T. Thelin, H. Petersson, P. Runeson and C. Wohlin, "Applying Sampling to Improve Software Inspections", Journal of Software and Systems, Vol. 73, No. 2, pp. 257-269, 2004.

Applying Sampling to Improve Software Inspections

Thomas Thelin[1], Håkan Petersson[1], Per Runeson[1] and Claes Wohlin[2]

[1]Dept. of Communication Systems,
Lund University
Box 118, SE-221 00 LUND, Sweden
{thomas.thelin, hakan.petersson,
per.runeson}@telecom.lth.se

[2]Dept. of Software Eng. and Computer Science
Blekinge Institute of Technology
Box 520, SE-372 25 Ronneby, Sweden
claes.wohlin@bth.se

Abstract

The main objective of software inspections is to find faults in software documents. The benefits of inspections are reported from researchers as well as software organizations. However, inspections are time consuming and the resources may not be sufficient to inspect all documents. Sampling of documents in inspections provides a systematic solution to select what to be inspected in the case resources are not sufficient to inspect everything. The method presented in this paper uses sampling, inspection and resource scheduling to increase the efficiency of an inspection session. A pre-inspection phase is used in order to determine which documents need most inspection time, i.e. which documents contain most faults. Then, the main inspection is focused on these documents. We describe the sampling method and provide empirical evidence, which indicates that the method is appropriate to use. A Monte Carlo simulation is used to evaluate the proposed method and a case study using industrial data is used to validate the simulation model. Furthermore, we discuss the results and important future research in the area of sampling of software inspections.

Keywords: Empirical study, Monte Carlo Simulation, Software Inspection, Sampling, Efficiency

1. Introduction

Inspection is an efficient technique to detect faults throughout the software development process. Several research papers have reported on the benefits of using inspections (Ebenau and Strauss, 1994; Fagan, 1976; Gilb and Graham, 1993; Weller, 1993). The main purpose of inspections is to detect faults in software documents[1]. Empirical research on inspections focuses on improving them to be more effective and efficient. Different variants of inspection processes have been proposed since the formalization of the first inspection process (Fagan, 1976). In this paper, we assume the existence of an established inspection process in a company with three main phases: *preparation*, *meeting* and *fault correction*. The aim of these phases is fault searching, fault gathering, and fault correction, respectively.

Several aspects making software inspections more efficient and effective have been investigated empirically. The research focuses on three improvement factors for inspections, *lead-time*, *effectiveness* and *efficiency (effort)*. Reduction of lead-time (defined as the calendar time between two points during development) has been evaluated by cancelling the inspection meeting. Efficiency and effectiveness have been investigated by improving the reading technique and by changing the process.

Sampling, i.e. the action of evaluating the quality of an object by means of selecting a subset, can be used in inspections to increase the efficiency of an inspection session when resources are limited. This paper presents and evaluates a method, denoted Sample-Driven Inspections (SDI), that

1. By document, we refer to any kind of artefact developed, e.g. specifications, design, code and test plans.

improves software inspections by the use of sampling. The aim of SDI is to concentrate the inspection effort on the software documents that contain most faults. SDI divides the inspection effort into two parts. First, samples of the documents are inspected in order to estimate which documents contain most faults (*pre-inspection*). Second, an inspection is carried out for the chosen documents (*main inspection*). It is important to notice that SDI does not require any special type of reading technique nor any special preparations for the reviewers. This means that any type of reading technique can be used when applying SDI. This paper investigates the effects of using SDI. Specifically, we investigate how large part of the documents that has to be pre-inspected, how the number of reviewers and their abilities affect the result, and how the characteristics of the faults and the documents affect the result.

Gilb and Graham (1993) and Burr and Owen (1996) have proposed sampling of software documents, although leaving several open research questions, including, for example, sample size and number of reviewers. They suggest inspecting parts of a software document in order to determine whether the document is ready for the main inspection. Gilb and Graham argue that the same types of faults exist in different forms throughout the same document. Hence, a sample of one or some pages would be enough to get a picture of the fault distribution in a software document. Gilb and Graham, as well as Burr and Owen, describe sampling as a way to make inspections more efficient and effective. SDI integrates sampling in a method, which provides software organizations with the opportunity to increase their inspection performance.

