Nominal teams outperform interacting teams in defect detection effectiveness.

Table 5.1 Relationships between goals and experimental hypotheses

Goal	Experimental Hypothesis
G1. Investigate the performance of interacting teams (IT): CBR vs. PBR	H1.1. CBR and PBR differ in false positives H1.2. CBR and PBR differ in overlaps H1.3. CBR and PBR differ in effectiveness detecting different defect types
G2. Test propositions from theory proposed by Sauer et al. [Sauer et al. 2000]	H2.1. IT do not improve group performance by discovering new defects H2.2. IT finds fewer false positives than nominal team (NT) H2.3. NT outperform IT in defect detection effectiveness

5.3 Variables

The experiment manipulates two independent variables – the reading technique (CBR and PBR), and the inspection team type (interacting team and nominal team). Each subject was exposed to two treatments: individual reading, followed by an interacting team meeting. The data of nominal teams was generated from individual reading scores. The following dependent variables were measured for interacting teams:

(1) *Number of true defects*, extracted from the team defect report forms.

(2) *Defect detection effectiveness*.

(3) *Number of false positives*, extracted from the team defect report forms.

(4) *Number of defect overlaps* (overlapping defects from individual reading), extracted from individual and team defect report forms.

(5) *Meeting gains* (number of true defects first detected during team meeting), extracted from individual and team defect report forms.

(6) *Meeting losses* (number defects first detected during individual reading, but not reported by a team), extracted from individual and team defect report forms.

The dependent variables of nominal teams are: defect detection effectiveness and number of false positives. They were extracted from individual report forms, counting the combined lists of unique defects and unique false positives (i.e. with duplicates removed). We used the same nominal team membership as that of interacting teams. This allows direct comparison between nominal and interacting teams.

5.4 Data Analysis

5.4.1 Collected Data

The data of 54 individuals and 18 inspection teams was collected during experiment. The data consisted of true defects and false positives. In those cases when duplicated defects or duplicated false positives were reported within the same defect form, we counted multiple identifications only once. Nine teams used CBR during individual reading, another nine – PBR. The data of interacting teams was extracted from team report forms, while the data of nominal teams was extracted from individual report forms. The same membership was used for both interacting and nominal teams.

Table 5.2 and Table 5.3 summarize the defect detection performance of teams that inspected Seminar and Hospital systems respectively. The following notation is used in these tables: “1” – defect detected during individual reading and included into team defect report form; “-1” – defect detected during individual reading but not included into team defect report form; “+1” – defect first detected during team meeting; “1” – overlap, i.e. the defect detected during individual reading by more than one team member; “IT” – no. of defects detected by interacting team; “NT” – no. of defects detected by nominal team; “OV” – no. of defect overlaps.

Table 5.4 summarizes the means and standard deviations for all the dependent variables.

Table 5.2 Defect detection in Seminar system diagrams

Team No.	Defects													IT	NT	OV
	1	2	3	4	5	6	7	8	9	10	11	12	13			
CBR1	1	1	1	1	1		-1		-1		-1	-1	1	6	10	5
CBR2		-1	1	1	1		1		1	-1		-1	1	6	9	6
CBR3	1	1	1	1	1	-1			1			1		7	8	6
CBR4	1	1	1	1	1	-1	-1	-1	1	1			1	8	11	8
CBR5		1	1	1	1		1		-1			1	1	7	8	3
PBR1	-1	1	1	1			-1	1				1	1	6	8	3
PBR2		1	1	1	-1	-1		1	1			1	-1	6	9	4
PBR3	1		1	1	1		1	1	1					7	7	3
PBR4			1	1			1	-1	1	-1		1	1	6	8	2

Table 5.3 Defect detection in Hospital system diagrams

Team No.	Defects														IT	NT	OV
	1	2	3	4	5	6	7	8	9	10	11	12	13	14			
CBR6		1	1	1	1	1		1		1	-1		1	1	9	10	6
CBR7	1	-1	1	1	1	-1		-1	1	1	-1		+1		7	10	6
CBR8		-1	-1	1	1	-1		1	1	1	1	1	1	1	9	12	6
CBR9		1	1	1	1	1		1	1	-1	1	1	1		10	11	6
PBR5	1	1		1	1				-1		1	1	1		7	8	2
PBR6	1	1		-1	1	1					+1	1	1	1	8	8	2
PBR7	-1	1		1	1	1	-1		1	1	1		1		8	10	2
PBR8	1	1	1	1	1	1		1	1		1		1		10	10	3
PBR9	1	1	1	1	1	1	+1	+1		-1	1	+1	1		11	9	2

Table 5.4 Means and standard deviations of the dependent variables

	Seminar system				Hospital system			
	CBR		PBR		CBR		PBR	
	Mean	Std. d.	Mean	Std. d.	Mean	Std. d.	Mean	Std. d.
Interacting team (IT)								
No. of true defects	6.8	0.84	6.25	0.5	8.75	1.26	8.8	1.64
Effectiveness (%)	52.31	6.44	48.08	3.85	62.5	8.99	62.86	11.74
No. of false positives	3.8	2.17	3.25	2.06	6.5	1.29	4	1.73
No. of overlaps	5.6	1.82	3	0.82	6	0	2.2	0.45
Meeting gains	0	0	0	0	0.25	0.5	0.8	1.3
Meeting losses	2.4	1.34	1.75	1.26	2.25	1.5	1.2	0.84
Nominal team (NT)								
No. of false positives	14.2	3.11	9	1.41	11.75	4.03	8.4	1.82
Effectiveness (%)	70.77	10.03	61.54	6.28	76.79	6.84	64.29	7.14

5.4.2 Results of Comparison between CBR and PBR Interacting Teams

We compared number of false positives, number of overlaps, and the effectiveness detecting different defect types of interacting teams using CBR vs. those using PBR. Table 5.5 summarizes the results of statistical testing of hypotheses H1.1-H1.3. We used independent samples t-test [Norŭsis 1995; Wohlin et al. 2000] with significance level 95% ($p<0.05$) to test these hypotheses.

Table 5.5 Results of hypotheses H1.1-H1.3 testing

Hypothesis	p-values	
	Seminar system	Hospital system
H1.1. CBR IT and PBR IT differ in false positives	0.709 (2-tailed)	0.042 (2-tailed)*
H1.2. CBR IT and PBR IT differ in overlaps	0.029 (2-tailed)*	0.00005 (2-tailed)*
H1.3. CBR IT and PBR IT differ in effectiveness detecting different defect types:		
Syntactic defects	0.778 (2-tailed)	0.096 (2-tailed)
Semantic defects	0.776 (2-tailed)	0.446 (2-tailed)
Consistency defects	0.874 (2-tailed)	0.25 (2-tailed)

* indicates significant results, i.e. $p<0.05$

The results of hypotheses testing are as follows:

- Hypothesis H1.1 asserts that the number of false positives reported by CBR interacting teams is not the same as that of PBR teams. This hypothesis is supported for the teams that inspected Hospital system: PBR teams (mean 4) find fewer false positives than CBR teams (mean 6.5) with significance level 0.042. However, hypothesis is not supported for teams that inspected Seminar system.
- Hypothesis H1.2 asserts that the number of defect overlaps in CBR interacting teams is not the same as that of PBR teams. This hypothesis is supported, i.e. CBR teams report significantly more overlaps than PBR teams. For teams that inspected Seminar system, CBR teams (mean 5.6) report on average 46% more overlaps than PBR teams (mean 3) with significance level 0.029. For teams that inspected Hospital system, CBR teams (mean 6) report on average 63% more overlaps than PBR teams (mean 2.2) with

significance level 0.00005.

- Hypothesis H1.3 asserts that CBR and PBR interacting teams differ in effectiveness detecting different defect types. This hypothesis is not supported, i.e. hypothesis testing did not reveal significant difference between CBR and PBR teams in detecting syntactic, semantic and consistency defects.

The results revealed that PBR teams report significantly less overlaps than CBR teams. Testing the number of false positives between CBR and PBR teams did not provide a conclusive answer: PBR teams reported fewer false positives while inspecting Hospital system; however they report similar number of false positives as CBR teams while inspecting Seminar system. In addition, hypotheses testing did not reveal any difference between CBR and PBR teams in detecting different types of defects.

5.4.3 Results of Inspection Theory Proposition Testing

The results of Hypotheses H2.1-H2.3 testing are summarized in the Table 5.6. We used paired t-test [Norŭsis 1995; Wohlin et al. 2000] with significance level 95% ($p<0.05$) to test these hypotheses.

Table 5.6 Results of hypotheses H2.1-H2.3 testing

Hypothesis	p-values	
	Seminar system	Hospital system
H2.1. IT improve group performance by discovering new defects	CBR: --- (1-tailed) PBR: --- (1-tailed)	CBR: 0.196 (1-tailed) PBR: 0.121 (1-tailed)
H2.2. IT find fewer false positives than NT	CBR: 0.0003 (1-tailed)* PBR: 0.011 (1-tailed)*	CBR: 0.023 (1-tailed)* PBR: 0.004 (1-tailed)*
H2.3. NT outperform IT in defect detection effectiveness	CBR: 0.008 (1-tailed)* PBR: 0.034 (1-tailed)*	CBR: 0.020 (1-tailed)* PBR: 0.389 (1-tailed)

* indicates significant results, i.e. $p<0.05$

The results of hypotheses testing are listed as follows:

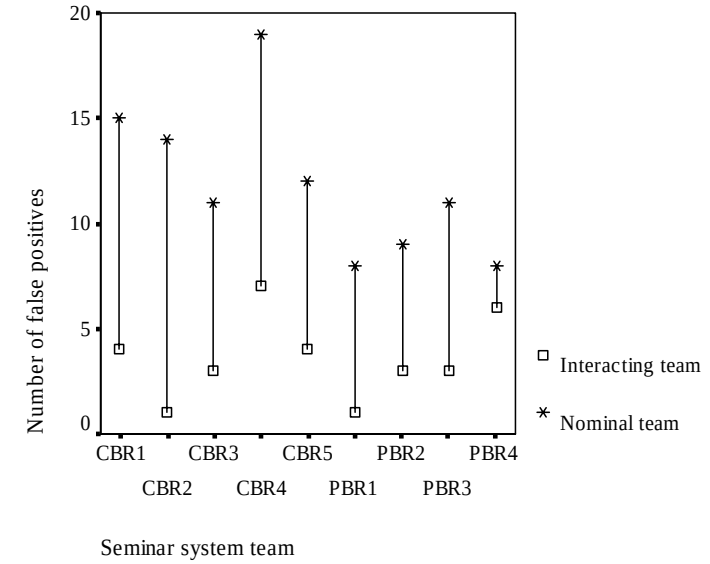
- Hypothesis H2.1 asserts that interacting teams improve group performance by discovering new defects. To test this hypothesis we evaluated if the number of new defects detected during inspection meeting (meeting gains) is

different from 0. The hypothesis is not supported, i.e. inspection teams do not detect significant number of new defects during meeting. CBR and PBR teams that inspected Seminar system did not find any new defects during inspection meeting; CBR teams that inspected Hospital system found on average 0.25 new defects during meeting (observed significance level 0.391), PBR teams – 0.8 new defects (observed significance level 0.242). In addition, we evaluated if meeting gains are different from meeting losses. The results revealed, that for CBR teams meeting losses are significantly greater than meeting gains (for CBR teams that inspected Seminar system p=0.041; for CBR teams that inspected Hospital system p=0.016). However, there is no statistically significant difference between meeting gains and meeting losses of PBR teams.

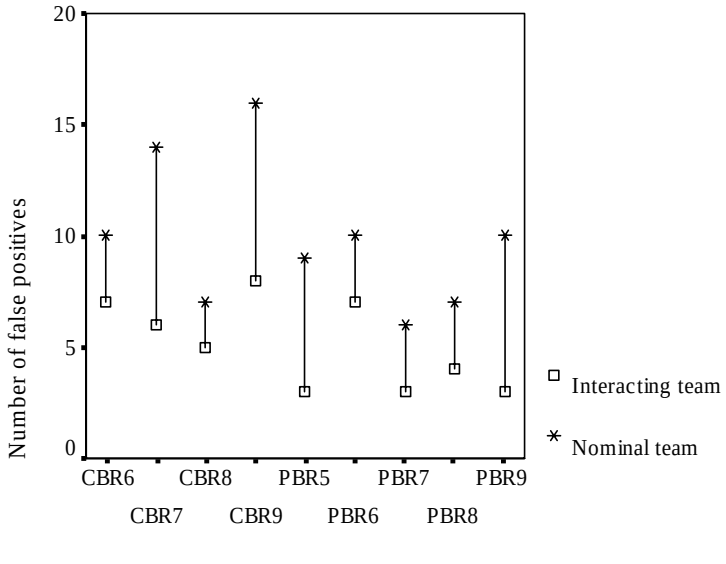
- Hypothesis H2.2 asserts that the number of false positives reported by the interacting team is smaller than that reported by the nominal team. This hypothesis is supported. From the teams that inspected Seminar system, CBR interacting teams (mean 3.8) reported on average 73% less false positives than nominal teams (mean 14.2) with significance level 0.0003; PBR interacting teams (mean 3.3) reported on average 64% less false positives than nominal teams (mean 9) with significance level 0.011. From the teams that inspected Hospital system, CBR interacting teams (mean 6.5) reported on average 45% less false positives than nominal teams (mean 11.8) with significance level 0.023; PBR interacting teams (mean 4) reported on average 52% less false positives than nominal teams (mean 8.4) with significance level 0.004. The number of false positives detected by each interacting and nominal team is depicted in Figure 5.1 (the vertical line segment indicates the difference between interacting and nominal team).

- Hypothesis H2.3 asserts that nominal teams outperform interacting teams in defect detection effectiveness. This hypothesis is supported for all CBR teams, and PBR teams that inspected Seminar system. However, it is not supported for PBR teams that inspected Hospital system. From the teams that inspected Seminar system, CBR nominal teams (mean 70.8) significantly outperform interacting teams (mean 52.3) by an average of 26% (observed significance level 0.008); PBR nominal teams (mean 61.5) significantly outperform interacting teams (mean 48.1) by an average of 22% (observed significance level 0.034). From the teams that inspected Hospital systems, CBR nominal teams (mean 76.8) significantly outperform interacting teams

(mean 62.5) by an average of 19% (observed significance level 0.020); however, PBR nominal teams do not outperform the interacting ones. The defect detection effectiveness for each interacting and nominal team is depicted in Figure 5.2 (the vertical line segment indicates the difference between interacting and nominal team).

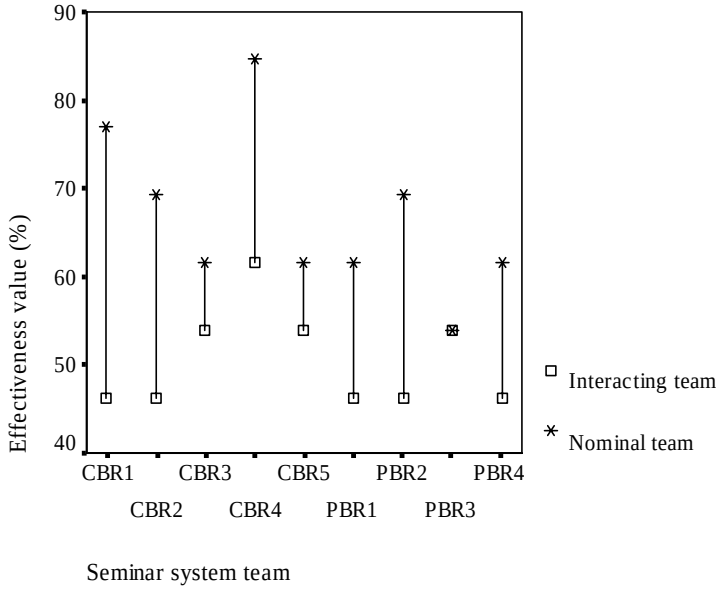


(a)

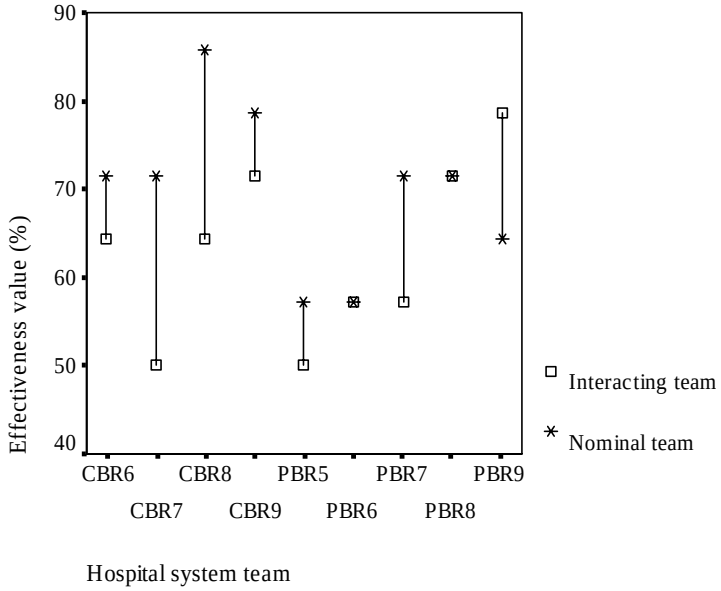


(b)

Figure 5.1 Number of false positives for: (a) Seminar system; (b) Hospital system.



(a)



(b)

Figure 5.2 Defect detection effectiveness for: (a) Seminar system; (b) Hospital system.

Therefore, the results of hypotheses H2.1 and H2.2 testing support the propositions P4 and P8 of the inspection theory. The proposition P10 is partially supported by hypothesis H2.3 testing: CBR and PBR nominal teams that inspected Seminar system, and CBR nominal teams that inspected Hospital system outperform interacting teams, but PBR nominal teams that inspected Hospital system do not outperform interacting teams. However, if the data of Seminar and Hospital systems is analyzed as a whole, nominal teams outperform

the interacting ones.

5.5 Interpretation of Results

In an initial analytical step we compared interacting inspection teams using CBR vs. those using PBR. The results showed that:

- PBR teams report fewer defect overlaps than CBR teams;
- There was no significant difference in effectiveness between CBR and PBR detecting syntactic, semantic and consistency defects;
- PBR teams reported fewer false positives than CBR teams, however this result was significant only for one (out of two) systems inspected.

The data analysis did not reveal significant difference in effectiveness detecting different defect types between CBR and PBR. In addition, the ability of PBR and CBR to discriminate between true defects and false positives was significant only for one inspected system. Consequently, it is required a future investigation of these variables.

In a second stage we tested the following propositions from the theory of software inspection:

P4. The interacting group meeting does not improve group performance over nominal group by discovering new defects. This proposition was fully supported. The analysis of the data showed that interacting groups do not detect a significant number of new defects. Moreover, meeting gains were significantly smaller than meeting losses for CBR teams; and meeting gains were offset by meeting losses for PBR teams;

P8. The performance advantage of an interacting group over a nominal group is a function of the level of false positives discovered by individuals. This proposition was fully supported. Interacting teams reported fewer false positives than nominal teams. Therefore, we can conclude, that the performance advantage of interacting teams is the ability to discriminate between true defects and false positives identified by individual inspectors;

P10. Nominal groups outperform interacting groups at the discovery task. This proposition was partially supported: nominal teams outperform interacting teams in defect detection effectiveness for CBR teams, and for PBR teams of one (out of two) inspected systems. However, if the data of Seminar and

Hospital systems is analyzed as a whole, nominal teams outperform the interacting ones.

5.6 Conclusions

This chapter presented a further investigation on inspection meetings. The main goals of the chapter were twofold: to perform more detail analysis of team performance as compared to Chapter 4, and to test several propositions from the behaviourally motivated inspection theory [Sauer et al. 2000].

The results of the data analysis revealed that PBR teams report fewer defect overlaps, and in some cases fewer false positives than CBR teams. Furthermore, the results supported all three propositions of the theory that have been tested: 1) interacting teams did not detect a significant number of new defects during inspection meeting, however, 2) they reported fewer false positives than nominal teams, and 3) nominal teams outperform interacting teams in defect detection effectiveness. The theory has to be further tested in order to verify all aspects of individual and group behaviour during inspection more completely.

