Fraunhofer Institut
Experimentelles
Software Engineering

Investigating the Impact of Reading Techniques on the Accuracy of Different Defect Content Estimation Techniques

Authors:
Bernd Freimut
Oliver Laitenberger
Stefan Biffl

Accepted for publication in the
proceedings of Metrics'2001

IESE-Report No. 061.00/E
Version 1.0
April 1, 2001

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competetive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
D-67661 Kaiserslautern

Abstract

Software inspections have established an impressive track record for early defect detection and correction. To increase their benefits, recent research efforts have focused on two different areas: systematic reading techniques and defect content estimation techniques. While reading techniques are to provide guidance for inspection participants on how to scrutinize a software artifact in a systematic manner, defect content estimation techniques aim at controlling and evaluating the inspection process by providing an estimate of the total number of defects in an inspected document. Although several empirical studies have been conducted to evaluate the accuracy of defect content estimation techniques, only few consider the reading approach as an influential factor.

In this paper we examine the impact of two specific reading techniques - a scenario-based reading technique and checklist-based reading - on the accuracy of different defect content estimation techniques. The examination is based on data that were collected in a large experiment with students of the Vienna University of Technology. The results suggest that the choice of the reading technique has little impact on the accuracy of defect content estimation techniques. Although more empirical work is necessary to corroborate this finding, it implies that practitioners can use defect content estimation techniques without any consideration of their current reading technique.

Keywords: Software Inspection, Defect Content Estimation, Experimentation

Table of Contents

1 Introduction

Software organizations must deliver high quality products on time and within budget to remain competitive in the marketplace. However, the reliable and predictable development of high quality software continues to be a major problem, largely due to the inadequate and late removal of defects.

One of the proposed solutions for early detection and removal of defects is software inspection [15] [17]. A software inspection represents a method that allows the detection and removal of defects immediately after software documents have been created. Since its inception in the early 1970s, the software inspection methodology has evolved into one of the most cost-effective methods for early defect detection and removal.

To increase the benefits of the inspection method, recent research work has concentrated on systematic reading techniques and defect content estimation techniques. A reading technique can be defined as a series of steps or procedures that help an inspector perform the defect detection activity of an inspection. In this way it supports an inspector in acquiring a deep understanding of the software artifact. Understanding is a major prerequisite for detecting subtle and/or complex defects, which often cause most of the problems if detected in later life cycle phases. In a sense, a reading technique can be regarded as a mechanism for an inspector to detect defects in the software artifact. While most industrial inspection implementations use either no specific reading approach (often termed ad-hoc) or checklist-based reading (CBR) during defect detection [15][17], researchers recently suggested more procedural scenario-based techniques, such as defect-based or perspective-based reading, and validated them empirically [2] [19] [26].

Defect content estimation techniques (DCET), on the other hand, are used as a basis for estimating the number of remaining defects in a software document after an inspection. Based on this information, the inspection team can decide whether to re-inspect a document to ensure that it is below a pre-specified quality threshold, and that the inspection process itself has attained a minimal level of effectiveness. Several classes of DCETs have been described and empirically studied: Subjective defect content estimation (SDCE) [14], Capture-Recapture (CR) models [13][5] [23], and curve-fitting models (CF) [36].

SDCE requires inspection participants, such as the most experienced inspector to estimate the number or percentage of defects detected. Based on this estimate, one can calculate the total number of defects in the document and decide whether to perform a re-inspection. CR models originate from wildlife re-

search and have been applied in biology to the estimation of the size of animal populations. The same models can be used in an inspection context to estimate the number of defects in a software document. CF models involve fitting a curve to the data obtained from an inspection, and using the curve for predicting the total number of defects in a document.

It becomes clear from the description that both approaches, reading techniques and defect content estimation techniques, address the cost-effectiveness of inspections from a different angle. While reading techniques focus on finding a maximum number of defects in the inspected document, DCETs help evaluate the document and inspection process quality. Hence, it should be beneficial to use and optimize both approaches in an inspection implementation. However, some of the DCETs make assumptions that are clearly violated by reading techniques in general and the more procedural scenario-based ones in particular. Thus, the effect of using a particular reading technique on the accuracy of different DCETs needs to be investigated empirically. After an initial investigation based on simulated inspection data [29], this paper examines this issue using data from inspections performed.

This paper is organized as follows. In Section 2 we explain defect content estimation techniques, reading techniques, and their combination in more detail. Section 3 describes the experiment in which the data were collected. Section 4 presents a discussion of our results. Finally, Section 5 concludes the paper with a summary and suggestions for future work.

2 Background

This section presents background information on reading techniques, defect content estimation approaches, and their combination.

2.1 Reading Techniques

In practice, most industrial inspection implementations use either no specific reading approach (often termed ad-hoc) or checklist-based reading (CBR) during the defect detection activity of an inspection. Ad-hoc reading, as its name implies, provides no explicit advice for inspectors as to how to proceed, or what specifically to look for, during the reading activity. Hence, the results of the reading activity in terms of potential defects or problem spots are fully dependent on human experience and expertise. Checklists offer stronger support mainly in the form of yes/no-questions that inspectors need to answer while reading a software document. Gilb and Graham [17] state that checklist questions interpret specified rules within a project or an organization. Although a checklist provides advice on what to look for in an inspection, it does not describe how to identify the necessary information and how to perform the required checks. Moreover, for CBR as well as for ad-hoc it remains unclear to what degree a systematic reading process was applied.

Recently, Victor Basili [2] proposed scenario-based reading techniques. These techniques make use of so-called scenarios. Scenarios are algorithmic guidelines for the inspector that describe how to go about finding the required information in a software document, as well as what that information should look like. Since these techniques offer more procedural support for the defect detection activity, they are more prescriptive than either the ad-hoc or the checklist-based technique.

Several scenario-based reading techniques have been suggested so far [2]. Amongst them were defect-based reading [26], perspective-based reading [1], and traceability-based reading [32]. The scenario-based approach that we are focusing on in this paper is a combination of perspective-based reading (PBR)[1][19] and traceability-based reading (TBR) [32]. The idea of PBR is to use a multi-view approach that allows different inspection participants to adopt different stakeholder perspectives. The viewpoints from which to read are primarily derived from the roles in the software development process. When using the PBR technique the documentation to be inspected is read, for example, from the perspective of the developer of the previous documents and the developer of the subsequent documents, and the tester. The essence of the

traceability-based reading technique (TBR) is the tracing of information between various parts of an artifact to ensure their consistency, correctness, and completeness. Hence, this technique provides a well-defined, systematic set of checks that an inspector needs to perform.