Kusumoto et al. (1996) present a study that investigates time allocation for software inspections. The goal of their method is to increase the effectiveness of inspections by allocating more inspection time to documents of low quality and less time to documents of high quality. They use sampling of documents to decide the inspection time for each document. Time is allocated to each document based on the number of faults detected during the sampling and the size of the documents. Their inspection process is based on checklist-based reading and to control the inspection time, they assume that each check item in the checklist take equal amount of time. The time allocation method is evaluated in a comparative case study of two software projects (on code inspections). The conclusion is that the fault detection rate is improved from 28% to 40% by applying sampling. However, since the comparison is based on two software projects it is difficult to control the noise parameters. Hence, a more controlled study is needed as a complement to the case study.

This paper consists of two parts. The first part introduces the concept of SDI, i.e. *sampling*, *pre-inspection*, *resource scheduling* and *main* inspection. The second part is a Monte Carlo simulation that provides an investigation of SDI as an inspection method. The important parameters that are investigated are sample accuracy, the number of reviewers, the ability of the reviewers and the percentage inspected in the pre-inspection (sample size). The simulation parameters are validated with empirical data from industrial inspections. The novel contribution is the combination of the method constituents and the empirical investigation of the method.

The main result from the simulation study is that the method works and is appropriate to use. The results are valid even in case the sample accuracy is low, which indicates that the method is robust enough to be further studied. The empirical data confirms that the parameters chosen for the simulation are relevant. Further work includes experimentation with sampling and effort scheduling, as well as case study applications in industrial settings.

The paper is structured as follows. The Sample-Driven Inspection method and different aspects related to SDI are presented in Section 2. In Section 3, the simulation model is described together with a discussion of the document profiles used, where a document profile refers to how the faults are distributed across the document. The results of the simulations are presented in Section

4. A discussion of the results is presented in Section 5 and finally in Section 6, the main conclusions are presented and some areas for future research are identified.

2. Sample-Driven Inspections: Method Description

2.1 Overview and Motivation

In software development, driven by market forces, projects have to move very fast through the development because of the strong demand of releasing new or upgraded products as often as possible. In order to deliver software with desired quality, different methods for quality control and improvement are used. One such method is software inspections.

Inspections are an integrated part of a company's development process. The project manager, or quality manager, has to take care of the planning of the inspections and consider a number of different questions: *how* to inspect, *what* to inspect, *when* to inspect, and *who* should perform the inspection. Since inspections require resources that otherwise are used for development, the concept of resource scheduling is closely related to performing inspections.

As resources in terms of staff hour and lead time are scarce, project management may have to select a subset of documents to inspect. Instead of selecting documents ad hoc, SDI offers a systematic approach to schedule the available resources to the software documents that need it the most. However, since the quality of the documents is unknown, it has to be estimated. The estimation is achieved by applying sampling to support the scheduling task. In addition to the *inspection process* and *resource scheduling* performed by project management, we add the task of *sampling*. Furthermore, we have split the inspection into two phases, *pre-inspection* and *main inspection*. By pre-inspecting a small sample of all available documents, we estimate which documents have most faults and make sure that these get most attention during the main inspection.

2.2 Sampling and Resource Scheduling

Each of the four parts of SDI, *sampling*, *pre-inspection, resource scheduling* and *main inspection*, allows for a lot of variation. The sampling can be performed using different strategies, the resource decisions, based on the sampling, can be done in various ways and the inspections can be performed using different reading techniques.

The sampling part is of special focus in SDI. It is crucial to whether the method works properly or not. The goal of the sampling is to achieve, with a minimum of resources, a representative sample of each document's quality. This would lead to a correct evaluation of which document that needs most attention during the main inspection. Different sampling strategies are probably needed for different types of software documents. A textual document, for example, a requirements specification or a user guide can be sampled by choosing the number of pages to cover beforehand. While for code or design documents a subset of the functionality may be selected or a time limit be set, and eventually it is estimated how large part was covered.

Based on the outcome of the pre-inspection samples, estimates of the number of faults in the documents are made. The pre-inspection can be conducted by either one reviewer or a group of reviewers. After the pre-inspection, the documents are sorted and resources are distributed for the main inspection. The sorting may be performed using different criteria, e.g. number of faults or fault density; the latter is probably more relevant when the documents are of significantly different size.

In many cases, not only the quality of the document determines how important it is to inspect. Different parts of the product could be more safety critical or contain functionality of higher importance than other parts. The prioritisation of where to spend the inspection effort will then be based on a combination of quality and criticality. Since the criticality of documents vary between companies as well as within a company, the sorting criteria have to be designed for each specific case.

The resources may also be divided in many ways. An example is that different number of reviewers inspects different documents. Another example would be to assign a given amount of inspection time to each document.