CHAPTER 6

Extended Metrics to Evaluate Cost-Effectiveness of Software Inspections

6.1 Introduction

In order to evaluate the effectiveness of inspections with respect to software development cost, several metrics have been previously proposed [Collofello & Woodfield 1989; Fagan 1976; Kusumoto et al. 1992]. Collofello & Woodfield have taken into account all the costs consumed and saved by inspections and proposed a metric, called *Cost Effectiveness* [Collofello & Woodfield 1989]. Kusumoto et al. proposed a metric M_k for evaluating the cost-effectiveness of inspections, based on the degree to which costs to detect all faults from the software in a project are reduced by inspections [Kusumoto et al. 1992]. However, none of those metrics considers false positives introduced during inspection, although the rework of false positives is costly and can introduce new defects [Land et al. 1997b; Sauer et al. 2000]. Consequently, this chapter introduces:

a) An inspection cost model that describes all costs related to inspections;

b) Four new metrics to evaluate the cost-effectiveness as well as the losses in the preparation and inspection meeting stages.

6.2 Inspection Process and Lifecycles of a Defect and a False Positive

Software inspection as a structured process was first described by Fagan (1976). It consists of the following stages: planning, overview, preparation, inspection meeting, rework and follow-up (see Section 2.1.1).

We introduce two diagrams (Figure 6.1 and Figure 6.2) to explain the life-cycles of defects and false positives respectively. These diagrams show in what stages defects and false positives are being introduced, detected and removed:

- The life-cycle of a defect is depicted in Figure 6.1. It shows four cases of

defect's life-cycle: d1, d2, d3 and d4. In all the cases, defects are introduced before inspection process begins, for instance during designing or coding. According to Fagan's model [Fagan 1976], defects are detected and confirmed during inspection meeting stage, and removed by author during rework (case d1). However, defects are usually detected by individual reviewers during preparation, confirmed by inspection teams during inspection meeting, and removed by author during rework (case d2). In some cases, defects are detected during preparation, however not confirmed as defects during inspection meeting (case d3). Some defects are not detected during inspection at all (case d4). In cases d3 and d4, defects are detected and removed only during testing;

- The life-cycle of a false positive is shown in Figure 6.2. It identifies five cases of false positive's life-cycle: f1, f2, f3, f4 and f5. False positives can be introduced during preparation or inspection meeting stages. In cases when false positives are introduced during preparation stage, they can be detected and excluded from defect list by inspection team during inspection meeting (case f1) or by author during rework (case f2). False positives introduced during inspection meeting can be detected and excluded from defect list by author during rework (case f4). However, if false positive is not excluded from the defect list, the rework will be done and consequently a defect may be introduced, which will be detected and removed only during testing (cases f3 and f5).

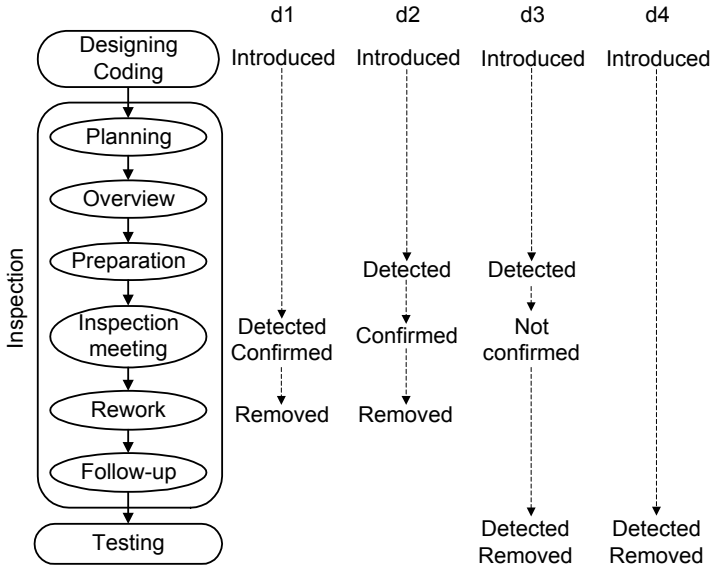


Figure 6.1 Life-cycle of a defect.

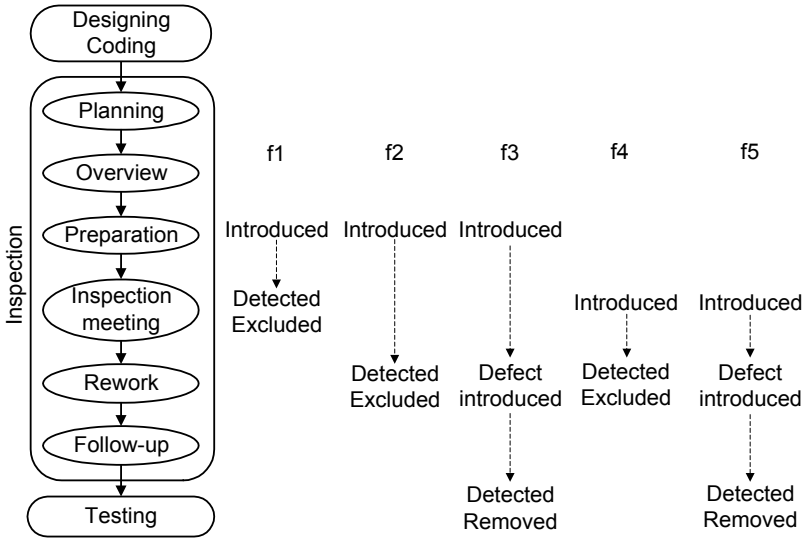


Figure 6.2 Life-cycle of a false positive.

The goal of the inspection is to ensure that the minimum number of defects and false positives reaches testing.

6.3 Inspection Cost Model

6.3.1 Traditional Cost Model

The traditional inspection cost model consists of the following components [Kusumoto et al. 1992] (Figure 6.3):

- C_r – cost spent for inspection;
- C_t – cost needed for testing;
- ΔC_t – testing cost saved by inspection;
- *Virtual testing cost* – testing cost if no inspections are executed. By spending cost C_r during inspection, the cost ΔC_t is being saved during testing.

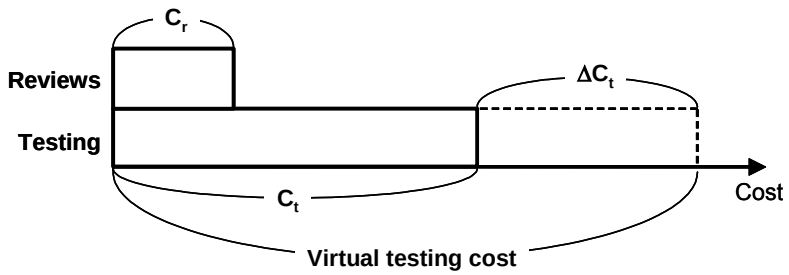


Figure 6.3 Traditional cost model.

6.3.2 Extended Cost Model for Preparation Stage of Inspection

In order to evaluate the influence of false positives introduced during preparation stage of inspection over the testing cost, we extend the traditional cost model. To do so, the following additional costs are defined (Figure 6.4):

- C_{rDEF} – cost spent to detect actual defects during preparation;
- C_{rFP} – cost spent to detect false positives during preparation;
- C_{tDEF} – cost needed for testing to detect remaining defects;
- C_{tFP} – cost needed for testing to detect defects introduced by false positives during preparation.

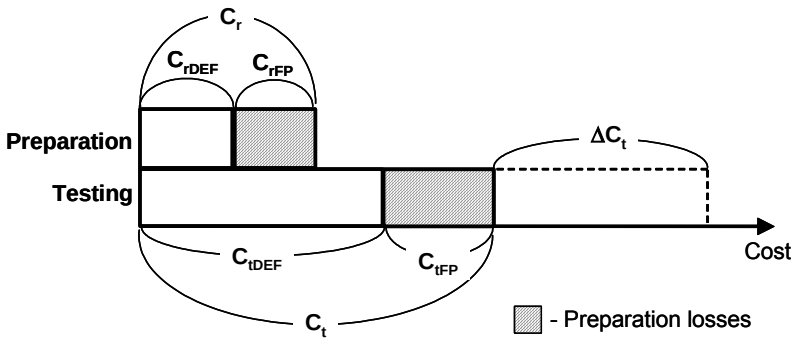


Figure 6.4 Extended cost model for preparation stage.

In this model, by spending cost C_{rDEF} during inspection, the testing cost ΔC_t is being saved. However, by spending the cost C_{rFP} during inspection, the cost C_{tFP} is being added to the testing cost. Therefore, the costs C_{rFP} and C_{tFP} represent

the losses of the preparation stage (Figure 6.4).

6.3.3 Extended Cost Model for Preparation and Inspection Meeting Stages of Inspection

Inspection meetings are usually carried out after the preparation stage is completed. However, several authors question the usefulness of such meetings [Johnson & Tjahjono 1998; Porter & Johnson 1997; Votta 1993]. To enable the evaluation of the benefits and the losses of inspection meetings, we propose an extended cost model for the preparation and inspection meeting stages that includes the following costs (Figure 6.5):

- C_{mDEF} – cost spent to confirm actual defects detected during preparation stage;
- C_{mFP} – cost spent to confirm false positives detected during preparation stage;
- C_{mADD_DEF} – cost spent to detect additional defects during inspection meeting stage;
- C_{mADD_FP} – cost spent to detect additional false positives during inspection meeting stage;
- C_{mLOST_DEF} – cost spent to eliminate actual defects detected during preparation stage;
- C_{mELIM_FP} – cost spent to eliminate false positives detected during preparation stage;
- C_{tADD_FP} – cost needed for testing to detect defects introduced by additional false positives detected during inspection meeting stage;
- C_{tLOST_DEF} – testing cost needed to detect defects lost during inspection meeting stage;
- ΔC_{tADD_DEF} – testing cost saved by additional defects detected during inspection meeting stage;
- ΔC_{tELIM_FP} – testing cost saved by false positives eliminated during inspection meeting stage;

- ΔC_{tDEF} – testing cost saved by defects detected during preparation and confirmed during inspection meeting stages.

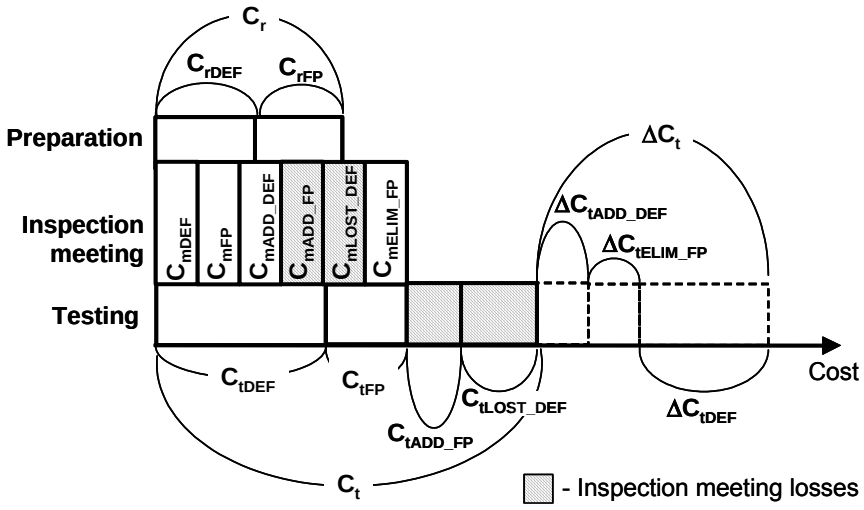


Figure 6.5 Extended cost model for preparation and inspection meeting stages.

In this model, cost C_{mELIM_FP} spent to eliminate false positives, reduces the testing cost by ΔC_{tELIM_FP}; and cost C_{mLOST_DEF}, spent to eliminate actual defects, and increases testing cost by C_{tLOST_DEF}. Inspection meeting cost spent to detect additional defects, C_{mADD_DEF}, reduces the testing cost by ΔC_{tADD_DEF}; and inspection meeting cost spent to detect additional false positives, C_{mADD_FP}, increases the testing cost by C_{tADD_FP}. In other words, the costs C_{mLOST_DEF} , C_{mADD_FP}, C_{tLOST_DEF} and C_{tADD_FP} are the costs lost by inspection meeting (Figure 6.5).

Inspection meeting costs C_{mDEF} and C_{mFP}, spent to confirm defects and false positives detected during preparation stage, do not increase or reduce testing costs; however, they increase the overall inspection cost.

6.4 Metrics to Evaluate Software Inspections

6.4.1 Fagan's Metric

Fagan [Fagan 1976] introduced the *Error Detection Efficiency* metric M_f for measuring inspection efficiency. M_f is defined as the number of defects found during inspection over the total number of defects in the product existing before inspection. We define the total number of defects in the product existing before inspection as DEF_{total}, and the number of defects found during inspection as DEF_r.

Then we get the following equation:

$$M_f = \frac{DEF_r}{DEF_{total}} \tag{6.1}$$

As we can see from equation (6.1), metric M_f does not account for the cost expended for inspection.

6.4.2 Collofello's Metric

Collofello and Woodfied (1989) proposed *Cost Effectiveness* metric M_c, which is defined as a ratio of the "cost saved by the process" to the "cost consumed by the process". Using the notation described in Section 6.3.1, we get the following equation:

$$M_c = \frac{\Delta C_t}{C_r} \tag{6.2}$$

Although metric M_c takes into account the costs consumed and saved by inspections, it does not take into account the total cost to detect all defects in the software product by inspection and testing.

6.4.3 Kusumoto's Metric

Kusumoto et al. (1992) proposed a metric M_k to evaluate the cost effectiveness of software inspections in terms of reduction of cost to detect and remove all defects from software. Using the notation described in Section 6.3.1, we get the following equation:

$$M_k = \frac{\Delta C_t - C_r}{C_t + \Delta C_t} \tag{6.3}$$

M_k is a ratio of the reduction of the total costs to detect and remove all defects from documents using inspections in a project to the virtual testing cost. The testing cost is reduced by ΔC_t compared to the virtual testing cost ($C_t + \Delta C_t$) if no inspection is executed (see Figure 6.3).

6.5 Extended metrics

6.5.1 Need for New Metrics

Among metrics M_f, M_c and M_k, metric M_k is the most practical one to evaluate the cost effectiveness of inspections. However, it includes only the total cost spent on inspection, not taking into consideration the composition of the inspection costs described in Section 6.3.

We decided to extend metric M_k to conform to the extended cost model. Section 6.5.2 presents metrics for the extended cost model for preparation stage, and Section 6.5.3 presents metrics for the extended cost model of preparation and inspection meeting stages.

6.5.2 Extension of Metric M_k for Preparation Stage

In accordance with the extended cost model for preparation stage (see Section 6.3.2), cost ΔC_t is a testing cost saved by preparation, and costs C_{rFP} and C_{tFP} are the costs lost by inspections, since additional effort is being spent during inspection for the detection of false positives, and those false positives cause additional cost during testing (Figure 6.4). We introduce a new metric M_{l_IDV} to evaluate the *Preparation Losses*, which is the ratio of "cost lost by inspections" by "cost saved by inspections":

$$M_{l\_IDV} = \frac{C_{rFP} + C_{tFP}}{\Delta C_t} \tag{6.4}$$

Not only the costs C_{rFP} and C_{rDEF}, but also C_{tFP} is the cost caused by inspection, since additional effort is being spent during testing to remove the defects introduced by false positives during preparation stage. Therefore, we introduce a new metric M_{g_IDV}, which is an extension of metric M_k (see equation 6.3), to evaluate the *Extended Cost Effectiveness of Preparation Stage of Inspection*. It can be expressed using the following formula:

$$M_{g\_IDV} = \frac{\Delta C_t - C_r - C_{tFP}}{C_{tDEF} + \Delta C_t} = \frac{\Delta C_t - C_{rDEF} - C_{rFP} - C_{tFP}}{C_{tDEF} + \Delta C_t} \tag{6.5}$$

In metric M_k (see Section 6.4.3), the virtual testing cost (the testing cost if no inspection is executed) is defined as (C_t + ΔC_t). However, inspection increases

the testing cost by C_{tFP} if false positives have been detected (see Figure 6.4). Therefore, if no inspection is executed, the cost C_{tFP} should not be included into the virtual testing cost. In equation (6.5), we exclude this cost from the virtual testing cost, and define virtual testing cost as ($C_{tDEF} + \Delta C_t$). In addition, since the testing cost C_{tFP} is the cost caused by inspection as well, we add it to inspection costs in equation (6.5).

In case if no false positives have been introduced during preparation stage $M_{g\_IDV} = M_k$, otherwise $M_{g\_IDV} < M_k$.

6.5.3 Extension of Metric M_k for Preparation and Inspection Meeting Stages

According to the extended cost model for preparation and inspection meeting stages (see Section 6.3.3, Figure 6.5), two additional testing costs are introduced by inspection meeting: C_{tLOST_DEF} and C_{tADD_FP}. C_{tLOST_DEF} is a testing cost spent to detect defects lost during inspection meeting, and C_{tADD_FP} is a cost spent to eliminate additional false positives, introduced during inspection meeting. Inspection meeting may save testing cost by finding additional defects detected during inspection meeting stage ΔC_{tADD_DEF}, and by eliminating false positives detected during preparation stage ΔC_{tELIM_FP}.

Inspection meeting costs C_{mDEF} and C_{mFP}, spent to confirm actual defects and false positive defects detected during preparation stage, do not have influence over testing cost, however they increase the overall inspection cost.

Costs C_{mADD_DEF} and C_{mELIM_FP}, spent to detect additional defects and to eliminate false positives, reduce testing costs by ΔC_{tADD_DEF} and ΔC_{tELIM_FP} respectively. Costs C_{mADD_FP} and C_{mLOST_DEF} spent to detect additional false positives and to eliminate actual defects, increase testing costs by C_{tLOST_DEF} and C_{tADD_FP} respectively. The purpose of the inspection meeting should be to minimise the costs C_{mADD_FP} and C_{mLOST_DEF}, and to maximise the costs C_{mADD_DEF} and C_{mELIM_FP}.

Similarly as the metric M_{l_IDV} (see equation 6.4) to evaluate the losses of preparation stage, we introduce a new metric M_{l_MEET} to evaluate the *Inspection Meeting Losses*, which is the ratio of "cost lost by inspection meeting" by "cost saved by inspection meeting" (see Figure 6.5):

$$M_{l\_MEET} = \frac{C_{mADD\_FP} + C_{mLOST\_DEF} + C_{tADD\_FP} + C_{tLOST\_DEF}}{\Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP}} \quad (6.6)$$

Although the costs C_{mFP} and C_{tFP} are the additional costs caused by inspection, we do not include them into the metric M_{l_MEET} (equation 6.6), because they depend on both preparation and inspection meeting stages.

Similarly as the metric M_{g_IDV} (see equation 6.5), we propose a metric *Extended Cost Effectiveness of Preparation and Inspection Meeting Stages*, M_{g_MEET}, to evaluate cost effectiveness of software inspections, when both preparation and inspection meeting are performed:

$$M_{g\_MEET} = \frac{\Delta C_t - C_r - C_m - C_{tFP} - C_{tADD\_FP} - C_{tLOST\_DEF}}{C_{tDEF} + \Delta C_t} =$$

$$= \frac{\Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP} + \Delta C_{tDEF} - C_{rDEF} - C_{rFP}}{C_{tDEF} + \Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP} + \Delta C_{tDEF}}$$

$$- \frac{C_{mDEF} + C_{mFP} + C_{mADD\_DEF} + C_{mADD\_FP} + C_{mLOST\_DEF}}{C_{tDEF} + \Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP} + \Delta C_{tDEF}}$$

$$- \frac{C_{mELIM\_FP} + C_{tFP} + C_{tADD\_FP} + C_{tLOST\_DEF}}{C_{tDEF} + \Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP} + \Delta C_{tDEF}} \qquad (6.7)$$

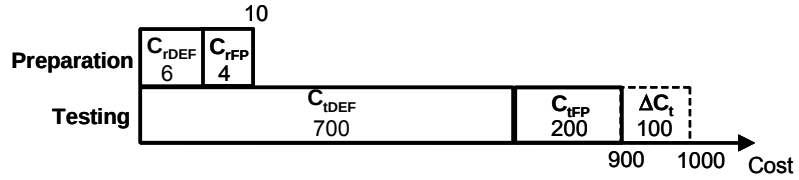
Metric M_{g_MEET} (equation 6.7) is a modification of metric M_k (equation 6.3). In equation (6.7), the virtual testing cost is defined as ($C_{tDEF} + \Delta C_{tADD\_DEF} + \Delta C_{tELIM\_FP} + \Delta C_{tDEF}$), and the testing costs C_{tFP}, C_{tADD_FP} and C_{tLOST_DEF}, caused by inspection, are added to the inspection cost.