Since the large number of required checks usually overwhelms a single inspector, we combined the PBR and TBR technique in the following manner to reduce the workload. We identified various stakeholder perspectives according to PBR principles. For each perspective, we developed a specific scenario. The scenario provides guidance for an inspector in the form of procedures for extracting the information relevant for a particular stakeholder as well as procedures for examining the extracted information. The latter were selected from the required consistency, completeness, and correctness checks of the TBR technique. In this way, an inspector performs only those checks that are relevant for a particular stakeholder perspective and not all of them in a systematic manner.

An inspection team, thus, consists of inspectors, each of whom has read the document from a different angle and has performed a particular set of checks. As a consequence, each inspector can make a unique contribution to the inspection results and little defect detection effort is duplicated. A more detailed description of these techniques can be found in [19] and [32].

2.2 Defect Content Estimation Techniques

Defect Content Estimation Techniques aim at estimating the total number of defects that are contained in an inspected software artifact. Using this estimate and the known number of defects found in the inspection, it is possible to estimate the number of remaining defects in the inspected document. Based on this information, the inspection team can make an informed decision whether to re-inspect the document to reduce its defect content before passing it on to the next activities of the development life cycle.

As an additional application of the defect content estimate it is possible to determine the percentage of total defects found. Using this information, the quality of the inspection process can be determined directly after the inspection meeting.

Several techniques have been proposed to estimate the total number of defects in a document after an inspection. As shown in Figure 2, these techniques can be classified as subjective and objective.

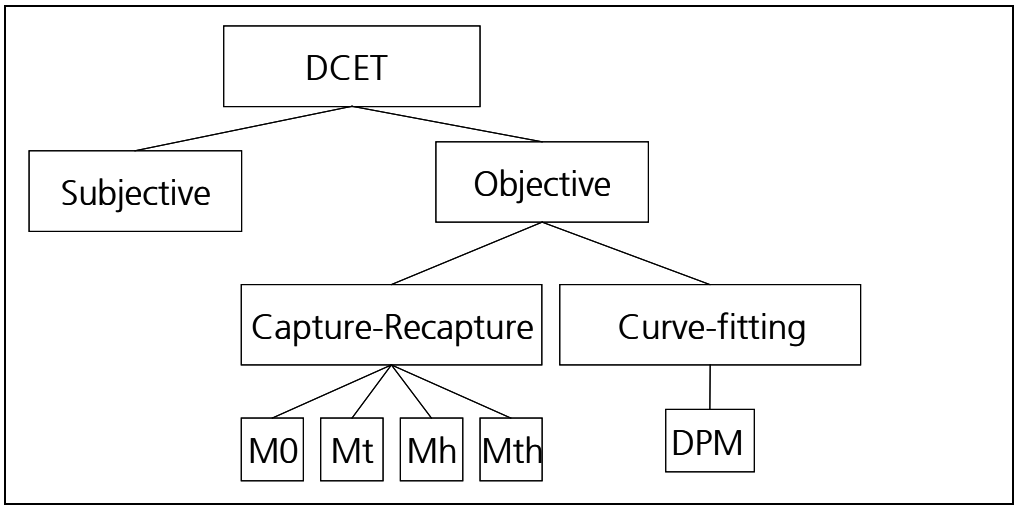


Figure 1: Overview of DCETs

In the following, each one of them is explained in more detail.

2.2.1 Subjective Approach.

The basic concept behind subjective estimation is to ask inspectors after an inspection to estimate the percentage of defects in the artifact they believe they have actually found or the number or percentage of defects still remaining in the inspected document. Using this estimate, one can calculate the total number of defects and the remaining number of defects in the inspected artifact. The latter is the basis for deciding whether to perform a re-inspection.

The overall advantage of the subjective approach is its simplicity. No specific data except the estimate need to be collected. On the other hand, this approach relies on human judgement and the ability of the inspectors as estimators. This raises issues with respect to human variation and the reliability of the defect content estimate. However, current evidence with professional developers indicates that subjective estimates can provide very accurate results [14].

In contrast to the point estimate studied in [14] we asked the inspectors in our study to provide three estimates of the number of remaining defects, namely for the most-likely value, the maximum value, and the minimum value for the number of remaining defects. Adding these estimates to the number of defects detected resulted in a three-point estimate of the total number of defects.

The rationale for eliciting a three-point estimate was that we considered it easier for the inspectors to express the uncertainty in their estimate by providing an additional range, which is given by minimum, most-likely, and maximum es-

timate, instead of forcing them to provide a single value [6]. When making the re-inspection decision, the magnitude of this range can give an indication of the risk associated with the decision. However, for this purpose the properties of the three-point estimate need to be investigated. Ideally, we would expect the maximum value to overestimate the actual number of defects, the minimum value to underestimate, and the most-likely one to be quite accurate.

2.2.2 Capture-Recapture Models.

CR models have their origin in biology and wildlife research. In these disciplines, the models are used to estimate the size of an animal population based on incomplete samples of the animal population captured in several trapping occasions [23]. Since this problem is comparable to estimating the total number of defects in an inspection based on the samples of defects found by the inspectors, it is possible to apply these models in an inspection context.

To illustrate the rationale behind CR models, the following example is given: Suppose two independent inspectors scrutinize a document for defects. The document has N defects and each inspector has the same probability p to detect a single defect. Let n_1 denote the number of defects detected by inspector 1, n_2 the number of defects detected by inspector 2, and n_{12} the number of defects detected by both inspectors (i.e., the number of defects both inspectors had detected in common).

With $n_1 = N \times p$, $n_2 = N \times p$, and $n_{12} = N \times p \times p$ one can estimate the total number of defects $\hat{N}$ in the following manner:

$$\hat{N} = \frac{(N \times p) \times (N \times p)}{N \times p \times p} = \frac{n_1 \times n_2}{n_{12}}$$

In biology, this estimator is known as the Lincoln-Peterson estimator.

Based on this rationale, different models and estimators have been suggested so far. They differ in the assumptions they make about the "detectability" of defects. For example, the model presented above assumes that the inspectors have the same probability of detecting defects and that all defects have the same probability of being detected.

This simple model is not realistic for inspections for two reasons. First, the probability of detecting defects usually differs among inspectors. For example, more experienced inspectors have a higher probability of detecting defects than less experienced ones. Second, defects usually have different probabilities of being detected. For example, defects that are easy to detect have a higher probability of being detected than defects that are difficult to detect.

Generally, existing CR models vary in the assumptions made on the probability of defects being detected and the probability of the inspectors to detect defects. If defects with varying probabilities of being detected are assumed, this is called the heterogeneity source of variation[1]. If inspectors with varying detection probabilities are assumed, this is called the time response source of variation.

With these two sources of variation, there exist four different CR models, each of which with one or two estimators. The models with their assumptions and estimators are depicted in Table 1.