2.3 Investigation

It is obvious that the SDI approach will aid the scheduling of resources if representative sampling is possible. Within the mere definition of *representative sample* lies the fact that it gives good enough estimates of the number of faults in the documents. In this paper, we investigate the effect of the SDI approach and its robustness to sample representativeness. The evaluation is performed by running a process simulation (Kellner et al., 1999) of an inspection process. The model used for simulating software inspections is closely related to, and based on, the model used for capture-recapture simulation, see for example Chao (1989), El Emam and Laitenberger (2001) and Otis et al. (1978). The parameters of the simulation model are validated using data from empirical studies, see Section 3.4.

To investigate the effects, we create a conceptual model of how SDI could be implemented within a software organization, see Figure 1. Through simulation of the model, we evaluate different parameter settings. The documents are sampled, and based on a pre-inspection of these samples the resources are scheduled. The scheduling is based on the sorting of the documents, which in turn is based on a defined criterion related to a fault estimate and the main inspection is then performed with the assigned resources. We have identified three types of sorting criteria, 1) the perceived fault content from the pre-inspection – number of faults per document, 2) the fault density – number of faults per page or another size measure, 3) the fault finding efficiency – number of faults found per time unit in the different documents. The fault estimate is denoted C_i.

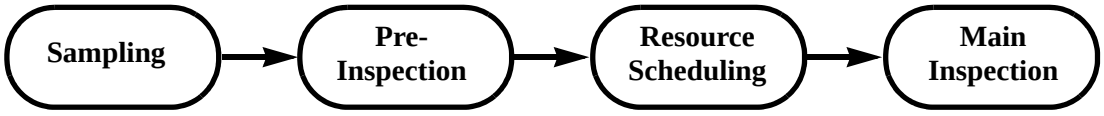

FIGURE 1. Conceptual model of SDI in a company.

The model in Figure 1 is very general and some restrictions have to be made in order to make an investigation possible.

- Faults used in the sorting of the documents are of equal importance – This can be viewed either as if all faults found are used or only the severe faults are used when sorting the documents.
- Documents are of equal importance – This assumption can be viewed from a project manager perspective. He/she could select a number of equally important documents to apply the SDI to.
- Representative sampling of a document is possible – This assumption is a starting point giving good quality estimates for the sorting criterion. However, investigations of SDI's behaviour, when disturbing the representativeness, are performed and further discussed in Section 2.5.

The simulation procedure of SDI can briefly be described as follows. First, a sample of each of the documents is inspected. Second, the documents with the highest estimate of faults are selected and the main inspection is concentrated on these. The following four steps gives a more detailed explanation, see also Figure 2. Notations used throughout the paper are defined inTable 1

1. Of each software document *i*, inspect a sample of size s_i during time t_i. The sample is a percentage share $a_i = s_i / s_{total}$ of the document. Denote the number of faults found ΔD_i.
2. Calculate the fault estimate C_i, either 1) estimated fault content $\Delta D_i / a_i$, 2) fault density $\Delta D_i / s_i$, or 3) fault finding efficiency $\Delta D_i / t_i$.
3. Rank the documents according to the fault estimate.
4. Focus the available inspection effort on the documents with highest ranks according to the fault estimate.

TABLE 1. Notations used in the paper

a_i	Percentage of document *i* inspected
B	Standard deviation of number of faults *between* documents
C_i	Fault estimate for document *i*
d_i	Number of size units within document *i*
ΔD_i	Number of faults detected in pre-inspection of document *i*
D_i	Number of faults detected in pre-inspection and main inspection
E_i	Number of faults exposed to inspection of document *i*
f_n	Probability for exposure of fault *n*
F_{nj}	Probability to find fault *n* for reviewer *j*
r_j	Relative ability of reviewer *j to* find faults
s_i	Sample size of document *i*
S_i	Standard deviation of number of faults per d_i *within* document *i*
t_i	Time used to inspect document *i*
T_i	Number of times the two documents with the highest number of faults are selected for the main inspection

2.4 An Example Scenario

As an example, assume that five design documents are to be inspected. The project manager selects 20% of each documents to be pre-inspected. Since the documents are about 10 pages, he/she selects two of the pages that he/she thinks is appropriate as a sample of the documents. Prior to the pre-inspection, the project manager has decided that only severe faults should be considered when sorting the documents and that the number of faults should be the sorting criterion measure[1].