6.6 Comparison of Proposed Metrics to Metrics M_c and M_k

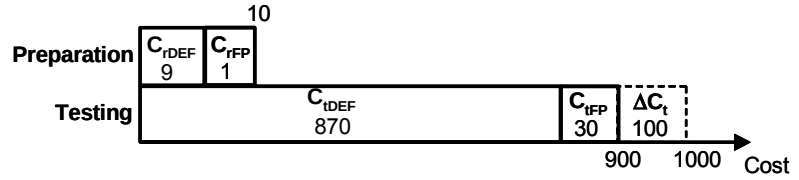
The differences among M_c, M_k and proposed metrics can be demonstrated with reference to five imaginary projects (Figure 6.6). In all projects, if no inspections had been executed, the testing cost would be 1000 units.

In cases Case I – Case III of Figure 6.6, the preparation stage of inspection has been performed. In cases Case IV and Case V of Figure 6.6, preparation and inspection meeting stages have been performed. The notation of Figure 6.6 is taken from the extended cost models (Figure 6.4 and Figure 6.5).

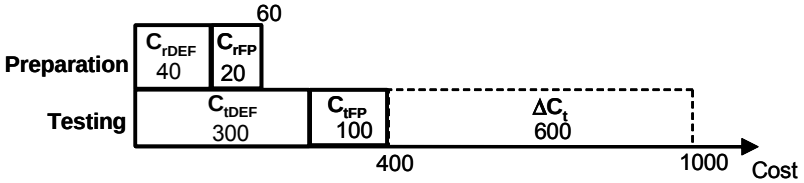
In Case I of Figure 6.6 (a), inspection consumes 10 units of cost (6 units to detect actual defects, and 4 units to detect false positives), saves 100 units of testing cost, and the testing cost is 900 (700 to detect remaining defects in software product, and 200 to detect defects introduced by false positives). Therefore, the total cost is 910.



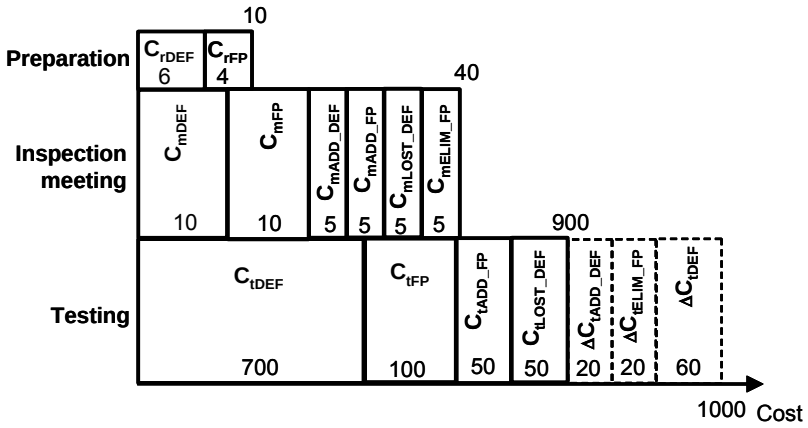
(a) Case I (Testing cost reduced by 100)



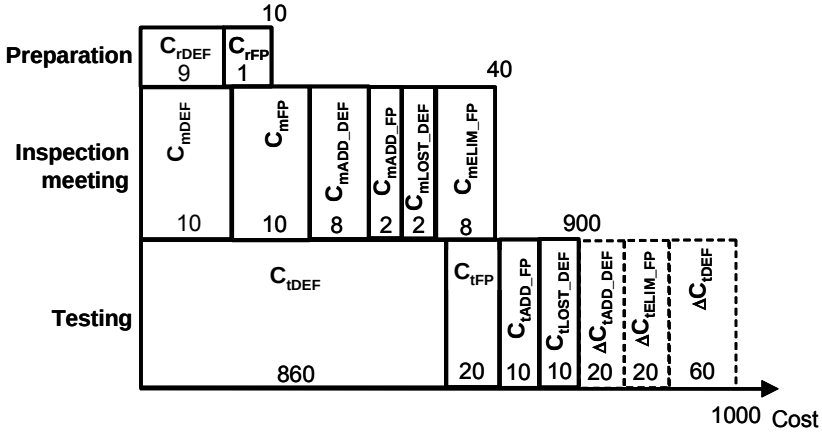
(b) Case II (Testing cost reduced by 100)



(c) Case III (Testing cost reduced by 600)



(d) Case IV (Testing cost reduced by 100)



(e) Case V (Testing cost reduced by 100)

Figure 6.6 Comparing five different cases of inspection.

Case II is similar to Case I, because it consumes the same costs on inspection and testing as in Case I, however the distribution of inspection and testing costs is different in Case II (during inspection, 9 units of cost are spent to detect actual defects and 1 unit to detect false positives; during testing, 870 units of cost are spent to detect remaining defects in software product and 30 to detect defects introduced by false positives).

In Case III of Figure 6.6 (c), inspection costs 60 units (40 units to detect actual defects, and 20 units to detect false positives), saves 600, testing cost is 400 (300 to detect remaining defects in software product, and 100 to detect defects introduced by false positives), and the total cost is 460.

If we apply Collofello's metric M_c to Cases I, II and III, the value of M_c is 10 in all cases. However, in Case III, inspection saved much more of the total defect detection cost than in Cases I and II, therefore Case III would be expected to be more cost effective.

The value of metric M_k for Case I and II is 0.09, and for Case III is 0.54. Thus, M_k indicates that the inspection in Case III is more effective than in Cases I and II. However, it does not show the difference between Cases I and II, although the inspection losses due to the false positives are greater in Case I. If we apply the metric M_{I_IDV} to evaluate inspection losses, the value of M_{I_IDV} is 2.04 in Case I, 0.31 in Case II, and 0.2 in Case III. Thus, M_{I_IDV} indicates that inspection losses are the greatest in Case I.

The values of the extended cost effectiveness metric M_{g_IDV}, which takes into consideration inspection losses, are -0.14 in Case I, 0.06 in Case II, and 0.49 in Case III. As we can see from those results, the metric M_{g_IDV} is more precise than M_k, because it shows that inspection in Case II is more effective than in Case I.

Case IV and Case V of Figure 6.6 demonstrate the projects in which preparation and inspection meeting stages are performed. In both cases, preparation consumes 10 units of cost, inspection meeting consumes 40 units of cost, the testing cost is 900, and 100 units of cost are saved by inspection. However, these cases differ in the distribution of costs. During inspection meeting, in Case IV 5 units of cost are spent to detect additional defects (Case V: 8 units), 5 units to detect additional false positives (Case V: 2 units), 5 units to eliminate actual defects (Case V: 2 units), and 5 units to eliminate false positives (Case V: 8 units). During testing, in Case IV 700 units of cost are spent to detect defects (Case V: 860), 100 units to detect defects introduced by false positives (Case V: 20), 50 units to detect defects introduced by additional false positives detected during inspection meeting (Case V: 10), and 50 units to detect defects

lost during inspection meeting (Case V: 10). The total cost in both cases is 950.

The value of metric M_c in both cases Case IV and Case V is 2, and the value of metric M_k in both cases is 0.05. The value of metric M_{l_MEET} in Case IV is 2.75, and in Case V is 0.6. Therefore, M_{l_MEET} indicates that inspection losses are greater in Case IV as compared to Case V. The value of metric M_{g_MEET} is -0.188 in Case IV, and 0.0104 in Case V. Consequently, metric M_{g_MEET} is more precise than M_c and M_k since it shows that inspection is more effective in Case V as compared to Case IV.

6.7 Summary and Conclusions

In this chapter, we have proposed two extended cost models: a model to describe the costs spent during preparation stage, and a model to describe the costs spent during the preparation and inspection meeting stages. Those models can be useful for other researchers to have a greater understanding of all costs related to inspections.

In addition, we have proposed two new metrics M_{l_IDV} and M_{l_MEET} to evaluate the *Preparation Losses* and the *Inspection Meeting Losses* respectively. Also, we have proposed two metrics M_{g_IDV} and M_{g_MEET}, which are the modifications of Kusumoto's metric M_k, to evaluate the *Extended Cost Effectiveness of Preparation* and the *Extended Cost Effectiveness of Preparation and Inspection Meeting* respectively. All those metrics enable a more precise evaluation of software inspections as compared to the conventional metrics.

The following chapter presents an experimental evaluation of new metrics, which was made using the data collected from Experiment 2 (see Chapter 4).

CHAPTER 7

Experimental Evaluation of the New Metrics

7.1 Introduction

To demonstrate the validity of the extended metrics proposed in Chapter 6, we apply those metrics along with Kusumoto's metric M_k to the data collected from Experiment 2 (see Chapter 4), and compare the resultant values obtained by these metrics.

7.2 Evaluation Approach

This section describes the evaluation approach used in this research. Two types of evaluation have been made:

1) *Evaluation of Metrics for Preparation Stage of Inspection M_{l_IDV} and M_{g_IDV}.* The values of extended cost effectiveness metric for preparation stage M_{g_IDV} have been compared against the values of metric M_k. In addition, the values of preparation losses metric M_{l_IDV} have been calculated and analyzed;

2) *Evaluation of Metrics for Preparation and Inspection Meeting Stages M_{l_MEET} and M_{g_MEET}.* The values of extended cost effectiveness metric for preparation and inspection meeting stages M_{g_MEET} have been compared against the values of metric M_k. Furthermore, the values of inspection meeting losses metric M_{l_MEET} have been calculated and analyzed.

7.2.1 Inspection Costs

As it was mentioned in Section 2.2.1, there are three types of inspection costs: indirect, opportunity, and direct [Biffl et al. 2001]. In this analysis we define such costs in the following way:

- *Opportunity costs:* we assume that there are no opportunity costs, i.e. inspectors do their most valuable job while inspecting;
- *Indirect costs:* we assume that there are no indirect costs;
- *Direct costs:* the direct costs of preparation stage were calculated using the

time each inspector has spent on preparation; and the direct costs of inspection meeting stage were calculated using the time team members have spent on team meeting. The total cost of inspection was calculated as a sum of direct costs of preparation and inspection meeting stages.

7.2.2 Testing Costs

The data collected from Experiment 2 included inspection data, however it did not include testing data, since testing has not been performed. In order to calculate the values of the metrics, the testing data is needed as well.

The testing costs can be calculated from the set of defects found and assumptions on the benefit of finding a defect during inspection [Biffl & Gutjahr 2001].

Benefits of a Defect

The benefit of the defect comes from the estimated savings of rework, if a defect has to be detected and removed later in development or operation. It depends on the severity of the defect. In this work we distinguish two severity levels:

- Major defects;
- Minor defects.

There are several approaches to determine the benefit for a defect of a given severity class [Biffl et al. 2001]:

1) One of approaches is to assign each defect a single benefit value. Gilb and Graham [Gilb & Graham 1993] assume average savings of 8 hours for a major defect, and savings of 1 hour for a minor defect;

2) Another approach is to assume for each defect class a probability distribution of benefits. This can be, for example, a triangle distribution for the best, most likely, and the worst cases [Biffl et al. 2001];

3) A more sophisticated approach includes estimates on the benefit for several development stages, for example, the estimates for the early stages (design) and the later stages (testing or operation).

In this work we use the first approach. We assume that:

a) A major defect detected during inspection saves 8 hour of testing;

b) A minor defect detected during inspection saves 1 hour of testing.

Assumptions Regarding Defects

In this work we assumed that:

- All defects undetected during inspection propagate into testing;
- No additional defects are introduced into software products between inspection and testing;
- All defects that reach testing are detected and removed during testing.

Influence of False Positives over Testing Costs

To the best of our knowledge, there is a lack of research considering false positives, especially estimating cost/benefit of a false positive. Therefore, we defined five different cases of the influence of false positives on testing:

Case 0*: no false positives propagate into testing*. We assume, that all the false positives have been corrected before testing;

Case 1*: half part of false positives propagates into testing and each of them introduces a minor defect*. We assume that 50% of false positives have been corrected and did not reach testing, while another 50% of false positives have reached testing and introduced minor defects;

Case 2*: all false positives propagate into testing and each of them introduces a minor defect*. This case is more pessimistic than case 1, since we assume that all false positives propagate into testing;

Case 3*: half part of false positives propagates into testing and each of them introduces a major defect*. In this case we assume that 50% of false positives have been corrected and did not propagate into testing, while another 50% of false positives have reached testing and introduced major defects;

Case 4: *all false positives propagate into testing and each of them introduces a major defect*. This case is the most pessimistic one, since false positives have the maximum impact on testing.

Computing Testing Costs

Testing costs have been computed in the following way:

- To compute the testing cost spent to detect defects undetected during inspection, we multiplied the number of undetected major defects by 480 minutes (we assume that it takes 8 hours of testing to detect and remove a major defect) and the number of undetected minor defects by 60 minutes (we assume it takes 1 hours of testing to detect and remove a minor defect), and added the resultant values;

- Testing cost needed to detect defects introduced by false positives was computed depending on the number of false positives introduced, and the influence of false positives on testing (see section 7.2.2). For example, in Case 2 (all false positives propagate into testing and introduce minor defects) testing cost needed to detect defects introduced by false positives equals to the number of false positives multiplied by 60 minutes; and in Case 4 (all false positives propagate into testing and introduce major defects) testing cost needed to detect defects introduced by false positives equals to the number of false positives multiplied by 480 minutes;

- In order to compute testing cost saved by inspection, we multiplied the number of major defects detected during inspection by 480 minutes and the number of minor defects detected during inspection by 60 minutes, and added the resultant values;

The detail formulas to compute all testing costs are described in sections 7.3.1 and 7.4.1.

7.2.3 Computing Metrics Values

The metric values were computed for preparation stage (**I**), and for preparation and inspection meeting stages (**II**) of inspection process separately (see Table 7.1).

While computing metrics values, we wanted to investigate the influence of false positives over the metrics values. Therefore, we computed metrics values for each of five different cases (Case 0 – Case 4) of the influence of false positives on testing (see Section 7.2.2). In total, 10 different sets of metrics values have been computed as it is summarized in Table 7.1.

Table 7.1 Computing Metrics Values

No	Metrics	Influence of False Positives
I	Preparation stage's metrics *Mk, Mg_IDV, Ml_IDV*	Case 0 Case 1 Case 2 Case 3 Case 4
II	Preparation and inspection meeting stages' metrics *Mk, Mg_MEET, Ml_MEET*	Case 0 Case 1 Case 2 Case 3 Case 4

7.3 Evaluation of Metrics for Preparation Stage of Inspection M_{g_IDV} and M_{l_IDV}

7.3.1 Experimental data

The data used for metrics M_{l_IDV} and M_{g_IDV} evaluation is summarized in tables E1 and E2 in Appendix E. Data collected from preparation stage of Experiment 2 is shown in Table E1. In total, the data of 18 teams was collected. For each team, the following data is given:

- Number of defect detected by team members (total number *D_Tot*, number of major defects *D_Maj*, number of minor defects *D_Min*);
- Number of false positives *FP*;
- Number of defects which were not detected during preparation (total number *UD_Tot*, number of major defects *UD_Maj*, number of minor defects *UD_Min*);
- Cost spent on preparation *Cr*. It is a sum of time spent on preparation by three team members and the time spent for explanations before experiment (20 min per person).

The data necessary for metrics M_{l_IDV} and M_{g_IDV} calculation is given in E2 in Appendex E. It consists of the following data points:

- *CrDEF* – cost spent to detect actual defects during preparation. It was calculated using the following formula:

$$CrDEF = \frac{Cr \times D\_Tot}{D\_Tot + FP}$$

- *CrFP* – cost spent to detect false positives during preparation. It was calculated using the following formula:

$$CrFP = \frac{Cr \times FP}{D\_Tot + FP}$$

- ΔCt – cost saved by preparation stage of inspection. It was calculated considering that 8 hours of testing were saved by detecting each major defect, and 1 hour was saved by detecting each minor defect:

$$\Delta Ct = (D\_Maj \times 480) + (D\_Min \times 60)$$

- *CtDEF* – cost needed for testing to detect remaining defects. It was calculated considering that 8 hours will be needed to detect each major defect which was missed during preparation, and 1 hour will be needed to detect each minor defect:

$$CtDEF = (UD\_Maj \times 480) + (UD\_Min \times 60)$$

- *CtFP* – cost needed for testing to detect defects introduced by false positives during preparation. Different values of *CtFP* were calculated for each case of evaluation (see section 7.2.2):
  - *Case 0* (no false positives propagate into testing). In this case *CtFP=0*;
  - *Case 1* (half part of false positives propagates into testing and introduces minor defects). In this case one hour of testing will be necessary to detect and remove each defect:

$$CtFP = \frac{FP}{2} \times 60$$

  - *Case 2* (all false positives propagate into testing and introduce minor defects). In this case *CtFP* was calculated using the formula:

$$CtFP = FP \times 60$$

  - *Case 3* (half part of false positives propagates into testing and introduces major defects). In this case eight hours of testing will be necessary to detect and remove each defect:

$$CtFP = \frac{FP}{2} \times 480$$

- *Case 4* (all false positives propagate into testing and introduce major defects). In this case *CtFP* was calculated using the formula:

$$CtFP=FP\times 480$$

7.3.2 Analysis of the Results of Metrics M_k, M_{g_IDV} and M_{l_IDV}

In this subsection, we show the advantages of metric M_{g_IDV} over M_k. The values of metrics were calculated using the following formulas from Chapter 6:

- M_k: equation (6.3);
- M_{g_IDV}: equation (6.5).

In addition to the metrics M_{g_IDV} and M_k, we have calculated the values of metric M_{l_IDV} which evaluates preparation stage's losses. It was calculated using the following formula:

- M_{l_IDV}: equation (6.4).

In the following subsection we analyze the results of the metrics values for each case of evaluation (Case 0 – Case 4) (see Section 7.2.2).

Case 0

In this case of evaluation we assume that all false positives have been corrected before testing, and thus they have no influence on testing. Table 7.2 shows the values and ranks of three metrics M_k, M_{g_IDV} and M_{l_IDV} for eighteen teams. The results of metrics M_k and M_{g_IDV} are equal, since we do not include false positives in the analysis. Team T10 showed the best result in metrics M_k and M_{g_IDV}, while team T7 showed the best result in metric M_{l_IDV}.

Case 1

In this case of evaluation we assume that half part of false positives propagates into testing and introduces minor defects. Table 7.3 shows the values and ranks of three metrics M_k, M_{g_IDV} and M_{l_IDV} for eighteen teams. Spearman's rank correlation test indicates that there is a strong correlation between M_k and M_{g_IDV} (correlation coefficient is 0.96 with significance level 0.05). Furthermore, the values of metric M_{g_IDV} are significantly smaller as compared to metric M_k ($p=2.19*10^{-9}$ using paired samples t-test with significance level 0.05).

Table 7.2 Results of applying metrics M_k, M_{g_IDV} and M_{l_IDV} in Case 0

Team	System	Method	$M_k = M_{g\_IDV}$ (rank)	M_{l_IDV} (rank)
T1	Hospital	CBR	0.486 (16)	0.061 (15)
T2			0.734 (7)	0.051 (9-10)
T3			0.773 (3)	0.029 (2)
T4			0.752 (5)	0.050 (7-8)
T5		PBR	0.700 (11)	0.047 (6)
T6			0.720 (9)	0.044 (4-5)
T7			0.865 (2)	0.025 (1)
T8			0.735 (6)	0.035 (3)
T9			0.595 (13)	0.053 (12-13)
T10	Seminar	CBR	0.876 (1)	0.044 (4-5)
T11			0.728 (8)	0.054 (14)
T12			0.714 (10)	0.050 (7-8)
T13			0.767 (4)	0.053 (12-13)
T14			0.583 (15)	0.065 (16)
T15		PBR	0.587 (14)	0.052 (11)
T16			0.605 (12)	0.051 (9-10)
T17			0.439 (17)	0.084 (18)
T18			0.458 (17)	0.066 (17)

Table 7.3 Results of applying metrics M_k, M_{g_IDV} and M_{l_IDV} in Case 1

Team	System	Method	M_k (rank)	M_{g_IDV} (rank)	M_{l_IDV} (rank)
T1	Hospital	CBR	0.446 (16)	0.397 (16)	0.222 (14)
T2			0.652 (8)	0.609 (8-9)	0.206 (12)
T3			0.727 (3)	0.710 (3)	0.104 (2)
T4			0.658 (6)	0.609 (8-9)	0.224 (15)
T5		PBR	0.648 (10)	0.620 (6)	0.151 (4)
T6			0.661 (5)	0.630 (5)	0.158 (5)
T7			0.821 (1)	0.812 (1)	0.083 (1)
T8			0.692 (4)	0.673 (4)	0.113 (3)
T9			0.546 (14)	0.505 (14)	0.188 (9-10)
T10	Seminar	CBR	0.771 (2)	0.739 (2)	0.188 (9-10)
T11			0.646 (11)	0.601 (10)	0.214 (13)
T12			0.649 (9)	0.614 (7)	0.178 (8)
T13			0.654 (7)	0.594 (11)	0.259 (17)
T14			0.526 (15)	0.474 (15)	0.232 (16)
T15		PBR	0.547 (13)	0.514 (13)	0.163 (6)
T16			0.559 (12)	0.523 (12)	0.172 (7)
T17			0.399 (18)	0.339 (18)	0.280 (18)
T18			0.427 (17)	0.385 (17)	0.203 (11)

Team T7 showed the best result in all three metrics, while in Case 0 it showed the best result only in metric M_{l_IDV}. Let's compare teams T7 and T10. Both teams detect the same number of defects during preparation (10), however they differ in the number of false positives: team T7 reports 6 false positives, while team T10 reports 15 false positives (see Table E1 in Appendix E). Case 0 did not take into consideration false positives, therefore team T10, having reported more false positives than T7, outperformed team T7. However, since Case 1 that takes into account false positives, T7 outperformed T10 in metrics M_k and M_{g_IDV}. The influence of false positives reflects in the value of metric M_{l_IDV} as well: team T7 is the best in terms of M_{l_IDV}, while team T10 is only 9-10th.