Model	Assumptions	Estimator
M0	Every defect has the probability p of being detected by every inspector. Thus, all defects have the same detection probability, and all inspectors have the same detection capability.	Maximum Likelihood Estimator M0(MLE) [23]
Mt	Every inspector i has the probability pi of detecting every defect. Thus, all different defects have the same detection probability, but the inspectors have different detection capabilities.	Maximum Likelihood Estimator Mt(MLE) [23]
		Chao's Estimator Mt(Ch)[9]
Mh	Every defect j has the probability pj of being detected, which is the same for every inspector. Thus, different defects can vary in their detection probability, but all inspectors have the same detection capability.	Jackknife Estimator Mh(JE) [8]
		Chao's Estimator Mh(JE) [10]
Mth	Every defect j has the probability pj of being detected and every inspector i has the probability pi of detecting defects. The probability pij that inspector i detects defect j is computed as pij = pipj. This allows for different detection probabilities for the different defects and inspectors.	Chao's Estimator Mth(Ch) [11]

Table 1: Overview of capture-recapture models

2.2.3 Curve-fitting Models.

Wohlin and Runeson proposed curve-fitting models for estimating defect content [36]. The rationale of this type of model is to sort the defect data according to a given criterion, plot the sorted defect data in a graph, and use regression techniques to fit a curve through the data points. Based on the fitted curve, an estimate of the total number of defects is made. The most-widely investigated approach following this procedure is the Detection Profile Method (DPM) [36].

[1] These expressions have their origin in the biological domain of the CR models

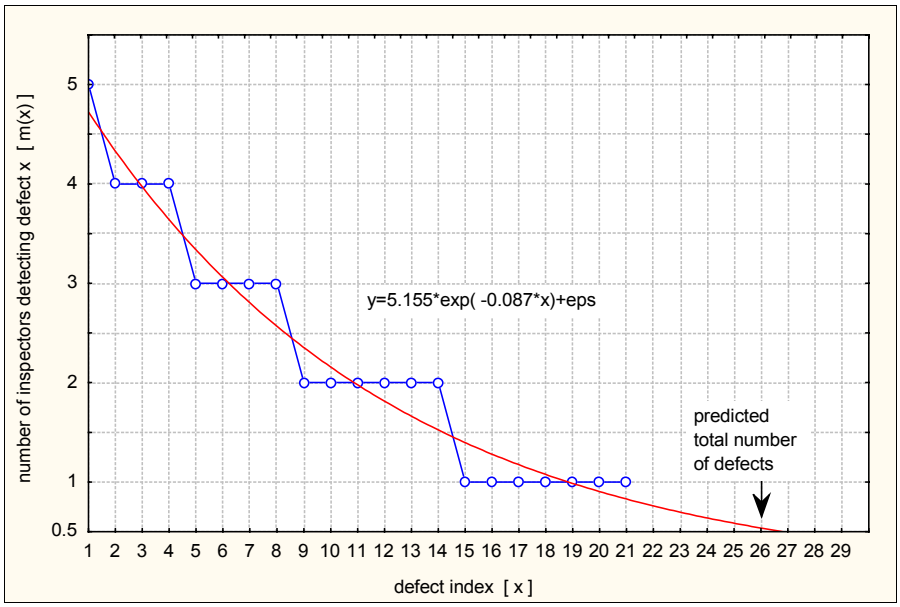


Figure 2: Example of the Detection Profile Method

To apply the DPM, the defects are plotted with the defect index on the x-axis and the number of inspectors that detected a defect on the y-axis. The defects are sorted in a decreasing order with the number of inspectors as sorting criterion. This results in a graph similar to the one depicted in Figure 2. The DPM approach assumes that the shape of the data points can be described by an exponential curve. Thus, an exponential curve is fitted through the data points, which is then used to estimate the total number of defects. In particular, the estimate of the total number of defects is the largest integer that results in the fitted curve being greater than 0.5. The rationale in selecting this value is that the continuous values of the exponential curve are rounded to the nearest integer value. Thus, in selecting 0.5 as a threshold, one essentially searches for the largest defect index that one inspector should have detected.

Based on the rationale of curve-fitting models, various alternatives to the DPM approach have been proposed and investigated [3] [24] [25] [30] [36]. However, in these studies the performance of the alternative CF models was overall found either to be inferior to the DPM approach or comparable to it. Therefore, we decided to select only the DPM as a representative of a CF model in our study.

2.3 The Combination of Reading Techniques and DCETs

When using existing state-of-the-practice reading techniques, such as CBR, inspectors look for all kinds of defects in the documents. Thus, all inspectors fo-

cus on the same set of defects. However, when following the description of a scenario, the inspectors target a specific set of defects. With a scenario-based approach, a subset of the defects in the document has, therefore, a high probability of being detected, while the remainder of the defects has a relatively low probability of being detected. Conversely, with CBR one would expect more uniformity in the detection probability of defects [19][21]. We can depict this in a simple case with a scenario-based and a CBR inspection team consisting of two inspectors respectively in Table 2.

	Team 1		**Team 2**	
	Inspector 1	Inspector 2	Inspector 3	Inspector 4
Reading Technique	Scenario A	Scenario B	Checklist	Checklist
Subset A	$P_{A\text{-}High}$	$P_{B\text{-}Low}$	P	P
Subset B	$P_{A\text{-}Low}$	$P_{B\text{-}High}$	P	P

Table 2: Detection Probabilities for Scenario- and Checklist-based Reading.

Table 2 represents an idealistic and simplified situation to explain the theory of our expectations. In practice, the situation is often complicated by the fact that inspection teams usually consist of more than two inspectors and that within the different defect subsets, some defects are easy and some are difficult to detect (i.e., there is further subdivision).

The specific subset focus of a scenario-based reading technique has two implications. First, individual inspectors using a scenario need not be more effective than if they were using a checklist. Any effectiveness benefit of the scenario-based reading technique should therefore become apparent on the team level. This line of reasoning is more elaborated upon in [21]. Second, since defect content estimates are often based on individual findings, the use of a scenario-based reading technique with a specific subset focus in combination with a specific DCET may lead to less accurate predictions. In the following we discuss the consequences of these effects in more detail according to the different types of DCETs.

2.3.1 Subjective Estimation

If inspectors apply the scenario-based technique, they will only look for a specific subset of defects. Hence, their subjective estimates represent the total number of defects in the subset or a number of subsets rather than the total number of defects in the document. Hence, we expect severe underestimation of the true defect content and, thus, less accurate results for the scenario-based approach compared to CBR.

2.3.2 Capture-Recapture Methods

The fundamental principle behind using CR models for software inspections is to let several inspectors draw samples from the population of defects. Based on the overlap of defects amongst inspectors, one can estimate the number of defects remaining in a software document using a statistical estimator.

Since the estimate is based on the overlap of defects amongst inspectors, its accuracy is biased when using a scenario-based reading technique. This effect stems from the fact that scenario-based techniques modify the defect detection probabilities of inspectors. The modification of defect detection probabilities, however, directly leads to a violation of the assumptions of all CR models as described above. Hence, we can expect less accurate estimates if inspectors follow a scenario-based reading technique and lower estimates for the scenario-based reading technique than for CBR [16].