The five documents contain 5, 5, 10, 15 and 20 severe faults, which are possible to find during inspection. Assume that in the 20% pre-inspection ΔD_i = (0, 1, 2, 1, 3) severe faults are found, and hence the fault estimates are C_i = (0, 5, 10, 5, 15) faults. This leads to ranking the documents

1. In this case with equal time, reviewer effectiveness, document and sample sizes, all three criteria give the same result.

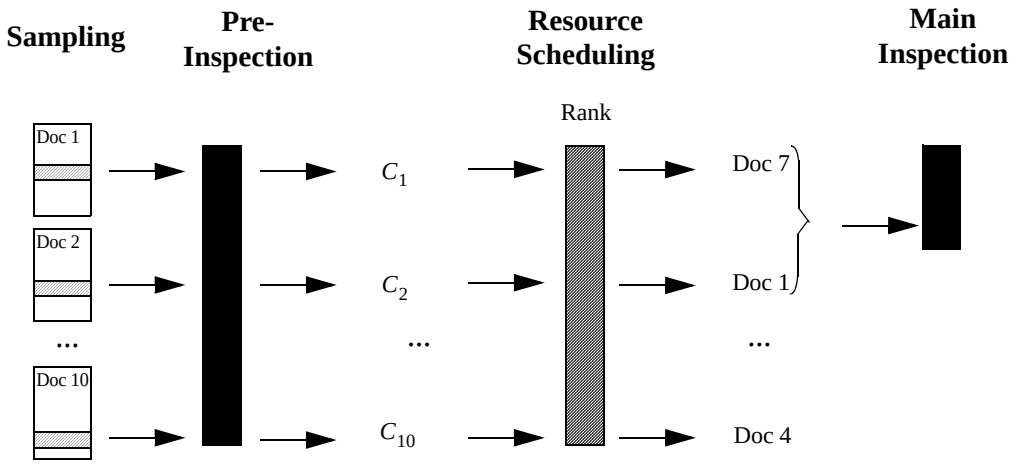

FIGURE 2. The steps in the Sample-Driven Inspection method.

in the order (5, 3, 2, 4, 1), containing (20, 10, 5, 15, 5) true faults. The project manager decides to focus the main inspection on the documents where 3 and 2 faults were found. If we count the approximate number of faults that have been exposed to an inspection, we find E = (0.20*(5+5+10+15+20)+0.80*(10+20)) = 35 faults, compared to the 55 faults exposed when inspecting all documents.

Note that *exposed* includes all faults that exists in the reviewed material not only the ones that the reviewer have found. The number of faults found depends on the effectiveness of the reviewers. However, we assume that the effectiveness is the same in both cases and make relative comparisons. Hence the effectiveness does not change the comparison. Assuming e.g. 55% reviewer effectiveness yields 19 faults detected using sampling versus 30 inspecting all documents.

This means that it is potentially possible to detect about 64% (35/55 or 19/30) of the faults using 52% ((0.20*5+0.80*2)/5*1.0) of the effort, compared to inspecting every document fully or selecting a document to inspect by random. This example shows that important project resources and time could be used more efficiently by the use of SDI.

2.5 Document Profiles

In Section 2.3, the assumption that representative samples are possible to achieve is made. In the simulation, this corresponds to distributing the faults uniformly in the documents as well as randomly selecting what to pre-inspect. To investigate the effect of what happens when disturbing the representativeness, both the uniform distribution and the random selection are removed. We define fault distribution profiles in two dimensions: *within* document profiles and *between* document profiles.

The *within* document profiles are defined as follows. If a document is divided into N equally sized parts, define d_i as the fraction of the total number of faults in part i. Then the standard deviation of $\{d_i, i=1...N\}$ is an approximate measure of how equally distributed the faults are within the document. We denote this measure S_i. Even if the faults have a uniform distribution, the most likely value of S_i is not zero, unless the number of faults were infinite. For example, for a document containing 100 faults with a uniform distribution, S_i has an average of 0.031.

The *between* document profiles define the differences between the different documents. We define similarly B as the standard deviation of the number of faults per document.

To evaluate the robustness towards non-representative sampling, we define the documents to have their faults distributed according to specific within document fault distribution profiles. The profile assigned to a document is randomly chosen. Five profiles, a) to e), are used, see Figure 3. The S_i measure of these profiles are {0.033 0.062 0.100 0.141 0.211}. The mix of the different profiles is 10:5:5:5:1 where a) is the most common and e) is the most unlikely.