Case 2

In this case of evaluation we assume that all false positives propagate into testing and introduce minor defects. Table 7.4 shows the values and ranks of three metrics M_k, M_{g_IDV} and M_{l_IDV} for eighteen teams. Spearman's rank correlation test indicates that there is a strong correlation between M_k and M_{g_IDV} (correlation coefficient is 0.98 with significance level 0.05). Furthermore, the values of metric M_{g_IDV} are significantly smaller as compared to metric M_k ($p=2.12*10^{-9}$ using paired samples t-test with significance level 0.05).

Comparing the results of the teams T7 and T10, team T7 showed the best result in all three metrics (same as in Case 1), while team T10 showed the 2nd result in metric M_k and the 4th result in metric M_{g_IDV} (in Case 1, T10 was the 2nd in both metrics M_k and M_{g_IDV}). In Case 2, the team T3 is the 2nd in metric M_{g_IDV}. Let's compare the teams T3 and T10. Team T3 reports less false positives (7) than T10 (15) (see Table E1 in Appendix E), therefore T3 outperforms T10 with respect to metrics M_{l_IDV} and M_{g_IDV}.

Case 3

In this case of evaluation we assume that half part of false positives propagates into testing and introduces major defects. Table 7.5 shows the values and ranks of three metrics M_k, M_{g_IDV} and M_{l_IDV} for eighteen teams. Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_IDV} (correlation coefficient is 0.80 with significance level 0.05). Furthermore, the values of metric M_{g_IDV} are significantly smaller as compared to metric M_k ($p=3.94*10^{-9}$ using paired samples t-test with significance level 0.05).

Table 7.4 Results of applying metrics M_k, M_{g_IDV} and M_{l_IDV} in Case 2

Team	System	Method	M_k (rank)	M_{g_IDV} (rank)	M_{l_IDV} (rank)
T1	Hospital	CBR	0.412 (16)	0.307 (17)	0.384 (14)
T2			0.587 (8)	0.484 (8)	0.362 (12)
T3			0.687 (3)	0.648 (2)	0.178 (2)
T4			0.585 (9)	0.466 (10)	0.398 (15)
T5		PBR	0.603 (6)	0.539 (6)	0.256 (4)
T6			0.611 (5)	0.541 (5)	0.272 (5)
T7			0.782 (1)	0.758 (1)	0.141 (1)
T8			0.654 (4)	0.610 (3)	0.190 (3)
T9			0.505 (14)	0.416 (14)	0.323 (9)
T10	Seminar	CBR	0.688 (2)	0.603 (4)	0.333 (10)
T11			0.581 (10)	0.474 (9)	0.373 (13)
T12			0.595 (7)	0.514 (7)	0.306 (8)
T13			0.570 (11)	0.421 (13)	0.466 (17)
T14			0.479 (15)	0.365 (15)	0.399 (16)
T15		PBR	0.512 (13)	0.442 (11)	0.274 (6)
T16			0.520 (12)	0.441 (12)	0.294 (7)
T17			0.366 (18)	0.239 (18)	0.477 (18)
T18			0.400 (17)	0.313 (16)	0.341 (11)

Table 7.5 Results of applying metrics M_k, M_{g_IDV} and M_{l_IDV} in Case 3

Team	System	Method	M_k (rank)	M_{g_IDV} (rank)	M_{l_IDV} (rank)
T1	Hospital	CBR	0.284 (17)	-0.228 (12)	1.351 (14)
T2			0.367 (9)	-0.266 (13)	1.295 (12)
T3			0.515 (2)	0.273 (2)	0.625 (2)
T4			0.351 (12)	-0.391 (17)	1.441 (16)
T5		PBR	0.426 (4)	0.057 (4)	0.884 (4)
T6			0.420 (5)	0.005 (5-6)	0.953 (6)
T7			0.606 (1)	0.437 (1)	0.487 (1)
T8			0.490 (3)	0.235 (3)	0.657 (3)
T9			0.347 (13)	-0.120 (9)	1.134 (9)
T10	Seminar	CBR	0.419 (6)	-0.215 (11)	1.198 (11)
T11			0.361 (11)	-0.290 (15)	1.327 (13)
T12			0.396 (7)	-0.086 (8)	1.074 (8)
T13			0.322 (14)	-0.615 (18)	1.705 (18)
T14			0.311 (15)	-0.289 (14)	1.399 (15)
T15		PBR	0.371 (8)	0.005 (5-6)	0.941 (5)
T16			0.365 (10)	-0.050 (7)	1.024 (7)
T17			0.244 (18)	-0.361 (16)	1.655 (17)
T18			0.290 (16)	-0.124 (10)	1.169 (10)

Comparing the results of the teams T7 and T10, team T7 is the best with respect to all metrics, while team T10 exhibited even greater difference between M_k and M_{g_IDV} values than in Case 2: now T10 is the 6th in M_k, and the 11th in M_{g_IDV}. This is due to the increased influence of false positives over testing costs: in Case 2 false positive introduced a minor defect that costs 1 hour of testing, while in Case 3 half part of false positives introduced a major defect that costs 8 hours of testing. Thus, since T10 reported a lot more false positives (15) than T7 (6), the difference between M_k and M_{g_IDV} ranks increased.

Case 4

In this case of evaluation we assume that all false positives propagate into testing and introduce major defects. Table 7.6 shows the values and ranks of three metrics M_k, M_{g_IDV} and M_{l_IDV} for eighteen teams. Spearman's rank correlation test indicates that there is no strong correlation between M_k and M_{g_IDV} (correlation coefficient is 0.68 with significance level 0.05).

Two teams that have the greatest difference between the ranks of M_k and M_{g_IDV} are T10 and T18. Team T10 has higher rank in metric M_k (6) as compared to the rank in metric M_{g_IDV} (15). The difference in ranks of this team could be influenced by a high number of false positives reported (15) (see Table E1 in Appendix E) and a high influence of false positives on testing costs (each false positive introduces a major defect) in this case: M_k does not take into consideration the number of false positives, therefore the rank of M_k is higher as compared to the rank of M_{g_IDV} that includes false positives. Team T18 is opposite to team T10, since it has higher rank in metric M_{g_IDV} (6-7) as compared to the rank in metric M_k (14-15). The difference in ranks is due to the small number of false positives false positives (8) and small number of major defects (3) reported by teams T18 (see Table E1 in Appendix E): the testing cost saved by inspection is low because few major defects are detected during inspection, therefore the rank of M_k is low; however the testing cost needed for testing to detect defects introduced by false positives is low since few false positives are introduced, therefore the rank of M_{g_IDV} is high.

After removing the data of these two teams (T10 and T18), the value of Spearman's rank correlation coefficient becomes 0.83. Furthermore, the values of metric M_{g_IDV} are significantly smaller as compared to metric M_k ($p=2.94*10^{-9}$ using paired samples t-test with significance level 0.05).

Table 7.6 Results of applying metrics M_k, M_{g_IDV} and M_{l_IDV} in Case 4

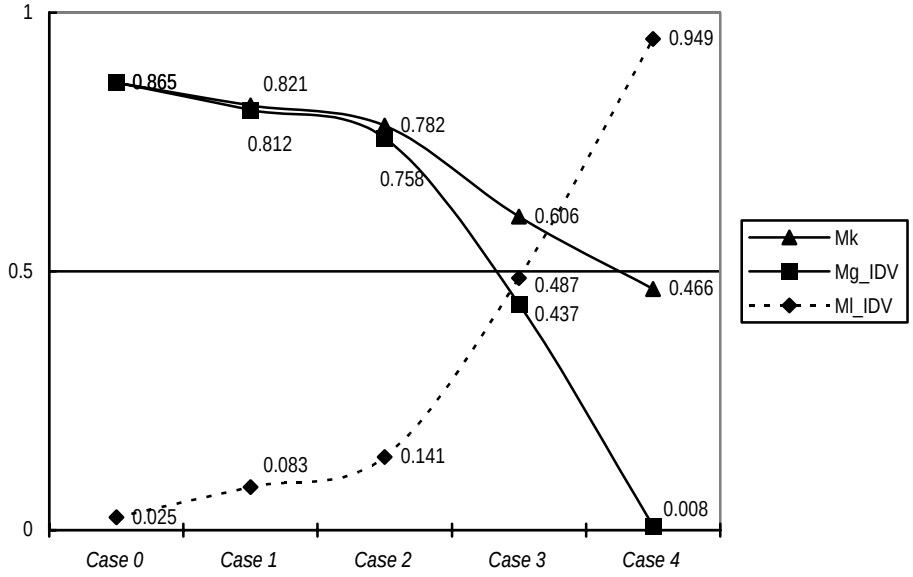
Team	System	Method	M_k (rank)	M_{g_IDV} (rank)	M_{l_IDV} (rank)
T1	Hospital	CBR	0.200 (17)	-0.943 (11)	2.642 (14)
T2			0.245 (10-11)	-1.266 (14)	2.540 (12)
T3			0.386 (2)	-0.227 (2)	1.221 (2)
T4			0.229 (13)	-1.534 (17)	2.833 (16)
T5		PBR	0.306 (4)	-0.586 (5)	1.721 (4)
T6			0.296 (5)	-0.709 (8)	1.862 (6)
T7			0.466 (1)	0.008 (1)	0.949 (1)
T8			0.368 (3)	-0.265 (3)	1.279 (3)
T9			0.245 (10-11)	-0.834 (9)	2.215 (9)
T10	Seminar	CBR	0.275 (6)	-1.306 (15)	2.352 (11)
T11			0.240 (12)	-1.308 (16)	2.600 (13)
T12			0.274 (7)	-0.886 (10)	2.097 (8)
T13			0.204 (16)	-1.997 (18)	3.357 (18)
T14			0.212 (14-15)	-1.162 (13)	2.732 (15)
T15		PBR	0.271 (8)	-0.577 (4)	1.829 (5)
T16			0.262 (9)	-0.705 (6-7)	1.997 (7)
T17			0.169 (18)	-1.161 (12)	3.227 (17)
T18			0.212 (14-15)	-0.705 (6-7)	2.272 (10)

7.3.3 Influence of False Positives on Preparation Stage's Metrics

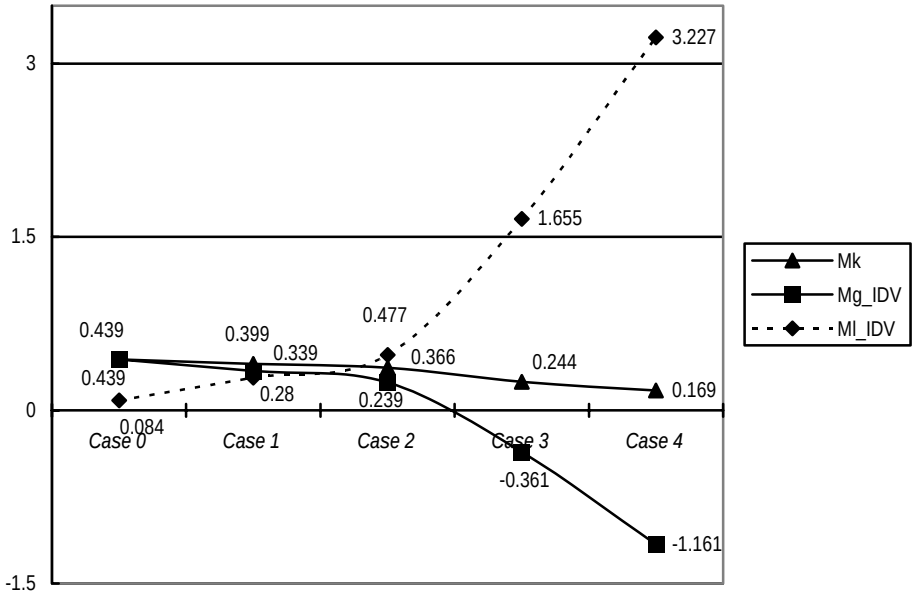
In this subsection we analyze the influence of false positives on metrics M_k, M_{g_IDV} and M_{l_IDV} values with respect to two teams:

(a) One of the best-performing teams – T7 (detected 6 major defects and 6 false positives);
(b) One of the worst-performing teams – T17 (detected 3 major defects and 11 false positives) (see Table E1 in Appendix E).

The graphs of metrics values for each case of evaluation (Case 0-4) are shown in Figure 7.1.



(a) Team T7 (one of the best-performing teams).



(b) Team T17 (one of the worst-performing teams).

Figure 7.1 Comparing preparation stage's metrics values of two inspection teams.

As we can see from Figure 7.1, when the influence of false positives on testing costs increases (going from Case 0 to Case 4), the values of:

- Metrics M_k and M_{g_IDV} decrease, i.e. inspection becomes less cost-effective;

- Metric M_{l_IDV} increases, i.e. inspection exhibits greater losses.

Comparing the results of the best-performing (T7) and worst-performing (T17) teams, we get to know that:

- Best-performing team exhibits smaller difference between metrics M_k and M_{g_IDV} values as compared to the worst-performing team;
- As the influence of false positives on testing costs increases, the increase in the value of metric M_{l_IDV} of the best-performing team is smaller as compared to the worst-performing team.

7.4 Evaluation of Metrics for Preparation and Inspection Meeting Stages M_{g_MEET} and M_{l_MEET}

7.4.1 Experimental Data

The data used for metrics M_{l_MEET} and M_{g_MEET} evaluation is summarized in tables E3-E6 in Appendix E.

Table E3 in Appendix E describes the data collected form inspection meeting stage. In total, the data of 18 teams was collected. For each team, the following data is given:

- Number of defect detected by team members (total number *TD_Tot*, number of major defects *TD_Maj*, number of minor defects *TD_Min*);
- Number of additional defects detected during team meeting (total number *AD_Tot*, number of major defects *AD_Maj*, number of minor defects *AD_Min*);
- Number of defects lost during team meeting (total number *LD_Tot*, number of major defects *LD_Maj*, number of minor defects *LD_Min*);
- Number of undetected defects left in inspection documents after inspection meeting (total number *TL_Tot*, number of major defects *TL_Maj*, number of minor defects *TL_Min*). They were calculated by subtracting the number of detected defects from the total number of defects existing in inspected diagrams. The defects lost during meeting (i.e. the defects which were detected during preparation but not included into defect list during inspection meeting) were not included into the number of undetected defects;
- Number of false positives detected by a team (total number *TFP*, new false

positives detected during inspection meeting *AFP*);

- Number of false positives eliminated during inspection meeting *EFP*;
- Time spent on team meeting *Cm*.

Table E4 in Appendix E summarizes the cost of preparation and inspection meeting stages for each team. While calculating costs *CmDEF*, *CmFP*, *CmADD_DEF*, *CmADD_FP*, *CmLOST_DEF* and *CmELIM_FP* we assumed, that equal time was spent to: a) confirm one defect; b) confirm one false positive; c) detect a new defect; d) detect an additional false positive; e) eliminate a defect; and f) eliminate a false positive during inspection meeting. The following data is given:

- *Cr* – cost of the preparation stage of inspection, which is a sum of time spent on preparation by three team members;
- *CmDEF* – cost spent to confirm actual defects detected during preparation. It was calculated using the following formula:

$$CmDEF = \frac{Cm \times TD\_Tot}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *CmFP* – cost spent to confirm false positives detected during preparation. It was calculated using the formula:

$$CmFP = \frac{Cm \times TFP}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *CmADD_DEF* – cost spent to detect additional defects during inspection meeting. It was calculated using the following formula:

$$CmADD\_DEF = \frac{Cm \times AD\_Tot}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *CmADD_FP* – cost spent to detect additional false positives during inspection meeting. It was calculated using the formula:

$$CmADD\_FP = \frac{Cm \times AFP}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *CmLOST_DEF* – cost spent to eliminate actual defects detected during preparation stage. It was calculated using the formula:

$$CmLOST\_DEF = \frac{Cm \times LD\_Tot}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *CmELIM_FP* – cost spent to eliminate false positives detected during preparation stage. It was calculated using the formula:

$$CmELIM\_FP = \frac{Cm \times EFP}{TD\_Tot + AD\_Tot + LD\_Tot + TFP + AFP + EFP}$$

- *Cm* – total inspection meeting cost, which is a sum of all meeting costs;

- *CtDEF* – testing cost spent to detect remaining defects after preparation and inspection meeting (excluding the cost spent to detect defects lost during inspection meeting). It was calculated assuming that 8 hours of testing will be necessary to detect 1 each major defect, and 1 hour to detect minor defect. The following formula was used:

$$CtDEF=(TL\_Maj\times 480)+(TL\_Min\times 60)$$

- *CtLOST_DEF* – testing cost needed to detect defects lost during inspection meeting. It was calculated using the formula:

$$CtLOST\_DEF=(LD\_Maj\times 480)+(LD\_Min\times 60)$$

Tables E5 and E6 in Appendix E define the testing costs, used for metrics evaluation of preparation and inspection meeting stages. For each team, the following costs are given:

- *ΔCtADD_DEF* – testing cost saved by additional defects detected during inspection meeting. It was calculated using the formula:

$$\Delta CtADD\_DEF=(AD\_Maj\times 480)+(AD\_Min\times 60)$$

- *ΔCtDEF* – testing cost saved by defects detected during preparation and confirmed during inspection meeting stage. It was calculated using the following formula:

$$\Delta CtDEF=(TD\_Maj\times 480)+(TD\_Min\times 60)$$

The costs *CtADD_FP, CtFP, ΔCtELIM_FP, Ct* and *ΔCt* were calculated for each evaluation case (Cases 0-4) separately:

- *Ct* was calculated as a sum of all costs spent on testing (*CtDEF, CtFP, CtADD_FP, CtLOST_DEF*);

- *ΔCt* was calculated as a sum of all testing costs saved by inspection (*ΔCtADD_DEF, ΔCtELIM_FP, ΔCtDEF*);

- Costs *CtADD_FP, CtFP, ΔCtELIM_FP* were calculated in the following way:

- Case 0 – no false positives propagate into testing. In this case:

$$CtADD\_FP=CtFP=\Delta CtELIM\_FP=0$$

- Case 1 – half part of false positives propagates into testing and introduces minor defects. In this case:

$$CtADD\_FP = \frac{AFP}{2} \times 60$$

$$CtFP = \frac{FP}{2} \times 60 - CtADD\_FP$$

$$\Delta CtELIM\_FP = \frac{EFP}{2} \times 60$$

- Case 2 – all false positives propagate and introduce minor defects:

$$CtADD\_FP=AFP*60$$

$$CtFP=FP*60 - CtADD\_FP$$

$$\Delta CtELIM\_FP=EFP*60$$

- Case 3 – half part of false positives propagates and introduces major defects:

$$CtADD\_FP = \frac{AFP}{2} \times 480$$

$$CtFP = \frac{FP}{2} \times 480 - CtADD\_FP$$

$$\Delta CtELIM\_FP = \frac{EFP}{2} \times 480$$

- Case 4 – all false positives propagate into testing and introduce major defects:

$$CtADD\_FP=AFP\times 480$$

$$CtFP=FP\times 480 - CtADD\_FP$$

$$\Delta CtELIM\_FP=EFP\times 480$$

7.4.2 Analysis of the Results of Metrics M_k, M_{g_MEET} and M_{I_MEET}

The values of metrics were calculated using the following formulas from Chapter 6:

- M_k: equation (6.3);

- M_{g_MEET}: equation (6.7);
- M_{l_MEET}: equation (6.6).