2.3.3 Curve-fitting Models

The idea behind using CF models is to fit a curve to the data obtained from an inspection, and to use the curve for predicting the total number of defects in a document. However, it is difficult to make a prediction of the behavior of CF models based on theoretical considerations.

On the one hand, one can expect that almost all defects might be found by more than one inspector when CBR is used. However, as discussed in [3], a problem of the DPM is that it tends to severely overestimate if no defect was found by exactly one inspector. This overestimation leads to less accurate results.

On the other hand, with a scenario-based approach one can expect that only one inspector detects defects of a specific subset. This may result in a very long tail of the fitted curve and, thus, also in less accurate estimations.

3 Empirical Study

In this section we describe the approach for collecting and analyzing the data.

3.1 Experimental Context and Subjects

The experiment was part of a two-semester university software development workshop that taught more than 200 undergraduate students on how to develop a medium-sized software product. 169 students participated in the experiment. The participants had a varying degree of development experience. While all of them knew how to develop small programs, about ten percent had professional experience with the development of larger systems.

3.2 Experimental Material

The inspected artifact was a requirements document describing a distributed administrative information system for managing ticket sales. The document notation was text in natural language (introduction, business functions, non-functional requirements), and graphical models (an object-oriented class model and class description, and a relational database model).

The size of the document was 35 pages containing 9000 words, 6 diagrams, and 86 seeded defects. The defects were the ones that have been found during the development of the requirements documents. All defects could be found without a need for reference to external documentation. A defect in the requirements document was classified as either missing, unnecessary, wrong, ambiguous, or inconsistent information [1].

The experiment primarily investigated the effects of checklist-based reading (CBR) and scenario-based reading techniques (SBR) for a requirements document. The checklist was developed before the development of the requirements specification. To avoid any bias later on, the person responsible for the creation of the checklist did not participate in the development of the specification. The checklist consisted of seven quality sections, e.g., completeness and testability. For each section, questions were asked that hinted on defects. Examples for questions are the following:

– ‘Are there any business functions that lack proper description of input or necessary information on processing?’

– 'Are there any defects with respect to the notation of data, function, or object models?'

To support SBR, we tailored scenarios from the perspective-based reading technique (PBR) [1] [19], which originally dealt with natural language requirements, and scenarios from the traceability-based reading technique (TBR) [32], which dealt with graphical models of the Unified Modeling Language, to our inspection situation. The scenarios were created for the perspectives of a user, a designer, and a tester. The user task was to derive use cases from context information and business functions, and to check business functions and their constraints with data model entities and their integrity rules. The designer task was the construction of sequence diagrams from the business functions and the object model, and the consistency check of these models. The tester task required the development of test cases from business function input and non-functional requirements, and involved checks of the consistency between entities and attributes of the data model and the object model.

3.3 Experimental Design and Conduct

The ideal of experimentation is to construct an environment in which all of the variation in the dependent variable is systematically related to the experimenter's manipulation of one or more independent variables [33]. In this case, systematic variation can only attributed to the treatment effect. The dependent variable in our experiment is the accuracy of the various DCETs, the independent variable is the reading technique.

Since the subjects in our experiment had varying degrees of development experience, and since our primary interest was in the inspection team results, we needed to ensure that the teams were relatively homogeneous. To control this source of variation, we used "blocking" [33] based upon the results of a pretest. The pretest consisted of rating students by workshop supervisors based on their scores for developing a small object-oriented application, and resulted in three student groups: Students with excellent skills, students with medium skills, and students with little skills.

To ensure homogeneity, we randomly selected students from all three groups for one inspection team. Following this approach, we could eliminate known sources of discrepancy in the experience level. The teams themselves were randomly assigned to the reading technique. This means that unknown sources of discrepancy are forced by randomization to contribute homogeneously to both treatments[2]. Within each team, each subject independently inspected the requirements document using either the checklist or one of the three scenarios. We formed teams with a varying number of subjects, i.e., 4 to 6, so that 3

[2] In the design of the experiment, we followed the suggestion presented in [7]: Block what you can and randomize what you cannot!

teams with 4 inspectors, 11 teams with 5 inspectors, and 17 teams with 6 inspectors participated in the experiment..

The experiment itself consisted of a training session, an individual preparation, and a meeting phase. In the training phase the subjects were taught inspection methods, their assigned defect detection technique, and the experimental procedure. Moreover, the subjects performed an exercise inspection on a sample software requirements document to familiarize themselves with the inspection process and the forms.

Throughout the individual preparation phase, each subject applied the assigned reading technique independently to scrutinize the document for defects. Results of this activity were a defect list and an inspection protocol, which includes estimates on the number of defects remaining in the document.

In the meeting phase, the inspection team met to consolidate their individual defect lists and to create a joint team defect list.

Inspection supervisors performed an initial data analysis: They checked the completeness and validity of the collected defect and effort data and tried to match each defect on the team defect list with a defect in the reference defect list, which had been provided by the inspection team. False positives were not considered in this study as it is assumed that the document author represented by the inspection supervisors would have identified and rejected those false positives.

3.4 Validity Considerations

As any empirical study, this experiment exhibits a number of threats to internal and external validity [27] [34].

The primary threat to internal validity is selection. This comes from the selection of subjects and their assignments to particular treatments. In our case, we used blocking to ensure homogeneity among the inspection teams.

A second threat to internal validity is process conformance. To address this issue, we checked whether the subjects performed the required tasks for the scenario-based approach in a qualitative manner (e.g., whether they created the models or test cases). Under the assumption that the application of the scenario-based approach might not have led to "scenario-based data[3]", the experiment still provides some value, since it examines whether different DCETs provide practically useful results.

[3] Scenario-based data is characterized by the fact that defect detection probabilities are clearly different for the various scenarios. However, this is difficult to determine in practice.

With respect to external validity, we took a specification from a real application context to deal with an inspection object that was representative of an industrial development situation. Moreover, we used inspection activities that had been installed in a number of professional development environments [20]. Of course, the subjects were students participating in a university course. As pointed out in the literature [12], students may not be representative of real developers. In our case, this can have two implications. First, participants may not be as effective in their defect detection activity as professional developers, i.e. they find fewer defects. Second, they find different (types of) defects than professionals. However, these effects impact all the DCETs in a similar manner. Hence, although our estimates may not be as accurate with students as with professional developers our findings are conservative with respect to the identification of the best models. Hence, our results expose some external validity.

3.5 Data Analysis Approach

The first step in the data analysis was the estimation of the defect content for each inspection team. This was performed for all DCETs presented in section 2.2.