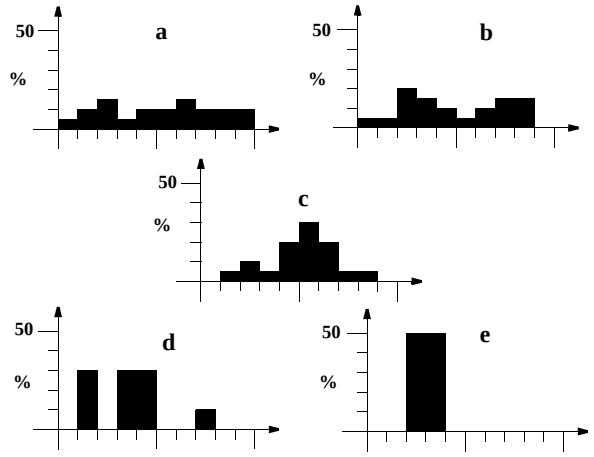

FIGURE 3. Within document fault distribution profiles

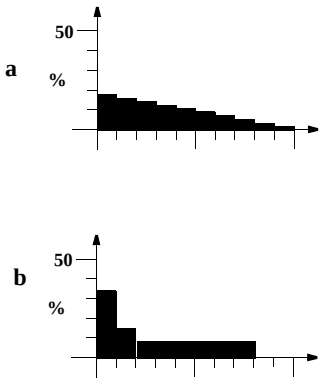

FIGURE 4. Between document fault distribution profile

In the first setup of the simulation, 10 documents are defined to contain 10 to 100 faults in steps of 10, hence the between document profile looks like Figure 4a. In the second setup, six documents are used out of which four contain 30 faults, one 60 faults and one 90 faults, hence the between document profile looks like Figure 4b. The *B* value of these profiles are 0.055 and 0.092 respectively. If the fault density or fault efficiency criteria are used as sorting criterion, the between document profile is defined in terms of the variation in fault density and fault efficiency respectively.

The design and mixture of these profiles do not rely on any specific empirical information of how faults are distributed in documents, but they are validated using empirical data, see Section 3.4. The approach and design are made to study the robustness of the sampling part of SDI. An empirical investigation of how faults are distributed in reality would be of interest, especially when designing the sampling technique to use. The performance of SDI applying these different within and between document profiles is investigated in Section 4.

2.6 Research Questions

Under the assumption that it is possible to achieve representative samples, it is obvious that the SDI approach would help the resource planning. But to what extent? Are the positive effects large enough to justify further research? This study concentrates on investigating whether the method succeeds and how it behaves when changing different parameters. The goal of the simulation study is to answer the following questions:

- How large part of the document has to be pre-inspected in order to make a good enough sorting?
- How does the number of reviewers affect the result?
- How do the abilities of the reviewers affect the result?
- How does the representativeness of the samples affect the result?

3. Simulation Model of Sample-Based Inspections

A Monte Carlo simulation is used to investigate whether the SDI method gives reliable results. Specifically, the simulation is designed to evaluate the research questions stated in Section 2.6. This section describes the simulation model, and explains and validates the parameters used in the model.

The simulation model consists of three parts. The first part is designed to simulate one inspection (Section 3.1). The second part is designed to simulate SDI (Section 3.2). The third part is used to evaluate the SDI method (Section 3.3).

The simulation is based on an inspection model called M_{th} in capture-recapture research (Miller, 1999). The model assumes that reviewers as well as faults are independent. This model was chosen after observing empirical data, showing the M_{th} model to be the most realistic one. A consequence of using this model is that if one reviewer has higher probability to detect a specific fault than other reviewers, he/she has higher detection probability of all faults. Another model that does not have this constraint was under consideration. However, during the model validation, this less constrained model showed to be less realistic using empirical data of the effectiveness.

3.1 Simulation Model of One Inspection

To simulate one inspection, five parameters are considered:

- *Size of a document* – A document is defined to consists of 1000 places, where a fault can be injected in each place.
- *Number of Reviewers* – The number of reviewers for one inspection is part of the simulation model of SDI and is described in Section 3.2.
- *Ability of reviewers* – The abilities of the reviewers represent how good they are to find faults and perform software inspections. The ability of reviewer *j* is denoted r_j. The choices of these abilities are described in Section 3.2.
- *Number of faults* – The number of faults in a document is part of the evaluation model and is described in Section 3.3.
- *Probability of faults* – The detection probabilities of the faults are either assumed to follow a uniform distribution or to be distributed according to certain within document profiles as discussed in Section 2.5. The detection probability for fault *n* is denoted f_n.

Note that the probability to find a specific fault for a specific reviewer is determined by multiplying the ability of the reviewer with the detection probability of the fault ($F_{nj}=f_n \cdot r_j$).