In the following subsections we analyze the results of the metrics values and compare metric M_{g_IDV} against metric M_k.

Case 0

In this case of evaluation we assume that all false positives have been corrected before testing. Table 7.7 shows the values and ranks of three metrics M_k, M_{g_MEET} and M_{l_MEET} for eighteen teams.

Table 7.7 Results of applying metric M_k, M_{g_MEET} and M_{l_MEET} in Case 0

Team	System	Method	*Mk* (rank)	*Mg_MEET* (rank)	*Ml_MEET* (rank)
T1	Hospital	CBR	0.324 (14)	0.211 (12)	---
T2			0.283 (15)	-0.296 (16-17)	25.192 (3)
T3			0.445 (8)	0.203 (13)	---
T4			0.714 (3)	0.708 (3)	---
T5		PBR	0.539 (7)	0.463 (6)	---
T6			0.681 (4)	0.620 (4)	1.148 (2)
T7			0.571 (5)	0.399 (8)	---
T8			0.716 (2)	0.716 (2)	---
T9			0.861 (1)	0.858 (1)	0.061 (1)
T10	Seminar	CBR	0.262 (18)	-0.764 (18)	---
T11			0.269 (17)	-0.296 (16-17)	---
T12			0.547 (6)	0.470 (5)	---
T13			0.430 (9)	0.175 (14)	---
T14			0.411 (11)	0.310 (10)	---
T15		PBR	0.275 (16)	-0.022 (15)	---
T16			0.402 (13)	0.269 (11)	---
T17			0.421 (10)	0.421 (7)	---
T18			0.404 (12)	0.381 (9)	---

Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_MEET} (correlation coefficient is 0.91 with significance level 0.05). Furthermore, the values of metric M_{g_MEET} are significantly smaller as compared to metric M_k (p=0.0046 using paired samples t-test with significance level 0.05).

Regarding metric M_{l_MEET}, for most of the teams it was impossible to compute the values of this metric. This is due to the assumption that false positives do not

have influence on testing in this case of evaluation. Therefore, since the denominator of the formula to compute M_{l_MEET} (see equation 6.6) is defined as a sum of testing cost saved by additional defect detected during inspection meeting and testing cost saved by false positives eliminated during inspection meeting (equals to zero in this case of evaluation), M_{l_MEET} could be computed only for the teams that detected additional defects during inspection meeting (T2, T6 and T9) (see Table E3 in Appendix E).

Case 1

In this case of evaluation we assume that half part of false positives propagates into testing and introduces minor defects. Table 7.8 shows the values and ranks of three metrics M_k, M_{g_MEET} and M_{l_MEET} for eighteen teams.

Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_MEET} (correlation coefficient is 0.93 with significance level 0.05). Furthermore, the values of metric M_{g_MEET} are significantly smaller as compared to metric M_k (p=0.0011 using paired samples t-test with significance level 0.05).

Table 7.8 Results of applying metric M_k, M_{g_MEET} and M_{l_MEET} in Case 1

Team	System	Method	*Mk* (rank)	*Mg_MEET* (rank)	*Ml_MEET* (rank)
T1	Hospital	CBR	0.327 (15)	0.172 (13)	4.300 (13)
T2			0.315 (18)	-0.233 (17)	5.038 (15)
T3			0.441 (11)	0.171 (14)	11.838 (18)
T4			0.692 (3)	0.666 (3)	0.422 (4)
T5		PBR	0.552 (7)	0.470 (6)	2.459 (9)
T6			0.656 (4)	0.569 (4)	0.970 (5)
T7			0.574 (5)	0.399 (8)	6.840 (17)
T8			0.699 (2)	0.688 (2)	0.000 (1)
T9			0.848 (1)	0.841 (1)	0.051 (2)
T10	Seminar	CBR	0.324 (16)	-0.468 (18)	5.471 (16)
T11			0.349 (14)	-0.071 (16)	3.529 (12)
T12			0.567 (6)	0.487 (5)	1.913 (8)
T13			0.460 (8)	0.208 (12)	2.858 (10)
T14			0.449 (10)	0.344 (9)	1.769 (6)
T15		PBR	0.316 (17)	0.050 (15)	4.613 (14)
T16			0.422 (12)	0.283 (11)	3.397 (11)
T17			0.454 (9)	0.440 (7)	0.122 (3)
T18			0.398 (13)	0.343 (10)	1.778 (7)

Case 2

In this case of evaluation we assume that all false positives propagate into testing and introduce minor defects. Table 7.9 shows the values and ranks of three metrics M_k, M_{g_MEET} and M_{l_MEET} for eighteen teams. Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_MEET} (correlation coefficient is 0.92 with significance level 0.05). Furthermore, the values of metric M_{g_MEET} are significantly smaller as compared to metric M_k (p=0.0003 using paired samples t-test with significance level 0.05).

Table 7.9 Results of applying metric M_k, M_{g_MEET} and M_{l_MEET} in Case 2

Team	System	Method	*Mk* (rank)	*Mg_MEET* (rank)	*Ml_MEET* (rank)
T1	Hospital	CBR	0.330 (18)	0.137 (14)	2.275 (13)
T2			0.340 (17)	-0.184 (17)	2.799 (16)
T3			0.436 (12)	0.141 (13)	6.086 (18)
T4			0.675 (3)	0.630 (3)	0.311 (4)
T5		PBR	0.564 (7)	0.476 (6)	1.301 (9)
T6			0.635 (4)	0.522 (4)	0.860 (5)
T7			0.578 (6)	0.399 (8)	3.620 (17)
T8			0.684 (2)	0.662 (2)	0.000 (1)
T9			0.836 (1)	0.826 (1)	0.044 (2)
T10	Seminar	CBR	0.372 (15)	-0.274 (18)	2.777 (15)
T11			0.412 (13)	0.085 (16)	1.800 (12)
T12			0.584 (5)	0.502 (5)	1.012 (6)
T13			0.482 (8)	0.233 (12)	1.429 (10)
T14			0.480 (9-10)	0.372 (9)	1.021 (7)
T15		PBR	0.351 (16)	0.112 (15)	2.306 (14)
T16			0.439 (11)	0.296 (11)	1.699 (11)
T17			0.480 (9-10)	0.456 (7)	0.117 (3)
T18			0.394 (14)	0.307 (10)	1.056 (8)

Case 3

In this case of evaluation we assume that half part of false positives propagates into testing and introduces major defects. Table 7.10 shows the values and ranks of three metrics M_k, M_{g_MEET} and M_{l_MEET} for eighteen teams. Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_MEET} (correlation coefficient is 0.92 with significance level 0.05). Furthermore, the values of metric M_{g_MEET} are significantly smaller as compared to metric M_k ($p=4.01*10^{-6}$ using paired samples t-test with significance level 0.05).

Table 7.10 Results of applying metric M_k, M_{g_MEET} and M_{l_MEET} in Case 3

Team	System	Method	*Mk* (rank)	*Mg_MEET* (rank)	*Ml_MEET* (rank)
T1	Hospital	CBR	0.341 (18)	-0.029 (18)	0.756 (14)
T2			0.427 (15)	-0.019 (17)	0.763 (16)
T3			0.420 (16)	-0.001 (16)	1.771 (18)
T4			0.625 (3)	0.494 (6)	0.228 (4)
T5		PBR	0.606 (6)	0.503 (5)	0.432 (8)
T6			0.559 (10)	0.316 (12)	0.614 (13)
T7			0.591 (7)	0.399 (9)	1.205 (17)
T8			0.620 (4)	0.531 (3)	0.000 (1)
T9			0.794 (1)	0.761 (1)	0.023 (2)
T10	Seminar	CBR	0.525 (12)	0.203 (14)	0.757 (15)
T11			0.616 (5)	0.492 (7)	0.504 (10)
T12			0.642 (2)	0.555 (2)	0.336 (5)
T13			0.547 (11)	0.310 (13)	0.357 (6)
T14			0.579 (8)	0.468 (8)	0.460 (9)
T15		PBR	0.496 (14)	0.345 (11)	0.577 (12)
T16			0.507 (13)	0.349 (10)	0.425 (7)
T17			0.574 (9)	0.518 (4)	0.113 (3)
T18			0.376 (17)	0.126 (15)	0.514 (11)

Case 4

In this case of evaluation we assume that all false positives propagate into testing and introduce major defects. Table 7.11 shows the values and ranks of three metrics M_k, M_{g_MEET} and M_{l_MEET} for eighteen teams. Spearman's rank correlation test indicates that there is strong correlation between M_k and M_{g_MEET} (correlation coefficient is 0.95 with significance level 0.05). Furthermore, the values of metric M_{g_MEET} are significantly smaller as compared to metric M_k ($p=1.43*10^{-5}$ using paired samples t-test with significance level 0.05).

Table 7.11 Results of applying metric M_k, M_{g_MEET} and M_{l_MEET} in Case 4

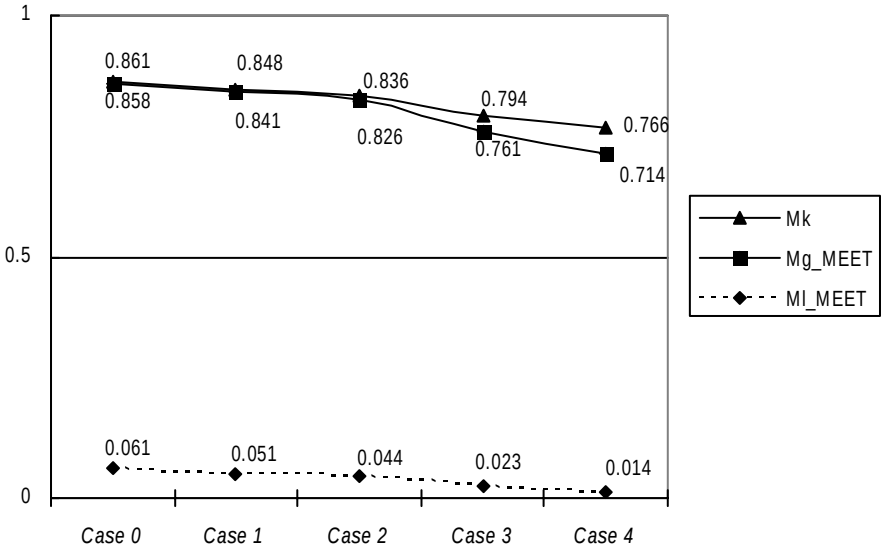
Team	System	Method	*Mk* (rank)	*Mg_MEET* (rank)	*Ml_MEET* (rank)
T1	Hospital	CBR	0.348 (18)	-0.174 (18)	0.503 (15)
T2			0.475 (15)	0.072 (15)	0.388 (12)
T3			0.408 (16)	-0.128 (17)	1.052 (18)
T4			0.600 (8-9)	0.407 (8)	0.214 (6)
T5		PBR	0.634 (4)	0.521 (6)	0.288 (8-10)
T6			0.514 (14)	0.151 (14)	0.525 (16)
T7			0.600 (8-9)	0.400 (10)	0.803 (17)
T8			0.572 (12)	0.401 (9)	0.000 (1)
T9			0.766 (1)	0.714 (1)	0.014 (2)
T10	Seminar	CBR	0.603 (7)	0.390 (11)	0.420 (13)
T11			0.725 (2)	0.663 (2)	0.288 (8-10)
T12			0.676 (3)	0.589 (3)	0.224 (7)
T13			0.578 (11)	0.348 (13)	0.179 (4)
T14			0.632 (5)	0.523 (5)	0.366 (11)
T15		PBR	0.598 (10)	0.496 (7)	0.288 (8-10)
T16			0.552 (13)	0.388 (12)	0.212 (5)
T17			0.630 (6)	0.560 (4)	0.112 (3)
T18			0.364 (17)	-0.049 (16)	0.424 (14)

7.4.3 Influence of False Positives on Preparation and Inspection Meeting Stage's Metrics

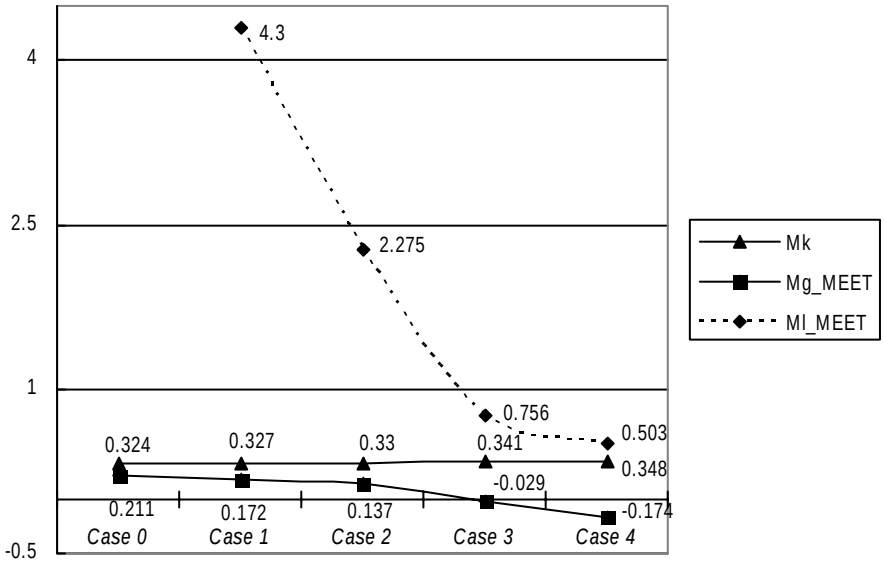
In this subsection we analyze the influence of false positives on metrics M_k, M_{g_MEET} and M_{l_MEET} values with respect to two teams:

(a) One of the best-performing teams – T9 (detected 6 major defects (2 of them during inspection meeting), reported 3 false positives, and eliminated 7 false positives during inspection meeting);
(b) One of the worst-performing teams – T1 (detected 2 major defects (0 during inspection meeting), reported 7 false positives, and eliminated 4 false positives during inspection meeting) (see Table E3 in Appendix E).

The graphs of metrics values for each case of evaluation (Case 0-4) are shown in Figure 7.2.



(a) Team T9 (one of the best-performing teams).



(b) Team T1 (one of the worst-performing teams).

Figure 7.2 Comparing preparation and inspection meeting stages' metrics values of two inspection teams.

As we can see from Figure 7.2, when the influence of false positives on testing costs increases (going from Case 0 to Case 4), the values of:

- Metric M_{I_MEET} decreases, i.e. inspection team experience smaller meeting

losses. This is due to the ability of inspection teams to eliminate false positives reported during preparation. Therefore, as the influence of false positives on testing cost increases, the usefulness of inspection meetings increases as well;

- Metrics M_k and M_{g_MEET} either increase or decrease. This happens because metrics depend on both preparation and inspection meeting stages, where false positives have different impact:
  - During preparation, false positives have only negative impact – if the number of false positives increases, cost-effectiveness decreases;
  - During inspection meeting, teams usually report fewer false positives as compared to the number of false positives they eliminate. Therefore, if a team is good at eliminating false positives during inspection meeting, its cost-effectiveness increases when the influence of false positives grows.

Comparing the results of the best-performing (T9) and worst-performing (T1) teams, we get to know that:

- Best-performing team exhibits smaller difference between metrics M_k and M_{g_MEET} values as compared to the worst-performing team;
- As the influence of false positives on testing costs increases, the value of metric M_{l_MEET} of the worst-performing team decreases more rapidly as compared to the best-performing team.

7.5 Correlation between Metrics M_k and M_{g_IDV}, and between Metrics M_k and M_{g_MEET}

The values of Spearman's rank correlation test for all cases of evaluation are summarized in Table 7.12. The mean value of rank correlation between metrics M_k and M_{g_IDV} is 0.89, and between metrics M_k and M_{g_MEET} is 0.93. This implies that metric M_k and metrics M_{g_IDV} and M_{g_MEET} produce nearly identical measurements.

Table 7.12 Spearman's rank correlation coefficients

Inspection stages	Correlation between metrics	Case of evaluation	Spearman's rank correlation coefficient
Preparation	*Mk* and *Mg_IDV*	Case 0	1
		Case 1	0.96
		Case 2	0.98
		Case 3	0.80
		Case 4	0.68
		Mean	0.89
Preparation and inspection meeting	*Mk* and *Mg_MEET*	Case 0	0.91
		Case 1	0.93
		Case 2	0.92
		Case 3	0.92
		Case 4	0.95
		Mean	0.93

7.6 Difference between M_k and M_{g_IDV}, and between M_k and M_{g_MEET}

The results of paired samples t-test to evaluate whether there is a significant difference between metrics M_k and M_{g_IDV}, and between metrics M_k and M_{g_MEET}, are summarized in Table 7.13.

Table 7.13 Results of comparison between metrics M_k and M_{g_IDV}, and between metrics M_k and M_{g_MEET}

Inspection stages	Comparison of metrics	Case of evaluation	Mean		p-value
			Mk	*Mg_IDV or Mg_MEET*	
Preparation	*Mk* and *Mg_IDV*	Case 0	0.67	0.67	---
		Case 1	0.61	0.57	2.19E-09*
		Case 2	0.56	0.48	2.12E-09*
		Case 3	0.38	-0.11	3.94E-09*
		Case 4	0.27	-0.90	2.94E-09*
Preparation and inspection meeting	*Mk* and *Mg_MEET*	Case 0	0.48	0.27	0.0046*
		Case 1	0.49	0.30	0.0011*
		Case 2	0.50	0.32	0.0003*
		Case 3	0.55	0.35	4.01E-06*
		Case 4	0.57	0.35	1.43E-05*

* indicates significant results, i.e. $p<0.05$

As we can see from Table 7.13, the values of metrics M_{g_IDV} and M_{g_MEET} are significantly smaller as compared to metric M_k in all but one cases of evaluation (in Case 0 of preparation stage, M_k=M_{g_IDV}).

7.7 Comparison of Preparation Stage's Metrics against Preparation and Inspection Meeting Stages' Metrics

We used paired samples t-test with significance level 0.05 to compare the metrics values of inspection teams between preparation stage, and preparation and inspection meeting stages. The following comparisons were made:

- The value of metric M_k of preparation stage has been compared against the value of metric M_k of preparation and inspection meeting stages;
- The value of metric M_{g_IDV} has been compared against the value of metric M_{g_MEET};
- The value of metric M_{l_IDV} has been compared against the value of metric M_{l_MEET}.

The results of the comparison are summarized in the Table 7.14. The following are the results:

- **Metric M_k**: when the influence of false positives on testing costs is low (Cases 0 and 1), inspection teams exhibit significantly higher cost-effectiveness during preparation stage as compared to preparation and inspection meeting stages. In Case 2, there is no significant difference. However, when the influence of false positives increases (Cases 3 and 4), cost-effectiveness of inspection teams during preparation stage becomes significantly smaller than during preparation and inspection meeting stages;
- **Metrics M_{g_IDV} and M_{g_MEET}**: when the influence of false positives on testing costs is low (Cases 0, 1 and 2), inspection teams exhibit significantly higher extended cost-effectiveness during preparation stage as compared to preparation and inspection meeting stages. However, when the influence of false positives increases (Cases 3 and 4), extended cost-effectiveness of inspection teams during preparation stage becomes significantly smaller than during preparation and inspection meeting stages.
- **Metrics M_{l_IDV} and M_{l_MEET}**: when the influence of false positives on testing costs is low (Cases 1 and 2), inspection teams exhibit significantly smaller preparation losses as compared to inspection meeting losses. However, when the influence of false positives increases (Cases 3 and 4), meeting

losses of preparation stage are significantly higher than those of inspection meeting stage.

Table 7.14 Results of comparison of metric values between preparation stage, and preparation and inspection meeting stages

Comparison between metrics	Case of evaluation	Mean		p-value
		Preparation stage	Preparation and inspection meeting stages	
Mk	Case 0	0.67	0.48	0.0004*
	Case 1	0.61	0.49	0.005*
	Case 2	0.56	0.50	0.063
	Case 3	0.38	0.55	0.00002*
	Case 4	0.27	0.57	0.00000002*
Mg_IDV and *Mg_MEET*	Case 0	0.67	0.27	0.0008*
	Case 1	0.57	0.30	0.003*
	Case 2	0.48	0.32	0.024*
	Case 3	-0.11	0.35	0.00001*
	Case 4	-0.90	0.35	0.00000005*
Ml_IDV and *Ml_MEET*	Case 1	0.19	3.19	0.0003*
	Case 2	0.32	1.70	0.0008*
	Case 3	1.13	0.55	0.0007*
	Case 4	2.21	0.35	0.00000001*

* indicates significant results, i.e. p<0.05

7.8 Comparison of Metric Values between CBR and PBR Teams

7.8.1 Metrics M_k, M_{g_IDV} and M_{l_IDV}

We used independent samples t-test [Norŭsis 1995] with significance level 95% (p<0.05) to test whether there is a difference in the values of metrics M_k, M_{g_IDV} and M_{l_IDV} values between CBR and PBR teams. The results of comparison are summarized in Table E7 in Appendix E. The following are the results:

- **Metric M_k**: there is no statistically significant difference between CBR and PBR, without influence inspected system. However, when separating the data into two groups (Hospital and Seminar systems):
  - Teams that inspected Hospital system did not show significant difference;
  - Teams that inspected Seminar system have exhibited statistically

significant difference in the cases of evaluation when the influence of false positives on testing is low (Cases 0, 1 and 2). In all these cases, CBR teams were more cost-effective as compared to PBR teams.