The second step was the selection of a set of criteria by which the defect content estimates were compared. Two criteria have been used in the past to evaluate DCETs: The relative error criterion [5] [22] [35] and the relative decision accuracy criterion [14]. The relative error criterion measures the accuracy of the defect content estimates in terms of the relative error (RE), which is defined as

$$RE = \frac{estimated\_number - actual\_number}{actual\_number}$$

RE allows us to distinguish between overestimation (too many defects are estimated, thus, a positive RE is obtained) and underestimation (too few defects are estimated, thus, a negative RE is obtained). To assess the performance of the estimators, we investigated the RE distributions generated from the set of RE values from all inspection teams. In doing so, we investigated the central tendency of the RE distribution measured as the median relative error. This median value can be seen as bias of the estimator. Additionally, it is also important to look at the RE variability. Variability tells us whether a large variation around the central tendency can be expected, e.g., whether the model produces extreme outliers. We use interquartile ranges as measure of variability.

In a recent study the relative decision accuracy of a defect content estimate was used as evaluation criterion [14]. This criterion assesses the suitability of a DCET for making the re-inspection decision. To do so, the re-inspection deci-

sion based on the defect content estimate is compared to the decision of never re-inspecting a document.

The primary objective of this study is to investigate the accuracy of the DCET estimates when different reading techniques are employed. To be able to investigate this relationship in this manner, the relative error criterion was preferred over the relative decision accuracy criterion.

To compare the RE distribution for various DCETs, boxplots were generated. These boxplots show the central tendency of the relative error by means of the median relative error and the variability around the central tendency by means of the interquartile range.

In addition to the interpretation of the boxplot, statistical testing was performed. Two questions were to be answered with the testing approach. The first one was whether there is any statistical significance between the DCETs' bias for CBR and SBR. For this purpose, the Mann-Whitney Test [28] was used to investigate whether there is a significant difference among the DCETs' *absolute* bias. We use absolute values here because we do not make a distinction between over- and under- estimation. For this test, the following null hypothesis was stated for each DCET:

H_{01}*: There is no difference in the absolute bias of DCETs between CBR and SBR.*

The second question was, which of the DCETs would represent a recommendable DCET to be used in the investigated environment. For this purpose, a similar testing strategy as proposed in [4] was chosen. First, the CR models where compared to each other to investigate whether one CR model could be selected as best CR model. Subsequently, the selected CR model, the curve-fitting model (i.e., the DPM), and the subjective approach (SDCE) were compared to each other for selecting the best type of DCET.

Prior to the two steps of selecting a best CR model and selecting the best type of DCET, the Kruskal-Wallis test [28], a non-parametric alternative to the ANOVA, was performed. For this test, the following null hypothesis was stated:

H_{02}*: There is no difference in the absolute bias between the considered DCETs.*

If this hypothesis can be rejected, a significant difference between the DCET exists and further testing takes place to identify the best one.

To address the first issue on selecting the best CR model in this environment, two steps were performed. First, we determined which of the two estimators for model Mh and Mt should be used. This was done using the two-tailed Mann-Whitney U Test on the absolute bias for the 31 inspections [28].

Subsequently, we determined which sources of variation ought to be taken into account: heterogeneity, time response, or both. We would expect that, as more sources of variation are considered, the absolute estimation bias would decrease. This means that M0(MLE) is expected to perform worst, and Mth(Ch) is expected to perform best. For this, we use a one-tailed test. For all tests, we consider an alpha level of $\alpha = 0.1$ as our significance level.

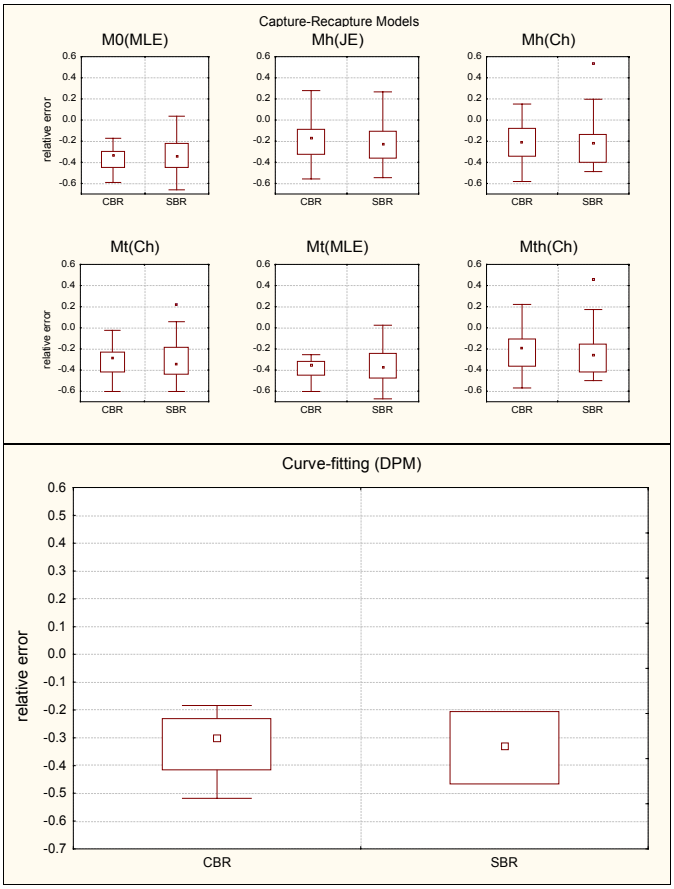
In several studies, the number of inspectors has been shown to have an impact on the accuracy of the defect content estimate [4][22]. Specifically, a minimal number of three to four inspectors was recommended for the application of DCETs in these studies. Since all our inspection teams consisted of four or more inspectors, we did not consider the number of inspectors as an additional factor in the analysis.

To address the second issue of selecting the best type of DCET in this environment, we performed a pair-wise comparison of the three DCETs using the two-tailed Mann-Whitney U Test.

4 Results and Discussion

4.1 The Difference between CBR and SBR

The first question we investigated was: "Is there a difference in the accuracy of the defect content estimates when different reading techniques are used?". To answer this question, the boxplots for the various DCETs are depicted in Figure 3. It shows the relative error distribution for the considered DCETs for both checklist-based reading (CBR) and scenario-based reading (SBR).



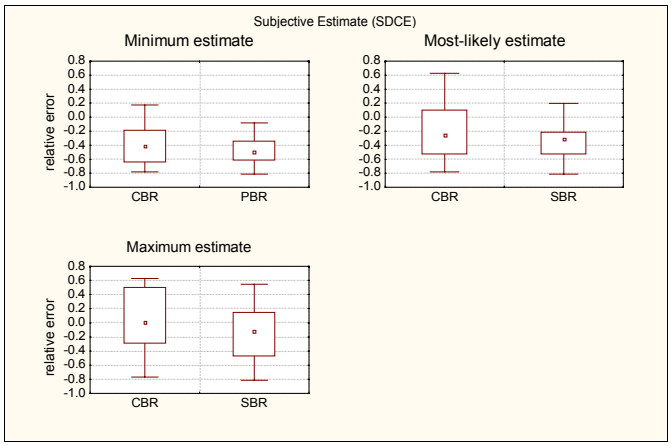


Figure 3: Accuracy of DCETs for CBR and SBR.