3.2 Simulation Model of SDI

The design of the simulation model of SDI is based on empirical data collected from 30 experiments and case studies in the area of software inspections and capture-recapture. An overview of the simulation model of SDI is shown in Table 2 and Figure 5. Both the pre-inspection and the main inspection parts are simulated.

The simulation model of SDI is designed to study four parameters:

- *Number of reviewers* – The number of reviewers is selected to be one, three or five. The same number of reviewers is used for pre-inspection and main inspection.
- *Percentage of pre-inspection* – To achieve accurate results, five percentages are used to investigate the size of the fractions in the pre-inspection (10%, 20%, 30%, 40% and 50%). A linear

relation is assumed between the sample size and the effort needed to perform the pre-inspection and the main inspection. Since it is assumed that the resources are limited, the percentage values decide the number of documents to be inspected during the main inspection. For example, if 10% of each document is pre-inspected, more documents can be inspected during the main inspection. On the other hand, if 50% of all documents are pre-inspected, a more confident decision can be made of which documents that are fault-prone. However, fewer documents are inspected during the main inspection. In other words, the decision becomes a trade-off between the confidence in pre-inspection and the resources left to conduct the main inspection.

- *Ability of reviewers* – The abilities of the reviewers are divided into three categories, see Table 2. The categories investigated are when all reviewers has a high ability to find faults (All good), all reviewers has a low ability to find faults (All bad) and a mixture of the abilities of the different reviewers (Mixed).

TABLE 2. The three different reviewer cases simulated during the study.

	Ability (r_j)		
	1 Reviewer	**3 Reviewers**	**5 Reviewers**
All Good	0.9	0.8, 0.9, 1	0.8, 0.85, 0.9, 0.95, 1
Mixed	0.6	0.4, 0.6, 0.8	0.3, 0.5, 0.6, 0.7, 0.9
All Bad	0.3	0.2, 0.3, 0.4	0.2, 0.25, 0.3, 0.35, 0.4

- *Document profiles* – The documents have either a uniform distribution of faults, or have the faults distributed according to the within document fault profiles, as presented in Section 2.5.

The simulation varies the four parameters: the number of reviewers, the percentage of pre-inspection, the ability of the reviewers and the within document distribution. This leads to 90 combinations. For each combination, 1000 inspections are simulated in order to get enough evaluation data.

3.3 Evaluation Model of SDI

The evaluation model of SDI has three parameters, *the number of documents, the number of faults* in the documents and a resource limit, which in this paper is called *work points*. We simulate two different setups, one to evaluate the SDI in general (setup I) and one to evaluate the SDI with a few severely fault-prone pages (setup II). Furthermore, to evaluate the success of the method, we measure *the number of exposed faults, the number of found faults* and *the number of times specific documents are selected*.

3.3.1 General Model

Setup I of the simulation model is used to evaluate the general behaviour of SDI. When the pre-inspection is conducted, the documents are sorted in descending order in terms of estimated faults in the document, see Section 2. To evaluate whether the method sorts the documents in correct order, 10 documents are used, which contain 10 to 100 faults in steps of 10, i.e. a total of 550 faults, see Figure 5.

The optimal result would be if the documents were sorted in correct order. Hence, the documents containing most faults can be selected for the main inspection. The success of the method is, however, not dependent of an exactly correct ranking. It is sufficient if the documents are sorted so that the correct documents are selected for the main inspection.

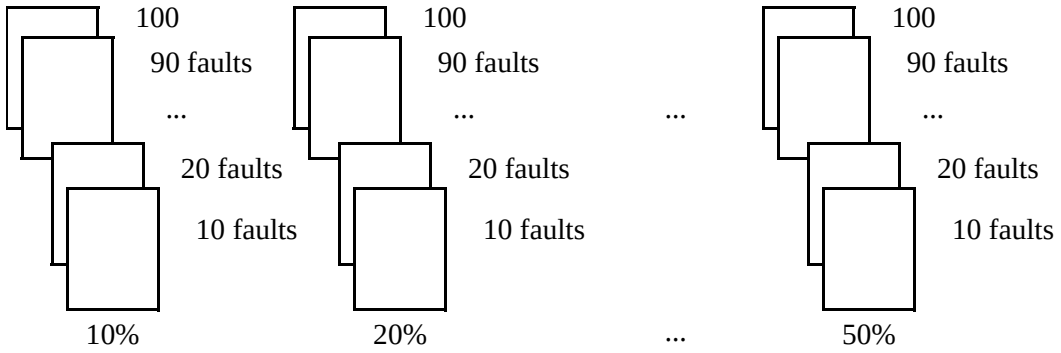

FIGURE 5. An overview of the simulation model. The percentages are varied from 10% to 50%. The faults are varied from 10 to 100 faults.