- **Metric M_{g_IDV}**: there is no statistically significant difference between CBR and PBR when the influence of false positives on testing costs is low (Cases 0, 1 and 2). However, when the influence of false positives increases (Cases 3 and 4), PBR becomes significantly more cost-effective than CBR (p=0.021 in Case 3; and p=0.005 in Case 4). When separating the data into two groups (Hospital and Seminar system):
  - There is no statistically significant difference for teams that inspected Hospital system;
  - The teams that inspected Seminar system have exhibited statistically significant difference in 3 cases of evaluation (Cases 0, 1, 4). In two cases (0 and 1), CBR teams had higher values of M_{g_IDV} as compared to PBR teams. In Case 4, PBR teams showed higher values of M_{g_IDV} than CBR teams.

- **Metric M_{l_IDV}**: there is no statistically significant difference between CBR and PBR when the influence of false positives on testing costs is low (Cases 0, 1 and 2). However, when the influence of false positives increases (Cases 3 and 4), PBR exhibits significantly smaller preparation losses than CBR (p=0.039 in Case 3; and p=0.036 in Case 4). When separating the data into two groups (Hospital and Seminar system), the results of metric M_{l_IDV} did not show significant difference between CBR and PBR.

7.8.2 Metrics M_k, M_{g_MEET} and M_{l_MEET}

The results of comparison of metrics M_k, M_{g_MEET} and M_{l_MEET} values between CBR and PBR teams are summarized in Table E8 in Appendix E. We used independent samples t-test [Norŭsis 1995] with significance level 95% ($p<0.05$) to test whether there is a difference between CBR and PBR. The following are the results:

- **Metric M_k**: there is no statistically significant difference between CBR and PBR, without influence inspected system. However, when separating the data into two groups (Hospital and Seminar systems):
  - PBR teams that inspected Hospital system are more cost-effective than

CBR teams;

- There is no statistically significant difference for teams that inspected Seminar system in four cases of evaluation (Cases 0, 1, 2, 4). In Case 3, CBR teams are significantly more cost-effective than PBR teams.

- **Metric M_{g_MEET}**: when the influence of false positives on testing costs is low (Cases 0, 1 and 2), PBR is significantly more cost-effective than CBR (p=0.023 in Case 0; p=0.021 in Case 1; and p=0.021 in Case 2). However, when the influence of false positives increases (Cases 3 and 4), there is no significant difference between CBR and PBR. When separating the data into two groups (Hospital and Seminar system):

  - The teams that inspected Hospital system show significantly higher values of M_{g_MEET} than CBR teams, i.e. PBR is more cost effective than CBR;
  - There is no statistically significant difference for teams that inspected Seminar system.

- **Metric M_{l_MEET}**: there is no statistically significant difference between CBR and PBR without influence of inspected system as well as separating the data into two groups (Hospital and Seminar systems).

7.9 Interpretation of Results

This section presents an interpretation of metrics evaluation results. The evaluation comprises the following steps:

1) At the beginning of analysis we calculated metrics values for different cases of evaluation as described in Table 7.1. Then, we compared the values of cost-effectiveness metric M_k and extended cost-effectiveness metrics M_{g_IDV} and M_{g_MEET} using two statistical tests:

   - Spearman's rank correlation test with significance level 0.05 to evaluate whether there is a strong correlation between the values of metrics (see Section 7.5);

   - Paired samples t-test with significance level 0.05 to evaluate whether there is a significant difference in metrics values (see Section 7.6).

The results of the Spearman's rank correlation test indicate that there is a strong correlation between metrics M_k and M_{g_IDV}, and between metrics M_k and

M_{g_MEET} (see Table 7.12).

Paired samples t-test revealed that the values of metrics M_{g_IDV} and M_{g_MEET} are significantly smaller as compared to metric M_k. The result was significant in four (out of five) but one cases of evaluation (in Case 0 of preparation stage, M_k=M_{g_IDV}) (see Table 7.13).

2) In addition, we analyzed the influence of false positives on the metrics values (see sections 7.3.3 and 7.4.3), i.e. we investigated the change in metrics values from Case 0 to Case 4 of evaluation.

The results of preparation stage's metrics showed that the values of metrics M_k and M_{g_IDV} decrease and of metric M_{l_IDV} increase, as the influence false positives on testing costs increases (going from Case 0 to Case 4). In other words, the inspection becomes less cost-effective and exhibits greater losses.

The results of preparation and inspection meeting stage's metrics showed that the values of Metrics M_k and M_{g_MEET} either increase or decrease, and of metric M_{l_MEET} decrease, as the influence of false positives on testing costs increases. This happens since metrics M_k and M_{g_MEET} depend on both preparation and inspection meeting stages where false positives have different impact: during preparation, false positives have only negative impact (they decrease cost-effectiveness and increase inspection losses); however, during inspection meeting, teams usually report fewer false positives as compared to the number of false positives they eliminate. Therefore, if a team is good at eliminating false positives during inspection meeting, its cost-effectiveness increases and inspection losses decrease as the influence of false positives grows.

3) Furthermore, we compared the resultant values of preparation stage's metrics against the values of preparation and inspection meeting stage's metrics (see Section 7.7).

The results (see Table 7.14) showed that inspection teams exhibit significantly higher cost-effectiveness during preparation stage as compared to preparation and inspection meeting stages when the influence of false positives on testing costs is low (Cases 0 and 1). The result is significant for metrics M_k, M_{g_IDV} and M_{g_MEET}. Moreover, inspection teams exhibit significantly smaller preparation losses as compared to inspection meeting losses.

As the influence of false positives increases (Cases 3 and 4), cost-effectiveness of inspection teams during preparation stage becomes significantly smaller than during preparation and inspection meeting stages. The

result is significant for metrics M_k, M_{g_IDV} and M_{g_MEET}. Furthermore, meeting losses of preparation stage are significantly higher than those of inspection meeting stages.

4) Finally, we compared metrics values between CBR and PBR teams (see Section 7.8).

The results (see tables E7 and E8 in Appendix E) indicate that there is no significant difference between CBR and PBR with respect to metric M_k.

As for extended cost-effectiveness metric M_{g_IDV} during preparation stage, PBR outperforms CBR when the influence of false positives is high. Furthermore, PBR outperforms CBR during preparation and inspection meeting stages (metric M_{g_MEET}) when the influence of false positives on testing is low. In other cases, there is no significant difference between CBR and PBR with respect to metrics M_{g_IDV} and M_{g_MEET} values.

As for inspection losses (metrics M_{l_IDV} and M_{l_MEET}), during preparation PBR exhibits significantly smaller losses than CBR in the cases when the influence of false positives is high. In all other cases, there is no significant difference between CBR and PBR.

7.10 Conclusions

This chapter presented an experimental evaluation of the new metrics proposed in Chapter 6. The evaluation was made using the inspection data collected from Experiment 2 (see Chapter 4). The resultant values of the new metrics have been compared against the values of Kusumoto's metric M_k. In addition, we investigated the influence of false positives on cost-effectiveness of inspections.

The results of data analysis revealed that:

- There is a strong correlation between the values of metric M_k and extended cost-effectiveness metrics M_{g_IDV} and M_{g_MEET};
- The values of metrics M_{g_IDV} and M_{g_MEET} are significantly smaller as compared to metric M_k;
- False positives have an important influence on metrics values. If a large amount of false positives is introduced during preparation stage and inspection meeting stage is not performed, they may propagate into testing. As the result, inspection will exhibit lower cost-effectiveness and greater losses;

- The decision whether to perform inspection meeting or not depends on the probability of false positives to propagate into testing and to introduce defects, and on the severity of the introduced defects. When the probability that false positives will propagate into testing is low, it is not cost-effective to perform inspection meetings. However, when the probability that false positives will propagate and introduce major defects increases, an inspection that includes both preparation and inspection meeting becomes more cost-effective as compared to an inspection that only includes the preparation stage;
- The reading technique used during inspection has some influence on metrics values. PBR technique outperformed CBR in extended cost-effectiveness during preparation stage when the influence of false positives is high, and during preparation and inspection meeting stages when the influence of false positives on testing is low. As for inspection losses, during preparation PBR exhibits significantly smaller losses than CBR in the cases when the influence of false positives is high.

The analysis presented in this chapter shows that new metrics can be successfully applied to inspections and they enable a more precise evaluation of inspections as compared to conventional metrics. Moreover, it demonstrates the influence of false positives on testing costs.

CHAPTER 8

Conclusions and Future Work

8.1 Summary of Major Results

This thesis has shown the way in which software inspection can be applied to defect detection in object-oriented design. Two reading techniques have been developed and empirically evaluated for inspection of design artifacts written in the notation of Unified Modelling Language. The evaluation was made by means of two controlled experiments in university environment. The main focus of the evaluation was the effectiveness and efficiency of reading techniques, as well as the usefulness of inspection meetings and the influence of false positives on inspection team performance. The results of individual inspector performance indicated that CBR and PBR are effective techniques for inspection of Object-Oriented design, and lead to detection of on average 70% of defects. Furthermore, CBR technique is more efficient than PBR, however individuals who use CBR need more time for inspection as compared to those who use PBR, and there is no difference in effectiveness between individual who use CBR as compared to those who use PBR technique. The analysis of 3-person inspection team performance revealed that CBR and PBR teams exhibit similar effectiveness and efficiency; however team meetings are less beneficial for CBR teams with respect to meeting gains and meeting losses.

In addition, three propositions from the recently proposed behaviourally motivated inspection theory [Sauer et al. 2000] were tested in order to verify the usefulness of the theory in practice. The results of the data analysis supported all three propositions and confirmed that interacting teams do not detect a significant number of new defects during inspection meeting. However, they reported fewer false positives than nominal teams, and finally, nominal teams are more effective than interacting teams.

The second part of the thesis was focused on cost-effectiveness evaluation of inspections. Models to describe inspection costs along with new metrics to evaluate inspection cost-effectiveness and inspection losses were proposed. The metrics consider false positives, and therefore they enable a more precise evaluation of software inspections as compared to the conventional metrics. Moreover, they enable the evaluation of cost-effectiveness in different stages of

inspection. The usefulness of new metrics was evaluated using data collected from a controlled experiment. During evaluation, new metrics were applied together with the cost-effectiveness metric M_k to the data collected from a controlled experiment, and resultant values were compared.

The results of the metrics evaluation indicated that there is a strong correlation between the values of metric M_k and extended cost-effectiveness metrics M_{g_IDV} and M_{g_MEET}. Moreover, the values of metrics M_{g_IDV} and M_{g_MEET} are significantly smaller as compared to metric M_k. This indicates that false positives have an important influence on metrics values, since they reduce cost-effectiveness of inspection. If a large amount of false positives is introduced during the preparation stage and inspection meeting stage is not performed, false positives may propagate into testing and eventually introduce new defects. As a result, inspection will exhibit lower cost-effectiveness and greater losses. The decision whether to perform inspection meeting or not, depends on the probability of false positives to propagate into testing and to introduce defects, and on the severity of such defects. If the probability that false positives will propagate into testing is low, it is not cost-effective to perform inspection meetings. However, as the probability that false positives will propagate and introduce major defects increases, an inspection including both preparation and inspection meeting becomes more cost-effective as compared to an inspection that only includes preparation stage.

The reading techniques used during inspection have some influence on metrics values. PBR technique outperformed CBR in extended cost-effectiveness during preparation stage when the influence of false positives is high, and during the preparation and inspection meeting stages, when the influence of false positives on testing is low. Regarding inspection losses, during preparation PBR exhibits significantly smaller losses than CBR in the cases when the influence of false positives is high.

This thesis contributes to both theory and practice, as follows.

Although several reading techniques have been developed to guide individual inspectors during inspection, there is a lack of techniques for object-oriented artifact inspection. As a result of this thesis, a checklist for inspection using Checklist-based reading technique and three scenarios for inspection using Perspective-based reading technique have been developed. They can be useful for researchers and practitioners to detect defects in design diagrams written using the notation of Unified Modelling Language.

This thesis observes a theoretical lack to explain the inspection. Recently, a theory focused on the behavioural aspects of individual and group work during inspection has been proposed [Sauer et al. 2000]. However, some of the propositions of this theory have never been tested, while others have shown mixed results. This thesis contributes to validation of this theory by testing some of its propositions.

In addition to true defects, false positives are often being reported during inspections. They do not add to the quality of software, because the rework of false positives is costly and may introduce new defects. Nevertheless, there is a lack of research in this area. This thesis contributes to the evaluation of impact of false positives on the performance of individual inspectors as well as inspector teams.

In order to evaluate the cost-effectiveness of inspections, several metrics have been previously proposed, however none of them considers false positives. To tackle this problem, this thesis presents two cost models to describe the impact of false positives on the inspection and testing costs. In addition, it proposes four new metrics for evaluating cost-effectiveness of inspection as well as inspection losses, considering false positives reported during inspection.

This thesis may be beneficial for other researchers and practitioners to utilize and evaluate inspections in their projects and organizations.

Overall, the reading techniques, cost models and metrics proposed in this thesis may facilitate the work of researchers and practitioners when utilizing and evaluating software inspection.

8.2 Directions for Future Research

From the work carried out in this thesis there are several issues that require further investigation.

(1) *Further experimental evaluation and refinement of reading techniques*. The experimental investigations described in this thesis were conducted in university environment, in which the third year students were used as subjects. In order to explore the validity of the results presented in this thesis, replications should be conducted. Moreover, to generalize the results of the experiments, the reading techniques presented in this thesis should be applied in an industrial environment. Furthermore, since the controlled experiments described in this thesis did not reveal significant difference in

effectiveness between CBR and PBR, these techniques have to be further refined.

(2) *Further evaluation of new metrics*. The evaluation of metrics proposed in this thesis has been made using the data of an experimental investigation, which included inspection data, but did not include testing data. Since testing data is required in order to compute metrics values, the values of testing have been calculated using the data collected from inspection. However, to enable a more precise evaluation of the new metrics, an experiment including both inspection and testing should be conducted.

(3) *Extension of new metrics*. In new metrics, we did not distinguish between inspection of different software artifacts such as requirements specifications, design, code, etc. Therefore, it is essential to collect data from inspection of different artifacts, and to investigate how it influences the overall cost-effectiveness of inspection. Furthermore, in this thesis we assumed that no defects remain in software after testing. However, in practice this not always occurs. Therefore, it is important to collect data about defects remaining after testing, and to apply it in order to extend the proposed metrics.

BIBLIOGRAPHY

[Ackerman et al. 1989] A.F. Ackerman, L.S. Buchwald, F. H. Lewski, Software inspections: An effective verification process, IEEE Software 6 (3) (1989) 31-36.

[Andersson et al. 2003] C. Andersson, T. Thelin, P. Runeson, N. Dzamashvili, An Experimental Evaluation of Inspection and Testing for Detection of Design Faults, Proceedings of the 2003 International Symposium on Empirical Software Engineering (2003) 174-184.

[Basili et al. 1994] V.R. Basili, G.Caldiera, H.D. Rombach, Goal Question Metric Paradigm, Encyclopedia of Software Engineering, J.J. Marciniak (ed.), John Wiley & Sons, New York, 1994.

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

[Basili et al. 1996b] V.R. Basili, G. Caldiera, F. Lanubile, and F. Shull, Studies on Reading Techniques, Proceedings of the Twenty-First Annual Software Engineering Workshop, Goddard Space Flight Center, USA (1996).

[Basili et al. 1999] V.R. Basili, F. Shull, F. Lanubile, Building Knowledge through Families of Experiments, IEEE Transactions of Software Engineering 25 (4) (1999) 456-473.

[Biffl & Gutjahr 2001] S.Biffl, W. Gutjahr, Influence of Team Size and Defect Detection Methods on Inspection Effectiveness, Proceedings of IEEE International Software Metrics Symposium, London, UK, (2001) 63-75.

[Biffl et al. 2001] S. Biffl, B. Freimut, O. Laitenberger, Investigating the Cost-Effectiveness of Reinspection in Software Development, Proceedings of the 23rd International Conference on Software Engineering (2001) 155-164.

[Biffl & Halling 2002] S. Biffl, M. Halling, Investigating the Influence of Inspector Capability Factors with Four Inspection Techniques on Inspection Performance, Proceedings of the Eighth IEEE Symposium on Software

Metrics (2002) 107-117.

[Biffl & Halling 2003] S. Biffl, M. Halling, Investigating the Defect Detection Effectiveness and Cost-Benefit of Nominal Inspection Teams, IEEE Transactions on Software Engineering, 29 (5) (2003) 385-397.

[Booch et al. 1999] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Modeling Language User Guide, Addison Wesley Longman, Inc, 1999.

[Briand et al. 1998] L. Briand, K. El Emam, O. Laitenberger, T. Fussbroich, Using Simulation to Build Inspection Efficiency Benchmarks for Development Projects, Proceedings of the 20th International Conference on Software Engineering, Kyoto, Japan, (1998) 340-349.

[Briand et al. 1999] L. Briand, B. Freimut, F. Vollei, Assessing the Cost-Effectiveness of Inspections by Combining Project Data and Expert Opinion, International Software Engineering Research Network (ISERN), ISERN-99-14 Version 2, 1999.

[Broy & Denert 2002] M. Broy, E. Denert (Eds.), Software Pioneers: Contributions to Software Engineering, Springer-Verlag, 2002.

[Cheng & Jeffery 1996] B. Cheng, R. Jeffery, Comparing Inspection Strategies for Software Requirements Specifications, Proceedings of the 1996 Australian Software Engineering Conference (1996) 203-211.

[Chernak 1996] Y. Chernak, A Statistical Approach to the Inspection Checklist Formal Synthesis and Improvement, IEEE Transactions on Software Engineering 22 (12) (1996) 866-874.

[Collofello & Woodfield 1989] J.S. Collofello, S.N. Woodfield, Evaluating the Effectiveness of Reliability-Assurance Techniques, Journal of Systems and Software 9 (3) (1989) 191-195.

[Conradi et al. 1999] R. Conradi, A.S. Marjara, B. Skåtevik, Empirical Studies of Inspection and Testing Data, Proceedings of International Conference on Product Focused Software Process Improvement, Oulu, Finland (1999) 263-284.

[Dunsmore et al. 2001] A. Dunsmore, M. Roper, M. Wood, Systematic Object-Oriented Inspection - An Empirical Study, Proceedings of the 23rd International Conference on Software Engineering (2001) 135-144.

[Dunsmore et al. 2002] A. Dunsmore, M. Roper, M. Wood, Further Investigations into the Development and Evaluation of Reading Techniques for Object-Oriented Code Inspection, Proceedings of the 24th International Conference on Software Engineering (2002) 47-57.

[Dunsmore 2002] A. Dunsmore, Investigating Effective Inspection of Object-Oriented Code, PhD thesis, University of Strathclyde, Glasgow, UK, 2002.

[ESEG] Procedural Techniques for Perspective-Based Reading of Requirements and Object-Oriented Designs, Lab package of The Experimental Software Engineering Group (ESEG) of the University of Maryland, http://www.cs.umd.edu /projects/SoftEng/ESEG.

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

[Fagan 1986] M. Fagan, Advances in Software Inspection, IEEE Transactions on Software Engineering 12 (7) (1986) 744-751.

[Fusaro et al. 1997] P. Fusaro, F. Lanubile, G. Visaggio, A Replicated Experiment to Assess Requirements Inspection Techniques, Empirical Software Engineering: An International Journal 2 (1) (1997) 39-57.

[Gilb & Graham 1993] T. Gilb, D. Graham, Software Inspection, Addison-Wesley, 1993.

[Hetzel 1998] B. Hetzel, The Complete Guide to Software Testing, John Wiley & Sons, USA, 1988.

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

[IEEE 1989] IEEE Standard for Software Reviews and Audits. IEEE Standards Collection. Software Engineering. 1997 Edition. IEEE, 1989.