The detailed statistics for the median relative error are shown in Table 3.

DCET	CBR-Bias	SBR-Bias
M0(MLE)	-0.337	-0.349
MH(JE)	-0.174	-0.233
MT(MLE)	-0.355	-0.372
MT(CH)	-0.285	-0.349
MH(CH)	-0.215	-0.221
MTH(CH)	-0.198	-0.256
DPM	-0.302	-0.330
SDCE(min)	-0.413	-0.500
SDCE(mlv)	-0.256	-0.314
SDCE(max)	0.006	-0.116

Table 3: Median relative error for CBR and SBR

According to Figure 3, there is very little difference in the bias between CBR and SBR estimates. The median relative error does not seem to be very different for the two reading techniques. The concrete numbers in Table 3 confirm this for the median relative error. The table shows that the absolute median relative error is between 1% and 6% smaller for CBR than for SBR. Thus, the defect content estimates are more prone to underestimation with SBR than with CBR. For CR models, this contradicts our expectation while our expectation for curve-fitting models and SDCE was met. However, it has to be investigated whether these differences are statistically significant (i.e., are not due to chance).

Therefore, we performed the Mann-Whitney U Test to investigate hypothesis H_{01}. The test results are depicted in Table 4. The table shows the U-value for each DCET. With sample sizes larger than 20, the sampling distribution rapidly

approaches the normal distribution. Hence, the Z-values from the normal distribution and the corresponding p-levels are given.

DCET	U	Z	p-level
M0(MLE)	108.5	-0.455	0.649
MH(JE)	94	-1.028	0.304
MT(MLE)	109.5	-0.415	0.678
MT(CH)	109.5	-0.415	0.678
MTH(CH)	102	-0.716	0.477
MH(CH)	95.5	-0.968	0.332
DPM	110	0.395	0.693
SDCE(mlv)	99.5	-0.810	0.418

Table 4: Mann-Whitney test results for the absolute median relative error (CBR vs. SBR)

It can be seen that for all estimators the p-levels are far from the chosen α-level of 0.1. Thus, no statistically significant difference in the absolute bias between CBR and SBR can be observed and we cannot reject H_{01}.

Also, when looking at the variability of the estimates in Figure 3, no difference between CBR and SBR can be observed. However, one observation is that Chao's estimators seem to be more prone to outliers for SBR than for CBR. For these estimators it was shown that they could produce large outliers when few data (e.g., due to a small number of inspectors) are available ([5]). Thus, this observation might indicate that Chao's estimators could be affected by using SBR. This, however, has to be investigated further.

The practical consequence of this result is that – although, strictly speaking, SBR violates the assumptions of CR models – it might be possible to rely on the estimates nevertheless. In [31] it was investigated whether the adherence of the inspection to the DCETs' assumptions had an impact on the accuracy of the models. However, no relationship could be determined. Possible explanations for the result in [31] were either that the measures for the adherence to the assumptions are not appropriate, or that other factors beside the assumption affect the accuracy of the defect content estimates. Our result can be seen as a confirmation of the latter explanation.

4.2 Comparing DCETs

The next question we addressed was, which DCET would have been a reasonable choice for the environment of this study. Since in the previous section we concluded that the reading technique does not influence accuracy in a statistically significant manner, we did not distinguish the different reading techniques.

Similarly to section 4.1, we first present the overall results as a boxplot. This is shown in Figure 4.

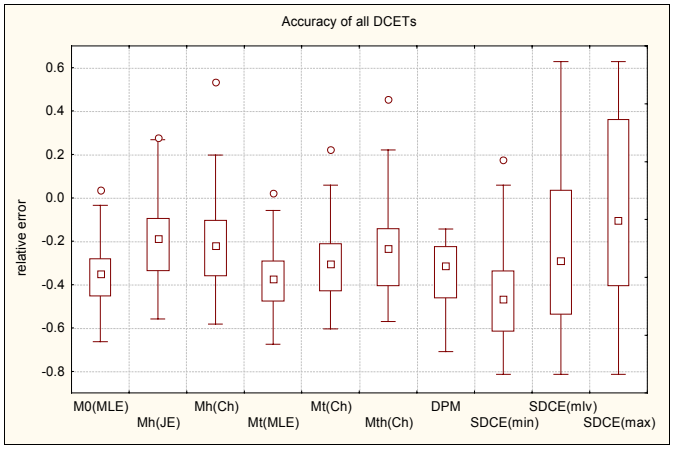


Figure 4: The accuracy for all DCETs

DCET	Bias
M0(MLE)	-0.349
MH(JE)	-0.186
MT(MLE)	-0.372
MT(CH)	-0.302
MH(CH)	-0.221
MTH(CH)	-0.233
DPM	-0.313
SDCE(min)	-0.465
SDCE(mlv)	-0.291
SDCE(max)	-0.105

Table 5: Bias (median relative error) for all DCETs

In Figure 4, the following observations can be made:

- Overall, the DCETs tend to underestimate the actual number of defects. This observation is confirmed by the median relative error for the DCETs in Table 5. The same observation was also made in previous studies [5][4][22]. The practical consequence of this underestimation is that a re-inspection decision based on a defect content estimate is "safe" in the sense that if the estimate calls for a re-inspection, a re-inspection is really necessary, since due to the underestimation even more defects remain in the document.
- Those models from the CR models seem preferable that include the source of heterogeneity. This finding corroborates the findings made in other studies [4] [5] [22]. Especially the Mh(JE) estimator shows good properties with a small bias and variability.

– For subjective estimates (SDCE) the bias of the most-likely estimate is lower than the bias for the CR models Mt and M0 and comparable to the curve-fitting model (i.e., the DPM). However, the variability is larger. We attribute the large variability to the lack of experience with the kind of estimation task. Studies with industry practitioners showed more accurate results with smaller variability [14]. We therefore recommend for practitioners to adhere to subjective estimates for making the reinspection decision if data collection for objective DCETs is not in place.[4]

In addition to the most-likely value, we asked the inspectors for estimates of minimum and maximum values. As discussed in section 2.2.1, we ideally expect the maximum value to overestimate and the minimum value to underestimate. However, the maximum estimate is *under*estimating in 50% of all inspections. Therefore, it is not recommendable to rely on this estimate as an upper bound of the defect content.

To test whether the differences are statistically significant, we follow the testing strategy described in section 3.5.

4.2.1 Statistical Testing to Select one CR Model

The Kruskal-Wallis test indicated that there is indeed a statistically significant difference (p=0.0041) between the CRs' median relative error (bias). Therefore, following the testing strategy was a reasonable approach.