To compare the parameters investigated, work points are used as a means for available resources. The different work points used are:

- 100 points – the resources used for inspecting 100% of one document.
- 550 points – the available resources in one inspection for pre-inspection and main inspection.

Ten documents are to be inspected in one inspection. This means that 5.5 documents out of 10 can be inspected. If 10% is used as a sample, 10% of 5 documents are inspected plus, 100% of the 5 documents estimated to be most fault-prone. In the 50% case, 50% of 9 documents are inspected plus 100% of one document. If two or more documents are predicted to be equally fault-prone, the remaining work points are distributed equally among these. The same amount of work points is used in the cases of one, three and five reviewers in the pre-inspection.

3.3.2 Fault-Prone Model

Setup II of the simulations is designed to investigate how well SDI manages to identify severely fault-prone documents, i.e. non-uniform *between* document profiles, see Figure 4b. Six documents are used out of which four contained 30 faults, one 60 faults and one 90 faults. For this simulation, 350 work points are used instead of 550, which means that when 30% is used for pre-inspection, two documents are selected for the main inspection.

The other simulation parameters regarding within document profiles, number of reviewers, reviewer abilities and percentage pre-inspection were varied in the same way as in setup I.

3.3.3 Evaluation Measures

The evaluation is carried out by measuring the faults that could be found (exposed) and the faults that are actually found (detected) in a main inspection.

- Exposed Faults – E_i – This is a measure of the quality of the document sorting. The measure counts the faults that are exposed in the pre-inspection and the main inspection. In the pre-inspection, a sample of all documents are used and in the main inspection, only the documents that are predicted to contain most faults are used.
- Detected Faults – D_i –This is a measure of the quality of the whole procedure and considers both the pre-inspection sampling and the main inspection. The measure counts the faults that are actually detected during the pre-inspection and the main inspection.

- Number of Times Selected – T_i – This is a measure for the fault-prone document evaluation and considers the number of times the two documents with the highest number of faults are selected for the main inspection.

3.3.4 Summary of Simulation Model

The above parameters are combined into two simulation setups, summarized inTable 3.

TABLE 3. Overview of simulation setups

Setup	I: General Model	II: Fault-Prone Model
Purpose	General evaluation	Robustness evaluation
Number of documents	10	6
Number of faults per document	10-20-...-90-100	30-30-30-30-60-90
Probability of faults	[Uniform, Profiles]	[Uniform, Profiles]
Number of reviewers	[1, 3, 5]	[1, 3, 5]
Reviewer abilities	[All Good; Mixed; All Bad]	[All Good; Mixed; All Bad]
Percentage of pre-inspection	[10%, 20%, 30%, 40%, 50%]	[10%, 20%, 30%, 40%, 50%]
Working points	550	350

3.4 Validation of Simulation Model

The parameters of the simulation models are selected to be in a the range of what real world data would provide. There is no single source of the parameters, but they are chosen and validated based on empirical studies in the field of software inspections.

The choice of the abilities of the reviewers was made through an investigation of 30 data sets from empirical studies. The mean of the F_{ij} values of these data sets was found to be 0.27. In order to resemble this characteristic, the mean of the abilities in the mixed case is set to 0.6. Since the probabilities of the faults are from the uniform distribution[1], this leads to the mean of the generated data sets in this case is F_{nj} = 0.6 * 0.5 = 0.3. For the other cases, lower respectively higher mean values are chosen. The data sets were collected from different types of software inspections, for example, perspective-based reading (Basili et al., 1996), checklist-based reading (Ebenau and Strauss, 1994), ad hoc inspections, and on different types of software documents, for example, requirements and code documents. The data sets are further described by Petersson and Wohlin (1999).

The document profiles are validated using data from an industrial case study comprising inspections of three requirements documents. The documents are real project documents and the data are collected for research purposes in the evaluation of a new inspection method to be used at the company (Berling and Runeson, 2000). The documents are 12, 33 and 55 pages respectively and contains 111, 284 and 555 requirements items respectively. In total, 29, 123 and 141 faults were identified during the inspection.