[Itoh et al. 2001] K. Itoh, T. Hirota, T. Fuji, S. Kumagai, R. Kawabata, Software Engineering Exercises, Ohmsha, 2001. (in Japanese)

[Jeffery & Scott 2002] R. Jeffery, L. Scott, Has Twenty-five Years of Empirical

Software Engineering Made a Difference?, Proceedings of the Asia-Pacific Software Engineering Conference, Gold Coast, Australia, 2002, IEEE Computer Society, Los Alamitos, California, USA (2002) 539-546.

[Johnson & Tjahjono 1998] P.M. Johnson, D. Tjahjono, Does Every Inspection Really Need a Meeting?, Empirical Software Engineering: An International Journal 3 (1) (1998) 9-35.

[Juristo & Moreno 2001] N. Juristo, A.M. Moreno, Basics of Software Engineering Experimentation, Kluwer Academic Publishers, 2001.

[Kusumoto et al. 1992] S. Kusumoto, K. Matsumoto, T. Kikuno, K. Torii, A New Metrics for Cost Effectiveness of Software Reviews, IEICE Transactions on Information and Systems E75-D (5) (1992) 674-680.

[Kusumoto 1993] S. Kusumoto, Quantitative Evaluation of Software Reviews and Testing Processes, PhD Dissertation, Osaka University, 1993.

[Laitenberger & Atkinson 1999] O. Laitenberger, C. Atkinson, Generalizing Perspective-based Inspection to handle Object-Oriented Development Artefacts, Proceedings of the 21st International Conference on Software Engineering (1999) 494-503.

[Laitenberger & DeBaud 2000] O. Laitenberger, J.M. DeBaud, An Encompassing Life Cycle Centric Survey of Software Inspection, The Journal of Systems and Software 50 (1) (2000) 5-31.

[Laitenberger et al. 2000] O. Laitenberger, C. Atkinson, M. Schlich, K. El Emam, An Experimental Comparison of Reading Techniques for Defect Detection in UML Design Documents, The Journal of Systems and Software 53 (2000) 183-204.

[Laitenberger 2000] O. Laitenberger, Cost-Effective Detection of Software Defects through Perspective-Based Inspection, PhD thesis, University of Kaiserslautern, Germany, 2000.

[Laitenberger et al. 2001] O. Laitenberger, K. El Emam, T.G. Harbich, An Internally Replicated Quasi-Experimental Comparison of Checklist and Perspective-Based Reading of Code Documents, IEEE Transactions on Software Engineering 27 (5) (2001) 387-421.

[Land et al. 1997a] L.P.W. Land, C. Sauer, R. Jeffery, Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews: Report of a Laboratory Experiment Using Program Code, Proceedings of the sixth European Software Engineering Conference held jointly with the fifth ACM SIGSOFT Symposium on the Foundations of Software Engineering, Zurich, Switzerland, Lecture Notes in Computer Science 1301, Springer-Verlag, Berlin (1997) 294-307.

[Land et al. 1997b] L.P.W. Land, R. Jeffery, C. Sauer, Validating the Defect Detection Performance Advantage of Group Designs for Software Reviews: Report of Replicated Experiment, Proceedings of the Australian Software Engineering Conference, Sydney, Australia (1997) 17-26.

[Land 2000] L.P.W. Land, Software Group Review and the Impact of Procedural Roles on Defect Detection Performance, PhD Dissertation, University of New South Wales, 2000.

[Lanubile & Visaggio 1996] F. Lanubile, G. Visaggio, Assessing Defect Detection Methods of Software Requirements Inspections Through External Replication, Technical report ISERN-96-01, International Software Engineering Research Network, 1996.

[McCarthy et al. 1996] P. McCarthy, A. Porter, S. Harvey, L. Votta, An experiment to assess cost-benefits of inspection meetings and their alternatives: A pilot study, *Proceedings of Third International Software Metrics Symposium*, Berlin, Germany, (1996) 25-26.

[Myers 1978] G.J. Myers, A Controlled Experiment in Program Testing and Code Walkthroughs / Inspections, Communications of the ACM 21 (9) (1978) 760-768.

[Norŭsis 1995] M.J. Norŭsis, SPSS: SPSS 6.1 Guide to Data Analysis, Prentice Hall, 1995.

[Parnas 2003] D.L. Parnas, The Limits of Empirical Studies of Software Engineering, Proceedings of the 2003 International Symposium on Empirical Software Engineering (2003) 2-5.

[Parnas & Lawford 2003] D.L. Parnas, M. Lawford, The Role of Inspection in Software Quality Assurance, IEEE Transactions on Software Engineering

29 (8) (2003) 674-676.

[Porter & Votta 1994] A.A. Porter, L.G. Votta, An Experiment to Assess Different Detection Methods for Software Requirements Inspection, Proceedings of the 16th International Conference on Software Engineering, Sorrento, Italy (1994) 103-112.

[Porter et al. 1995] A.A. Porter, L.G. Votta, V. Basili, Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment, IEEE Transactions on Software Engineering 21 (6) (1995) 563-575.

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

[Porter et al. 1997] A.A. Porter, H.P. Siy, C.A. Toman, L.G. Votta, An experiment to assess the cost-benefits of code inspections in large scale software development, IEEE Transactions on Software Engineering 23 (6) (1997) 329-346.

[Regnell et al. 2000] B. Regnell, P. Runeson, T. Thelin, Are the Perspectives Really Different? Further Experimentation on Scenario-Based Reading of Requirements, Empirical Software Engineering: An International Journal 5 (4) (2000) 331-356.

[Sabaliauskaite et al. 2002] G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, K. Inoue, An Experimental Comparison of Checklist-Based Reading and Perspective-Based Reading for UML Design Document Inspection, Proceedings of the 2002 International Symposium on Empirical Software Engineering (2002) 148-157.

[Sabaliauskaite et al. 2003] G. Sabaliauskaite, F. Matsukawa, S. Kusumoto, K. Inoue, Further Investigations of Reading Techniques for Object-Oriented Design Inspection, Information and Software Technology 45 (9) (2003) 571-585.

[Sauer et al. 2000] C. Sauer, R. Jeffery, L. Land, P. Yetton, The Effectiveness of Software Development Technical Reviews: a Behaviorally Motivated Program of Research, IEEE Transactions on Software Engineering 26 (1) (2000) 1-14.

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

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

[Thelin 2002] T. Thelin, Empirical Evaluations of Usage-Based Reading and Fault Content Estimation for Software Inspections, PhD Thesis, Lund University, 2002.

[Travassos et al. 1999a] G. Travassos, F. Shull, M. Fredericks, V. Basili, Detecting Defects in Object Oriented Designs: Using Reading Techniques to Increase Software Quality, Proceedings of the 1999 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages & Applications (1999) 47-56.

[Travassos et al. 1999b] G. Travassos, F. Shull, J. Carver, V.R. Basili, Reading Techniques for OO Design Inspections, Proceedings of the 24th Annual Software Engineering Workshop, NASA Goddard Space Flight Center (1999).

[Votta 1993] L.G. Votta Jr, Does Every Inspection Need a Meeting?, Proceedings of the 1993 ACM SIGSOFT Symposium on Foundations of Software Engineering, ACM Software Engineering Notes 18 (5) (1993) 107-114.

[Wohlin et al. 2000] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A.Wesslen, Experimentation in Software Engineering: an Introduction, Kluwer Academic Publishers, 2000.

[Wohlin et al. 2002] C. Wohlin, A. Aurum, H. Petersson, F. Shull, M. Ciolkowski, Software Inspection Benchmarking – A Qualitative and Quantitative Comparative Opportunity, Proceedings of the Eighth IEEE Symposium on Software Metrics (2002) 118-127.

APPENDICES

APPENDIX A: CBR CHECKLIST, PBR SCENARIOS AND INDIVIDUAL DEFECT REGISTRATION FORM USED IN EXPERIMENT 1

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 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 software code development?	☐ yes ☐ no
Activity Diagrams		
6.	Aren't there any inconsistencies between Activity Diagrams and Requirements Specification?	☐ yes ☐ no
7.	Are all the necessary States and Transitions defined?	☐ yes ☐ no
8.	Is the order of the States correct?	☐ yes ☐ no
9.	Are there no redundant elements in Activity Diagrams?	☐ yes ☐ no
Sequence Diagrams		
10.	Aren't there any inconsistencies between 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
Component Diagrams		
15.	Aren't there any inconsistencies between Component Diagrams and other diagrams?	☐ yes ☐ no
16.	Are all the Classes from Class Diagrams included in one of Components?	☐ yes ☐ no
17.	Are the necessary Software Components and their relationships depicted?	☐ yes ☐ no
18.	Are there no redundant elements in Component Diagrams?	☐ yes ☐ no
19.	Do Component Diagrams implement all use cases of the system from Use-Case Diagrams?	☐ yes ☐ no
20.	Do Component Diagrams have enough information for implementation?	☐ yes ☐ no

Figure A1. CBR Checklist.

USER'S SCENARIO		
Assume that you are a User of the system. 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 user requirements. It means that there must be no inconsistencies between the various analysis models, such as Requirements specification, Use-Case, Activity and Sequence diagrams. Please perform tasks following Step1 through Step5. For each step you must locate corresponding documents, follow the instructions, perform the necessary tasks and answer the given questions. Please perform the tasks in the Comment form. In addition, if you have any comments, write them in the Comment form as well. When you detect a defect, mark it on the diagram and fill the necessary information in the Defect registration form.		
Step 1	Locate Activity Diagrams	
	Please examine each Activity Diagram carefully and answer the following questions.	
	1.1.	Is every State associated with at least one another State?
	1.2.	Is the name of every State defined?
Step 2	Locate Activity Diagrams and Requirements Specification	
	Activity Diagrams must conform to user requirements, described in Requirements Specification. Compare Activity Diagrams to Requirements Specification and answer the following questions.	
	2.1.	Do Activity Diagrams conform to system requirements, described in Requirements Specification?
	2.2.	Is the order of States and Transitions correct?
	2.3.	Are there no missing of redundant elements in Activity Diagrams?
Step 3	Locate Sequence Diagrams	
	Sequence Diagrams show interactions between Objects in the system. Those interactions take place in specific sequence at appropriate time. Check the Sequence Diagrams and answer the following questions.	
	3.1.	Does every Object have at least one Message sent or received?
	3.2.	Are the names of every Object and every Message defined?
Step 4	Locate Sequence Diagrams and Requirements Specification	
	Sequence Diagrams must conform to system requirements, defined in Requirements Specification. Please make a list of Objects, mentioned in Sequence Diagrams. After that, answer the following questions.	
	4.1.	Are all the Objects of Sequence Diagrams related to the problem domain described in Requirements Specification?
	4.2.	Aren't there any inconsistencies between Sequence Diagrams and the Requirements Specification?
Step 5	Locate Sequence Diagrams and Use-Case Diagrams	
	Sequence Diagrams need to satisfy the behavior of the system, described in Use-Case diagrams. Please write the names of the Sequence diagrams, which correspond to each of Use Cases, on the Use-Case diagram, next to each Use Case. After that, answer the following questions.	
	5.1.	Is every Use Case of Use-Case Diagram implemented in any of Sequence Diagrams?
	5.2.	Are all the important Objects and Messages between Objects shown in Sequence Diagrams?

Figure A2. User's Scenario.

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. For each step you must locate corresponding documents, follow the instructions, perform the necessary tasks and answer the given questions. Please perform the tasks in the Comment form. In addition, if you have any comments, write them in the Comment form as well. When you detect a defect, mark it on the diagram and fill the necessary information in the Defect registration form.

Step 1	Locate Class Diagrams	
	Examine Class Diagrams and answer the following questions.	
	1.1.	Is the name of every Class defined?
	1.2.	Is the multiplicity of all Associations defined?
Step 2	Locate Class Diagrams, Requirements Specification and Use-Case Diagrams	
	Class Diagram needs to show all the necessary Object Classes, their Attributes, Methods and Associations between Classes. Compare Class Diagrams with Requirements Specification and Use-Case Diagrams to make sure that there are no inconsistencies among them. Make a list of Objects mentioned in the Requirements Specification. After that, answer the following questions.	
	2.1.	Are all the listed objects from Requirements Specification represented in Class Diagrams?
	2.2.	Are there no redundant elements in Class diagram?
	2.3.	Are all the important Classes and Associations defined?
	2.4.	Are all the Classes, necessary to perform Use Cases from Use-Case Diagrams, defined in Class Diagrams?
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 the 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 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?

Figure A3. Designer's Scenario.

IMPLEMENTER'S SCENARIO

Assume that you are an Implementer of the system. 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. Please perform tasks following Step1 through Step4. For each step you must locate corresponding documents, follow the instructions, perform the necessary tasks and answer the given questions. Please perform the tasks in the Comment form. In addition, if you have any comments, write them in the Comment form as well. When you detect a defect, mark it on the diagram and fill the necessary information in the Defect registration form.

Step 1	Locate Class Diagrams, Use Case Diagrams and Requirements Specification	
	Examine Class diagram in order to determine if it reflects user requirements from Requirements Specification, and if it is sufficient to perform Use Cases from Use-Case diagram. Please write the names of the Object Classes from Class Diagrams, which correspond to each of Use Cases, on the Use-Case diagram, next to each Use Case. Please check if the Classes written next to each Use Case have associations among them in Class Diagrams. After that, answer the following questions.	
	1.1.	Are all the Classes, necessary to realize the functions of Use Cases, defined in Class Diagrams?
	1.2.	Do all the necessary Associations between Classes exist?
	1.3.	Is the multiplicity of all Associations defined?
	1.4.	Do Class diagrams have enough information for software code development?
Step 2	Locate Component Diagrams and Class Diagrams	
	Component Diagrams show the relationships between software components in the system. Each Component can implement one of more Object Classes. Please examine the Component diagrams, and for each Component make a list of Classes from Class Diagram, which should be implementer by that Component. After that, answer the following questions.	
	2.1.	Are all the Classes from Class Diagrams included into one of software Components from Component Diagrams?
	2.2.	Are all the necessary Components defined?
	2.3.	Are all the necessary relationships between Components defined?
	2.4.	Are there no redundant Components or relationships?
Step 3	Locate Component Diagrams and Use-Case Diagrams	
	Compare Component Diagrams and Use-Case diagrams in order to detect any inconsistencies between them and answer the following questions.	
	3.1.	Are Component Diagrams sufficient to perform all the Use Cases from Use-Case diagrams?
	3.2.	Are Component Diagrams complete and ready to proceed with code development?
Step 4	Locate Class Diagrams and Sequence Diagrams	
	Compare Class Diagrams and Sequence Diagrams in order to detect any inconsistencies between them and answer the following questions.	
	4.1.	Are there no inconsistencies between Class Diagrams and Sequence Diagrams?
	4.2.	Are all the necessary interactions between Objects defined in the Sequence Diagrams?
	4.3.	Are the Class and Sequence Diagrams complete and ready to proceed with code development?

Figure A4. Implementer's Scenario.

Student:					Inspection start time:	
Defect No	UML diagram				Checklist/Scenario item	Detection time
	Activity	Class	Sequence	Component		
1						
2						
3						
…						
30						
					Inspection end time:	
1. Did you follow the instructions of Checklist/Scenario?						
2. How many percent of defects do you think you have found?						

Figure A5. Individual defect registration form.

APPENDIX B: DATA COLLECTED FROM EXPERIMENT 1

Table B1. Individual inspector data collected from Experiment 1

Subject No	System H(Hospital)/ S(Seminar)	Method CBR/PBR	PBR perspective	Detected defects	False positives	Inspection time (min)
1	H	CBR		11	10	61
2	H	CBR		10	8	73
3	H	CBR		12	4	67
4	H	CBR		10	9	61
5	H	CBR		8	6	60
6	H	CBR		11	10	92
7	H	CBR		8	14	67
8	H	CBR		11	4	94
9	H	CBR		12	5	66
10	H	CBR		12	7	60
11	H	PBR	U	3	8	53
12	H	PBR	U	6	3	61
13	H	PBR	U	5	1	70
14	H	PBR	U	3	2	29
15	H	PBR	U	5	1	25
16	H	PBR	U	4	3	55
17	H	PBR	U	5	2	45
18	H	PBR	D	4	7	70
19	H	PBR	D	4	3	30
20	H	PBR	D	5	5	73
21	H	PBR	D	3	3	50
22	H	PBR	D	4	3	60
23	H	PBR	D	3	3	72
24	H	PBR	I	6	5	65
25	H	PBR	I	7	6	77
26	H	PBR	I	5	0	60
27	H	PBR	I	6	7	76
28	H	PBR	I	7	5	56
29	H	PBR	I	7	5	44
30	S	CBR		8	7	71
31	S	CBR		13	3	67
32	S	CBR		13	6	78
33	S	CBR		10	5	69
34	S	CBR		13	8	62
35	S	CBR		9	7	78
36	S	CBR		9	6	79
37	S	CBR		10	6	70
38	S	CBR		8	3	80
39	S	CBR		12	9	77
40	S	CBR		11	7	90

41	S	PBR	U	6	10	67
42	S	PBR	U	3	4	46
43	S	PBR	U	4	4	50
44	S	PBR	U	5	1	50
45	S	PBR	U	6	9	50
46	S	PBR	U	4	10	70
47	S	PBR	U	3	8	90
48	S	PBR	D	5	9	60
49	S	PBR	D	5	7	51
50	S	PBR	D	5	6	59
51	S	PBR	D	4	6	80
52	S	PBR	D	6	1	68
53	S	PBR	D	5	7	75
54	S	PBR	I	6	10	81
55	S	PBR	I	7	12	83
56	S	PBR	I	9	14	83
57	S	PBR	I	5	2	78
58	S	PBR	I	7	3	40
59	S	PBR	I	5	7	95

APPENDIX C: TEAM DEFECT REGISTRATION FORM USED IN EXPERIMENT 2

Student 1: Student 2: Student 3:					Inspection meeting start time:		
Defect No	UML diagram				Student who detected defect during preparation		
	Activity	Class	Sequence	Component	Student 1	Student 2	Student 3
1							
2							
3							
…							
30							
					Inspection meeting end time:		

Figure C1. Team defect registration form.