4.2.1.1 Selecting one estimator per CR model

For model Mh, two estimators are available (Mh(JE) and Mh(Ch)), also for Model Mt (Mt(MLE) and Mt(Ch). The first step was to determine whether for a specific model one estimator is favorable in terms of bias or variability.

For both models the two available estimators are not statistically different in their absolute bias as shown in Table 6. According to Figure 4, the estimators do not differ in their variability either. Therefore, it was not possible to select one representative estimator per model based on these criteria. Hence, in the following both estimators were considered for each model.

[4] Although a reinspection decision is part of many industrial inspection implementations, it is often not based upon any kind of defect content estimate. Hence, we suggest the subjective approach, since it makes at least one decision criterion for reinspection visible to other members in the development team, such as quality assurance managers.

DCET	U	Z	p-level
Mh(JE) vs. Mh(Ch)	462.5	-0.253	0.800
Mt(MLE) vs. Mt(Ch)	374.5	1.492	0.136

Table 6: Test results for models Mt and Mh

4.2.1.2 Determining the most appropriate model

Figure 7 presents the results of the statistical tests. The nodes represent the models and their estimators with the absolute median relative error (absolute bias). The edges indicate a comparison using the Mann-Whitney U Test. The p values for each (one-tailed) comparison are also given.

As can be seen, the model Mh improves the mean absolute bias over M0 (Mh vs. M0), whereas model Mt adds no improvement to M0 (Mt vs. M0). On the other hand, if the time response source of variation is added to the heterogeneity source of variation (Mh vs. Mth), no significant improvement in the absolute bias can be obtained, while adding heterogeneity to time response (Mt vs. Mth) does improve the absolute bias.

This overall pattern of results suggests that the heterogeneity effect dominates the other sources of variation. We can therefore recommend either model Mh or Mth. Since between the estimators of these models no significant difference can be established, we select Mh(JE) as a representative for CR models, as it was also done in other studies[5] [5][22][30].

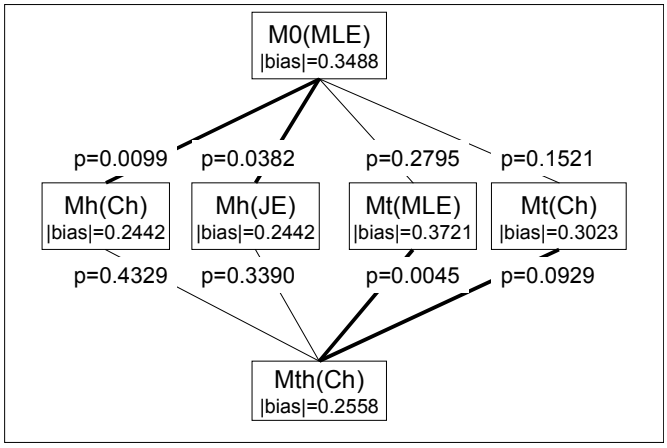


Figure 5: Test results for CR models

[5] When using multiple comparison procedures such as described in [5], there is no statistically significant difference between MH(JE)/M0(MLE) and Mt(Ch)/Mth(Ch). These test results however would not influence the decision to select Mh(JE) as recommended CR estimator.

4.2.2 Statistical Testing to Select a Type of DCET

The next question is whether there is a significant difference between the different types of DCETs: subjective estimates, CR models, and CF models. To test whether there is a difference in the population median of these samples, again, a Kruskal-Wallis test was performed. A p-value of p=0.0004 indicates that in terms of bias there is a statistically significant difference among these DCETs, i.e., we could reject H_0.

The absolute median relative error for these three types of models is shown in Table 7.

DCET	Absolute median relative error
Mh(JE)	0.2442
DPM	0.3126
SDCE(mlv)	0.3140

Table 7: Absolute bias for the three model types

Performing the Mann-Whitney U Test on each of these pairs reveals the results in Table 8. Mh(JE) is significantly better than the other two approaches, whereas the curve-fitting model and the subjective approach are not significantly different. Thus, overall the Mh(JE) seems a reasonable choice over all DCETs[6].

DCET	U	Z	p-level
Mh(JE) vs. DPM	330	2.1188	0.0341
Mh(JE) vs. SDCE	225	-3.597	0.0003
DPM vs. SDCE	478	0.0352	0.9719

Table 8: Comparison between DCETs

It is interesting to note that the subjective approach is not significantly worse than the more objective curve fitting models. In fact, the subjective approach is quite similar to most objective DCETs with respect to the median values. The fact that Mh(JE) outperforms the subjective approach may be based on the larger variation of the subjective approach. The latter might be explained by the fact that the subjects were students and not professional developers, i.e., their inexperience led to large variations in their predictions. Thus, we would expect less variation in the estimates of professional developers. In this case, if there is no established inspection data collection approach for objective DCETs in place, subjective estimates might offer a cost-effective alternative. Some initial empirical underpinning for our expectation is presented in [14].

[6] When using multiple comparison procedures such as described in [5], there is no statistically significant difference between MH(JE)/DPM and DPM/SDCE. The difference between the pair Mh(JE)/SDCE is statistically significant. These test results however do not impact our interpretation.

However, more research on SDCE is required. One question, for example, whether there is a difference when the inspectors provide an estimate for the number or percentage of defects detected or for the defects remaining in the document.

5 Conclusion

This paper examined the combined effects of reading and defect content estimation techniques for software inspections. The examination was based on data collected at an experiment performed at the Technical University of Vienna. The main objective of the investigation was the accuracy of various DCETs when two different reading techniques were used for defect detection. The reading techniques were a scenario–based approach (i.e., a combination of perspective-based reading and traceability-based reading) and checklist-based reading.

In contrast to our expectations we could not observe a difference in the accuracy of defect content estimates between the two reading techniques. These results allow for several conclusions. First, factors other than the reading technique are more important for the accuracy of the DCETs. These need to be identified and studied. Second, there is no difference in the data set although two reading techniques were applied. This requires analysis techniques to determine the crucial characteristics of a data set, e.g., to determine whether scenario-based data were available.

Although further corroborative evidence needs to be collected and replication is necessary, the practical consequences of this study are that practitioners can use DCETs independent of the reading technique used. If an objective model is suggested, Mh(JE) will be the most accurate estimator. Moreover, we found the subjective approach surprisingly accurate in comparison with the more objective DCETs. However, the estimates exhibit large variation, which may result from the lack of experience of our subjects. Since lack of experience is often not an issue in industrial environments, we therefore suggest following the subjective defect content estimation approach if the required data for using objective DCETs, is unavailable.

6 Acknowledgements

We thank Dr. Forrest Shull from the Fraunhofer Center for Experimental Software Engineering Maryland for providing us the necessary information on the various reading techniques. Moreover, we thank Michael Halling for his support in the data analysis, the participants of the experiment, and also Sonnhild Namigha for proofreading this paper.