APPENDIX D: DATA COLLECTED FROM EXPERIMENT 2

Table D1. Individual inspector data collected from Experiment 2

Subject No	System H/S	Method CBR/PBR	PBR perspective	Detected defects	Major defects	Minor defects	False positives	Time spent (min)
1	H	CBR		7	2	5	5	60
2	H	CBR		8	3	5	2	50
3	H	CBR		5	2	3	6	57
4	H	CBR		8	2	6	6	55
5	H	CBR		7	5	2	5	60
6	H	CBR		6	3	3	6	60
7	H	CBR		8	4	4	2	47
8	H	CBR		3	2	1	2	57
9	H	CBR		9	3	6	5	59
10	H	CBR		6	3	3	6	60
11	H	CBR		5	1	4	6	60
12	H	CBR		7	3	4	7	53
13	H	PBR	U	4	2	2	5	55
14	H	PBR	U	5	2	3	3	47
15	H	PBR	U	3	1	2	1	46
16	H	PBR	U	7	3	4	1	60
17	H	PBR	U	6	3	3	1	56
18	H	PBR	D	2	1	1	3	57
19	H	PBR	D	3	1	2	4	57
20	H	PBR	D	3	2	1	3	47
21	H	PBR	D	3	1	2	6	56
22	H	PBR	D	2	0	2	5	48
23	H	PBR	I	5	4	1	2	56
24	H	PBR	I	3	2	1	5	58
25	H	PBR	I	6	4	2	2	59
26	H	PBR	I	5	3	2	3	53
27	H	PBR	I	3	2	1	5	58
28	S	CBR		3	1	2	5	50
29	S	CBR		7	4	3	3	60
30	S	CBR		6	3	3	7	60
31	S	CBR		4	2	2	3	56
32	S	CBR		6	2	4	9	60
33	S	CBR		6	2	4	4	60
34	S	CBR		6	3	3	5	51
35	S	CBR		3	2	1	6	60
36	S	CBR		6	3	3	3	54
37	S	CBR		10	5	5	9	50
38	S	CBR		7	2	5	4	60
39	S	CBR		8	3	5	8	60
40	S	CBR		3	0	3	1	55
41	S	CBR		5	3	2	4	60

42	S	CBR		4	1	3	7	60
43	S	PBR	U	2	0	2	1	56
44	S	PBR	U	7	3	4	6	56
45	S	PBR	U	4	1	3	3	55
46	S	PBR	U	5	2	3	3	56
47	S	PBR	D	4	2	2	4	57
48	S	PBR	D	5	2	3	2	53
49	S	PBR	D	4	2	2	4	56
50	S	PBR	D	2	1	1	3	56
51	S	PBR	I	5	3	2	4	50
52	S	PBR	I	2	2	0	3	56
53	S	PBR	I	3	2	1	4	59
54	S	PBR	I	3	1	2	2	56

Table D2. Inspection meeting data collected from Experiment 2

Team No.	System H/S	Method	Detect. defects	Major defects	Minor defects	New defects found	New major defects	New minor defects	Defects lost	Lost major defects	Lost minor defects	False positives	New false positives	Eliminated false positives	Team meeting time (min)
T1	H	CBR	9	2	7	0	0	0	1	1	0	7	1	4	66
T2	H	CBR	7	2	5	1	0	1	4	3	1	6	0	8	75
T3	H	CBR	9	3	6	0	0	0	3	2	1	5	1	3	81
T4	H	CBR	10	5	5	0	0	0	1	0	1	8	2	10	69
T5	H	PBR	7	4	3	0	0	0	1	1	0	3	1	7	60
T6	H	PBR	8	5	3	1	1	0	1	0	1	7	2	5	69
T7	H	PBR	8	4	4	0	0	0	2	2	0	3	2	5	30
T8	H	PBR	10	5	5	0	0	0	0	0	0	4	0	3	66
T9	H	PBR	11	6	5	3	2	1	1	0	1	3	0	7	66
T10	S	CBR	6	2	4	0	0	0	4	4	0	4	1	12	105
T11	S	CBR	6	2	4	0	0	0	3	3	0	1	1	14	75
T12	S	CBR	7	4	3	0	0	0	1	1	0	3	1	9	69
T13	S	CBR	8	3	5	0	0	0	3	2	1	7	0	12	90
T14	S	CBR	7	3	4	0	0	0	1	1	0	4	3	11	90
T15	S	PBR	6	2	4	0	0	0	2	2	0	1	0	7	69
T16	S	PBR	6	3	3	0	0	0	3	1	2	3	0	6	69
T17	S	PBR	7	3	4	0	0	0	0	0	0	3	1	9	60
T18	S	PBR	6	3	3	0	0	0	2	0	2	6	1	3	60

APPENDIX E: DATA USED FOR METRICS EVALUATION AND EVALUATION RESULTS

Table E1. Data of preparation stage of inspection

Team	Method	Number of defects detected by inspectors			Number of false positives *FP*	Number of undetected defects after preparation			Preparation cost *Cr* (min)
		Total *D_Tot*	Major defects *D_Maj*	Minor defects *D_Min*		Total *UD_Tot*	Major defects *UD_Maj*	Minor defects *UD_Min*	
T1	CBR	10	3	7	10	4	3	1	227
T2	CBR	10	5	5	14	4	1	3	235
T3	CBR	12	5	7	7	2	1	1	223
T4	CBR	11	5	6	16	3	1	2	233
T5	PBR	8	5	3	9	6	1	5	228
T6	PBR	9	5	4	10	5	1	4	222
T7	PBR	10	6	4	6	4	0	4	212
T8	PBR	10	5	5	7	4	1	3	229
T9	PBR	9	4	5	10	5	2	3	222
T10	CBR	10	6	4	15	3	0	3	230
T11	CBR	9	5	4	14	4	1	3	236
T12	CBR	8	5	3	11	5	1	4	225
T13	CBR	11	5	6	19	2	1	1	230
T14	CBR	8	4	4	12	5	2	3	235
T15	PBR	8	4	4	8	5	2	3	223
T16	PBR	9	4	5	9	4	2	2	225
T17	PBR	7	3	4	11	6	3	3	230
T18	PBR	8	3	5	8	5	3	2	228

Table E2. Data necessary for preparation stage metrics calculation

Team	Method	CrDEF	CrFP	(Ct	CtDEF	CtFP				
						Case 0	Case 1	Case 2	Case 3	Case 4
T1	CBR	113.500	113.500	1860	1500	0	300	600	2400	4800
T2	CBR	97.917	137.083	2700	660	0	420	840	3360	6720
T3	CBR	140.842	82.158	2820	540	0	210	420	1680	3360
T4	CBR	94.926	138.074	2760	600	0	480	960	3840	7680
T5	PBR	107.294	120.706	2580	780	0	270	540	2160	4320
T6	PBR	105.158	116.842	2640	720	0	300	600	2400	4800
T7	PBR	132.500	79.500	3120	240	0	180	360	1440	2880
T8	PBR	134.706	94.294	2700	660	0	210	420	1680	3360
T9	PBR	105.158	116.842	2220	1140	0	300	600	2400	4800
T10	CBR	92.000	138.000	3120	180	0	450	900	3600	7200
T11	CBR	92.348	143.652	2640	660	0	420	840	3360	6720
T12	CBR	94.737	130.263	2580	720	0	330	660	2640	5280
T13	CBR	84.333	145.667	2760	540	0	570	1140	4560	9120
T14	CBR	94.000	141.000	2160	1140	0	360	720	2880	5760
T15	PBR	111.500	111.500	2160	1140	0	240	480	1920	3840
T16	PBR	112.500	112.500	2220	1080	0	270	540	2160	4320
T17	PBR	89.444	140.556	1680	1620	0	330	660	2640	5280
T18	PBR	114.000	114.000	1740	1560	0	240	480	1920	3840

Table E3. Data of preparation and inspection meeting stages

Team	Method	Number of defects detected by a team			Number of additional defects detected during team meeting			Number of defects lost during meeting			Number of undetected defects after inspection meeting (excluding defects lost during meeting)			Number of false positives (FP)		Number of FP eliminated during team meeting *EFP*	Time spent on team meeting *Cm* (min)
		Total *TD_Tot*	Major defects *TD_Maj*	Minor defects *TD_Min*	Total *AD_Tot*	Major defects *AD_Maj*	Minor defects *AD_Min*	Total *LD_Tot*	Major defects *LD_Maj*	Minor defects *LD_Min*	Total *TL_Tot*	Major defects *TL_Maj*	Minor defects *TL_Min*	Total *TFP*	New *AFP*		
T1	CBR	9	2	7	0	0	0	1	1	0	4	3	1	7	1	4	66
T2	CBR	7	2	5	1	0	1	4	3	1	3	1	2	6	0	8	75
T3	CBR	9	3	6	0	0	0	3	2	1	2	1	1	5	1	3	81
T4	CBR	10	5	5	0	0	0	1	0	1	3	1	2	8	2	10	69
T5	PBR	7	4	3	0	0	0	1	1	0	6	1	5	3	1	7	60
T6	PBR	8	5	3	1	1	0	2	1	1	4	0	4	7	2	5	69
T7	PBR	8	4	4	0	0	0	2	2	0	4	0	4	3	2	5	30
T8	PBR	10	5	5	0	0	0	0	0	0	4	1	3	4	0	3	66
T9	PBR	11	6	5	3	2	1	1	0	1	2	0	2	3	0	7	66
T10	CBR	6	2	4	0	0	0	4	4	0	3	0	3	4	1	12	105
T11	CBR	6	2	4	0	0	0	3	3	0	4	1	3	1	1	14	75
T12	CBR	7	4	3	0	0	0	1	1	0	5	1	4	3	1	9	69
T13	CBR	8	3	5	0	0	0	3	2	1	2	1	1	7	0	12	90
T14	CBR	7	3	4	0	0	0	1	1	0	5	2	3	4	3	11	90
T15	PBR	6	2	4	0	0	0	2	2	0	5	2	3	1	0	7	69
T16	PBR	6	3	3	0	0	0	3	1	2	4	2	2	3	0	6	69
T17	PBR	7	3	4	0	0	0	0	0	0	6	3	3	3	1	9	60
T18	PBR	6	3	3	0	0	0	2	0	2	5	3	2	6	1	3	60

Table E4. Data necessary for preparation and inspection meeting stages' metrics calculation (1)

Team	Method	*Cr*	*CmDEF*	*CmFP*	*CmADD_DEF*	*CmADD_FP*	*CmLOST_DEF*	*CmELIM_FP*	*Cm*	*CtDEF*	*CtLOST_DEF*
T1	CBR	227	27.000	21.000	0.000	3.000	3.000	12.000	66	1500	480
T2	CBR	235	20.192	17.308	2.885	0.000	11.538	23.077	75	600	1500
T3	CBR	223	34.714	19.286	0.000	3.857	11.571	11.571	81	540	1020
T4	CBR	233	22.258	17.806	0.000	4.452	2.226	22.258	69	600	60
T5	PBR	228	22.105	9.474	0.000	3.158	3.158	22.105	60	780	480
T6	PBR	222	22.080	19.320	2.760	5.520	5.520	13.800	69	240	540
T7	PBR	212	12.000	4.500	0.000	3.000	3.000	7.500	30	240	960
T8	PBR	229	38.824	15.529	0.000	0.000	0.000	11.647	66	660	0
T9	PBR	222	29.040	7.920	7.920	0.000	2.640	18.480	66	120	60
T10	CBR	230	23.333	15.556	0.000	3.889	15.556	46.667	105	180	1920
T11	CBR	236	18.000	3.000	0.000	3.000	9.000	42.000	75	660	1440
T12	CBR	225	23.000	9.857	0.000	3.286	3.286	29.571	69	720	480
T13	CBR	230	24.000	21.000	0.000	0.000	9.000	36.000	90	540	1020
T14	CBR	235	24.231	13.846	0.000	10.385	3.462	38.077	90	1140	480
T15	PBR	223	25.875	4.313	0.000	0.000	8.625	30.188	69	1140	960
T16	PBR	225	23.000	11.500	0.000	0.000	11.500	23.000	69	1080	600
T17	PBR	230	21.000	9.000	0.000	3.000	0.000	27.000	60	1620	0
T18	PBR	228	20.000	20.000	0.000	3.333	6.667	10.000	60	1560	120

Table E5. Data necessary for preparation and inspection meeting stages' metrics calculation (2)

Team	Method	*ΔCtADD_DEF*	*ΔCtDEF*	Case 0					Case 1				
				CtADD_FP	*CtFP*	*ΔCtELIM_FP*	*Ct*	*ΔCt*	*CtADD_FP*	*CtFP*	*ΔCtELIM_FP*	*Ct*	*ΔCt*
T1	CBR	0	1380	0	0	0	1980	1380	30	180	120	2190	1500
T2	CBR	60	1200	0	0	0	2100	1260	0	180	240	2280	1500
T3	CBR	0	1800	0	0	0	1560	1800	30	120	90	1710	1890
T4	CBR	0	2700	0	0	0	660	2700	60	180	300	900	3000
T5	PBR	0	2100	0	0	0	1260	2100	30	60	210	1350	2310
T6	PBR	480	2100	0	0	0	780	2580	60	150	150	990	2730
T7	PBR	0	2160	0	0	0	1200	2160	60	30	150	1290	2310
T8	PBR	0	2700	0	0	0	660	2700	0	120	90	780	2790
T9	PBR	1020	2160	0	0	0	180	3180	0	90	210	270	3390
T10	CBR	0	1200	0	0	0	2100	1200	30	90	360	2220	1560
T11	CBR	0	1200	0	0	0	2100	1200	30	0	420	2130	1620
T12	CBR	0	2100	0	0	0	1200	2100	30	60	270	1290	2370
T13	CBR	0	1740	0	0	0	1560	1740	0	210	360	1770	2100
T14	CBR	0	1680	0	0	0	1620	1680	90	30	330	1740	2010
T15	PBR	0	1200	0	0	0	2100	1200	0	30	210	2130	1410
T16	PBR	0	1620	0	0	0	1680	1620	0	90	180	1770	1800
T17	PBR	0	1680	0	0	0	1620	1680	30	60	270	1710	1950
T18	PBR	0	1620	0	0	0	1680	1620	30	150	90	1860	1710

Table E6. Data necessary for preparation and inspection meeting stages' metrics calculation (3)

Team	Method	Case 2					Case 3					Case 4				
		CtADD_FP	*CtFP*	*ΔCtELIM_FP*	*Ct*	*ΔCt*	*CtADD_FP*	*CtFP*	*ΔCtELIM_FP*	*Ct*	*ΔCt*	*CtADD_FP*	*CtFP*	*ΔCtELIM_FP*	*Ct*	*ΔCt*
T1	CBR	60	360	240	2400	1620	240	1440	960	3660	2340	480	2880	1920	5340	3300
T2	CBR	0	360	480	2460	1740	0	1440	1920	3540	3180	0	2880	3840	4980	5100
T3	CBR	60	240	180	1860	1980	240	960	720	2760	2520	480	1920	1440	3960	3240
T4	CBR	120	360	600	1140	3300	480	1440	2400	2580	5100	960	2880	4800	4500	7500
T5	PBR	60	120	420	1440	2520	240	480	1680	1980	3780	480	960	3360	2700	5460
T6	PBR	120	300	300	1200	2880	480	1200	1200	2460	3780	960	2400	2400	4140	4980
T7	PBR	120	60	300	1380	2460	480	240	1200	1920	3360	960	480	2400	2640	4560
T8	PBR	0	240	180	900	2880	0	960	720	1620	3420	0	1920	1440	2580	4140
T9	PBR	0	180	420	360	3600	0	720	1680	900	4860	0	1440	3360	1620	6540
T10	CBR	60	180	720	2340	1920	240	720	2880	3060	4080	480	1440	5760	4020	6960
T11	CBR	60	0	840	2160	2040	240	0	3360	2340	4560	480	0	6720	2580	7920
T12	CBR	60	120	540	1380	2640	240	480	2160	1920	4260	480	960	4320	2640	6420
T13	CBR	0	420	720	1980	2460	0	1680	2880	3240	4620	0	3360	5760	4920	7500
T14	CBR	180	60	660	1860	2340	720	240	2640	2580	4320	1440	480	5280	3540	6960
T15	PBR	0	60	420	2160	1620	0	240	1680	2340	2880	0	480	3360	2580	4560
T16	PBR	0	180	360	1860	1980	0	720	1440	2400	3060	0	1440	2880	3120	4500
T17	PBR	60	120	540	1800	2220	240	480	2160	2340	3840	480	960	4320	3060	6000
T18	PBR	60	300	180	2040	1800	240	1200	720	3120	2340	480	2400	1440	4560	3060

Table E7. Results of comparison of metric values between CBR and PBR teams for preparation stage of inspection

Case of evaluation	Metric	System	Mean		p-value
			CBR	PBR	
Case 0	*Mk, Mg_IDV*	Hospital	0.69	0.72	0.323
		Seminar	0.73	0.52	0.007*
		Both systems	0.71	0.63	0.101
	Ml_IDV	Hospital	0.05	0.04	0.209
		Seminar	0.05	0.06	0.119
		Both systems	0.05	0.05	0.5
Case 1	*Mk*	Hospital	0.62	0.67	0.247
		Seminar	0.65	0.48	0.011*
		Both systems	0.64	0.59	0.199
	Mg_IDV	Hospital	0.58	0.65	0.218
		Seminar	0.60	0.44	0.017*
		Both systems	0.59	0.56	0.265
	Ml_IDV	Hospital	0.19	0.14	0.083
		Seminar	0.21	0.20	0.372
		Both systems	0.20	0.17	0.079
Case 2	*Mk*	Hospital	0.57	0.63	0.203
		Seminar	0.58	0.45	0.017*
		Both systems	0.58	0.55	0.314
	Mg_IDV	Hospital	0.48	0.57	0.155
		Seminar	0.48	0.38	0.054
		Both systems	0.48	0.48	0.489
	Ml_IDV	Hospital	0.33	0.24	0.074
		Seminar	0.38	0.35	0.294
		Both systems	0.35	0.29	0.055
Case 3	*Mk*	Hospital	0.38	0.46	0.133
		Seminar	0.36	0.32	0.128
		Both systems	0.37	0.40	0.276
	Mg_IDV	Hospital	-0.15	0.12	0.073
		Seminar	-0.29	-0.13	0.107
		Both systems	-0.23	0.01	0.021*
	Ml_IDV	Hospital	1.18	0.82	0.066
		Seminar	1.34	1.19	0.232
		Both systems	1.27	0.99	0.039*
Case 4	*Mk*	Hospital	0.27	0.34	0.123
		Seminar	0.24	0.23	0.328
		Both systems	0.25	0.29	0.154
	Mg_IDV	Hospital	-0.99	-0.48	0.066
		Seminar	-1.33	-0.79	0.027*
		Both systems	-1.18	-0.61	0.005*
	Ml_IDV	Hospital	2.31	1.61	0.064
		Seminar	2.63	2.33	0.222
		Both systems	2.49	1.93	0.036*

* indicates significant results, i.e. $p<0.05$

Table E8. Results of comparison of metric values between CBR and PBR teams for preparation and inspection meeting stages of inspection

Case of evaluation	Metric	System	Mean		p-value
			CBR	PBR	
Case 0	*Mk*	Hospital	0.44	0.67	0.034*
		Seminar	0.38	0.38	0.453
		Both systems	0.41	0.54	0.058
	Mg_MEET	Hospital	0.21	0.61	0.043*
		Seminar	-0.02	0.26	0.164
		Both systems	0.08	0.46	0.023*
Case 1	*Mk*	Hospital	0.44	0.67	0.028*
		Seminar	0.43	0.39	0.290
		Both systems	0.44	0.55	0.067
	Mg_MEET	Hospital	0.19	0.59	0.034*
		Seminar	0.1	0.28	0.206
		Both systems	0.14	0.45	0.021*
	Ml_MEET	Hospital	5.39	2.06	0.115
		Seminar	3.11	2.48	0.299
		Both systems	4.13	2.25	0.093
Case 2	*Mk*	Hospital	0.45	0.66	0.024*
		Seminar	0.47	0.42	0.165
		Both systems	0.46	0.55	0.079
	Mg_MEET	Hospital	0.18	0.58	0.027*
		Seminar	0.18	0.29	0.263
		Both systems	0.18	0.45	0.021*
	Ml_MEET	Hospital	2.87	1.17	0.115
		Seminar	1.61	1.29	0.295
		Both systems	2.17	1.22	0.095
Case 3	*Mk*	Hospital	0.45	0.63	0.019*
		Seminar	0.58	0.49	0.035*
		Both systems	0.52	0.57	0.200
	Mg_MEET	Hospital	0.11	0.50	0.014*
		Seminar	0.41	0.33	0.254
		Both systems	0.27	0.43	0.074
	Ml_MEET	Hospital	0.88	0.45	0.149
		Seminar	0.48	0.41	0.281
		Both systems	0.66	0.43	0.136
Case 4	*Mk*	Hospital	0.46	0.62	0.025*
		Seminar	0.64	0.54	0.059
		Both systems	0.56	0.58	0.357
	Mg_MEET	Hospital	0.04	0.44	0.019*
		Seminar	0.50	0.35	0.15
		Both systems	0.30	0.40	0.22
	Ml_MEET	Hospital	0.54	0.33	0.198
		Seminar	0.29	0.26	0.324
		Both systems	0.40	0.30	0.198

* indicates significant results, i.e. $p<0.05$

GLOSSARY

The glossary contains some terms and acronyms commonly used in this thesis.

Term or acronym	Explanation
Empirical investigation, or Empirical study	An act or operation for the purpose of discovering something unknown or of testing a hypothesis, involving an investigator gathering data and performing analysis to determine what the data mean [Basili et al. 1999].
Experiment	A form of empirical study where the researcher has control over some of the conditions in which the study takes place and control over the independent variables being studied; an operation carried out under controlled conditions in order to test a hypothesis against observation [Basili et al. 1999].
Hypothesis	An educated guess that there exists a causal relation among constructs of theoretical interest.
Independent variables	The variables used to measure the causal construct.
Dependent variables	The variables used to measure the affected constructs.
Individual inspection, or Individual reading	The preparation stage of inspection where inspectors work independently of each other to produce the list of defects each.
Interacting team (or Interacting group) inspection	The inspection meeting stage of inspection where inspectors interact face-to-face in a team of three, using the results of their prior individual inspection to produce a single list of defect for the team.
Nominal team (or Nominal group) inspection	A nominal team is an artificial team that consists of the same members as an interacting team. However in this case, inspectors do not interact with one another. This is not a separate activity; its measurement is generated from individual inspection performance.
True defect	Defect detected during inspection, which requires rework. A true defect causes the program to fail.
False positive	Erroneously identified defect, which is not a true defect and requires no rework. The rework of false positives can introduce new defects.
CBR	Checklist-based reading technique, used to guide inspectors during preparation stage of inspection. Provides inspector with a checklist.
PBR	Perspective-based reading technique, used to guide inspectors during preparation stage of inspection. Provides inspector with a scenario.