7 References

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

[2] V.R. Basili. Evolving and Packaging Reading Technologies. Journal of Systems and Software, 38(1), July 1997.

[3] L. Briand, K. El Emam, B. Freimut, A Comparison and Integration of Capture-Recapture Models and the Detection Profile Method, in Proceedings of the 9th International Symposium on Software Reliability Engineering, pp. 32-41, 1998.

[4] L. Briand, K. El Emam, B. Freimut, O. Laitenberger, Quantitative Evaluation of Capture Recapture Models to Control Software Inspections, Proceedings of the 8th International Symposium on Software Reliability Engineering, pp. 234-244, November 1997.

[5] L. Briand, K. El Emam, B. Freimut, O. Laitenberger. A Comprehensive Evaluation of Capture-Recapture Models for Estimating Software Defect Content, IEEE Transactions on Software Engineering, vol. 26, no. 6, pp. 518-540, June 2000.

[6] L. Briand, B. Freimut, F. Vollei, Assessing the cost-effectiveness of inspections by combining project data and expert opinion, Proceedings of the 11th International Symposium on Software Reliability Engineering, October 2000.

[7] G. E. P. Box, W. G. Hunter, J. S. Hunter, Statistics for Experimenters, John Wiley & Sons, 1978

[8] K. P. Burnham and W.S. Overton, Estimation of the size of a closed population when capture probabilities vary among animals, Biometrika, vol. 65, pp. 625-633, 1978.

[9] A. Chao, Estimating Population Size for Sparse Data in Capture-Recapture Experiments, Biometrics, vol. 45, pp. 427-438, June 1989.

[10] A. Chao, Estimating the Population Size for Capture-Recapture Data with Unequal Catchability, Biometrics, vol. 43, pp. 783-791, Dec. 1987.

[11] A. Chao, S.M. Lee, and S.L Jeng, Estimation Population Size for Capture-Recapture Data When Capture Probabilities Vary by Time and Individual Animal, Biometrics, vol. 48, pp. 201-216, Mar. 1992.

[12] B. Curtis. By the Way, Did Anyone Study any Real Programmers? Empirical Studies of Programmers: First Workshop, pp. 256–262. Ablex Publishing Corporation, 1986.

[13] S. G. Eick, C. Loader, M. D. Long, L. G. Votta, and S. Vander Wiel, Estimating Software Fault Content before Coding, in Proceedings of the 14th International Conference on Software Engineering, pp. 59-65, 1992.

[14] K. El Emam, O. Laitenberger, T. Harbich, The Application of Subjective Estimates of Effectiveness to Controlling Software Inspections, Journal of Systems and Software, 54 (2) (2000) pp. 119-136.

[15] M.E. Fagan. Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal, vol. 15, no. 3, pp. 182-211, 1976.

[16] B. Freimut, Capture-Recapture Models to Estimate Software Fault Content, Master's thesis, University of Kaiserslautern, Germany, June 1997.

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

[18] W.S. Humphrey. Introducing the personal software process; Annals of Software Engineering 1; pp. 311-325; 1995

[19] O. Laitenberger, Cost-effective Detection of Software Defects through Perspective-based Inspections. PhD-thesis, University of Kaiserslautern, 2000.

[20] O. Laitenberger, J.M. DeBaud. An Encompassing Life-Cycle Centric Survey of Software Inspection. Journal of Systems and Software, vol. 1, no. 51, 2000.

[21] O. Laitenberger, K. El Emam, K., T. Harbich. An Internally Replicated Quasi-Experimental Comparison of Checklist and Perspective-based Reading of Code Documents. Accepted for Publication in IEEE Transacations on Software Engineering. Also appeared as a Report of the International Software Engineering Research Network, Technical Report ISERN-006-99.

[22] J. Miller, Estimating the number of remaining defects after inspection, Journal of Software Testing, Verifiaction and Reliability, vol. 4, no. 9, pp. 167-189, 1999.

[23] D. L. Otis, K. P. Burnham, G. C. White, and D. R. Anderson, Statistical Inference from Capture Data on Closed Animal Populations, Wildlife Monographs, vol. 62, pp. 1-135, Oct. 1978.

[24] H. Petersson and C. Wohlin, Evaluation of using Capture-Recapture Methods in Software Review Data, Proceedings Conference on Empirical Assessment and Evaluation in Software Engineering, Keele University, Staffordshire, UK, 1999.

[25] H. Petersson and C. Wohlin, Evaluating Defect Content Estimation Rules in Software Inspections, Proceedings Conference on Empirical Assessment and Evaluation in Software Engineering, Keele University, Staffordshire, UK, 2000.

[26] A A. Porter, L. G. Votta, and V. R. Basili, Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment, IEEE Transactions on Software Engineering, vol. 21, pp. 563-575, June 1995.

[27] C. Robson, Real World Research, Blackwell, 1993.

[28] S. Siegel and N. Castellan, Nonparametric Statistics for the Behavioural Sciences, 2nd edition, McGraw Hill, 1988.

[29] T. Thelin and P. Runeson. Capture-Recapture Estimations for Perspective-based Reading – A Simulated Experiment, Proceedings of the International Conference on Product Focused Process Improvement, pp182-200, 1999.

[30] T. Thelin and P. Runeson. Fault Content Estimations using Extended Curve Fitting Models and Model Selection, Proceedings Conference on Empirical Assessment and Evaluation in Software Engineering, Keele University, Staffordshire, UK, 2000.

[31] T. Thelin and P. Runeson, Robust Estimations of Fault Content with Capture-Recapture and Detection Profile Estimators, To appear in Journal of Systems and Software, 2000.

[32] G. Travassos, F. Shull, M. Fredericks, V.R. Basili, Detecting defects in object-oriented designs: Using reading techniques to increase software quality. Proceedings of the International Conference on Object-oriented Programming Systems, Languages & Applications (OOPSLA), 1999.

[33] B.J. Winer, D. R. Brown, K. M. Michels, Statistical Prinicples in Experimental Design, McGraw-Hill Series in Psychology, 3rd ed., 1991.

[34] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslen, Experimenation in Software Engineering - An Introduction. The Kluwer International Series in Software Engineering, Kluwer Academic Publishers, 2000.

[35] C. Wohlin, P. Runeson, J. Brantesham, An Experimental Evaluation of Capture-Recapture in Software Inspections, Journal of Software Testing, Verification and Reliability, Vol. 5, pp. 213-232, 1995.

[36] C. Wohlin and P. Runeson, Defect Content Estimations from Review Data; Proceedings of the International Conference on Software Engineering, pp. 400-409, 1998.

Document Information

Title:	Investigating the Impact of Reading Techniques on the Accuracy of Different Defect Content Estimation Techniques
Date:	April 1, 2001
Report:	IESE-061.00/E
Status:	Final
Distribution:	Public