The faults are quite evenly spread over the documents, most similar to document profile *a)* in Figure 3. The S_i values are calculated for each of the three documents, S_i = {0.088 0.020 0.014} respectively. Hence, the values are in the lower range of the profiles used in the simulations. The

1. The uniform distribution gives a mean of r_i at 0.5.

differences between the documents are larger, with a between document profile *B* value of 0.21, compared to 0.055 and 0.093 for the simulation models. Furthermore, the number of faults found per page is 2.4, 3.7 and 2.6 respectively. This is also within the range of the simulations, which is 1 to 10 faults per page.

The SDI is negatively affected by a large S_i value, i.e. a large variation within a document makes the estimates of the fault content more unsure. Furthermore, large *B* values, i.e. differences between the document are in favour of the SDI approach, as this makes it more worthwhile selecting more fault-prone documents over less fault-prone ones. Hence, it can be concluded from the case study data that the simulation model parameters are in a reasonable magnitude of order, and they are in the conservative direction seen from the SDI approach.

4. Evaluation of Sample-Based Inspections

In this section, the results and observations are presented. As the study is based on a simulation, no statistical tests are performed, since significance of existing differences in a simulation study can be shown at any statistical level by just increasing the sample size. The three evaluation criteria, exposed faults, detected faults and number of times selected, are presented in subsequent subsections. The analysis of the pre-inspection (Section 4.1) and the main inspection (Section 4.2) are based on setup I of the simulation model, while the fault-prone document analysis (Section 4.3) is based on setup II.

4.1 Pre-inspection

In Figure 6 and Figure 7, boxplots show the results of the SDI, assuming that the samples are representative, i.e. the within document fault distribution is uniform. This corresponds to the setup defined in Section 3.3.1. The results of having one and three reviewers in the pre-inspection phase are presented. The results for five reviewers are similar. The mean number of exposed faults when choosing documents by random is 302.5, i.e. on average 55% of the 550 faults are exposed. This value is shown as a straight line in the boxplots. An inspection covering all documents would find 550 faults (100%) using 100% of the effort.

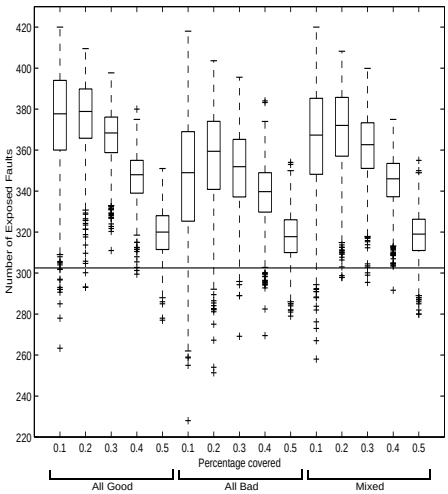

FIGURE 6. Number of exposed faults for one reviewer **(E_i)**, setup I, uniform faults.

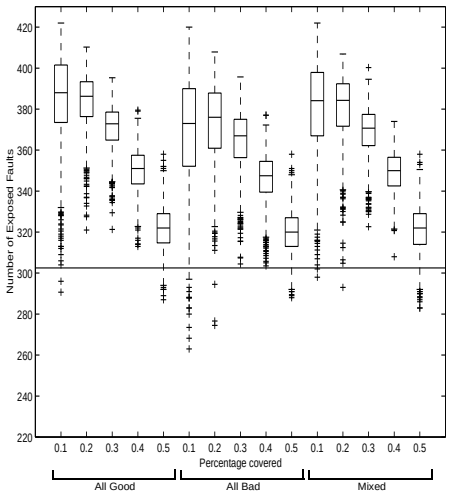

FIGURE 7. Number of exposed faults for three reviewers **(E_i)**, setup I, uniform faults.

A number of observations can be made from the boxplots:

- The higher percentage pre-inspected, the less dispersion. This reflects the fact that a higher percentage leads to a more confident classification. However, less resources are left for the main inspection.
- The more reviewers used, the more reliable results. However, the results show that the differences are very small. This points in the direction that few reviewers are needed in the pre-inspection.
- The higher ability of the reviewers, the less dispersion is obtained. This is a consequence of that competent reviewers find more faults. Hence, the sorting of the documents becomes more reliable.
- For the mixed case with a pre-inspection of 20%, in median 68% of the faults are exposed. This means that using 55% of the effort, 68% of the faults can be found during the inspection. Consequently, SDI exposes a higher percentage of the faults than percentage of the effort used.

Figure 8 and Figure 9 show boxplots of the simulations, using non-uniform within document fault distribution profiles, see Section 2.5.

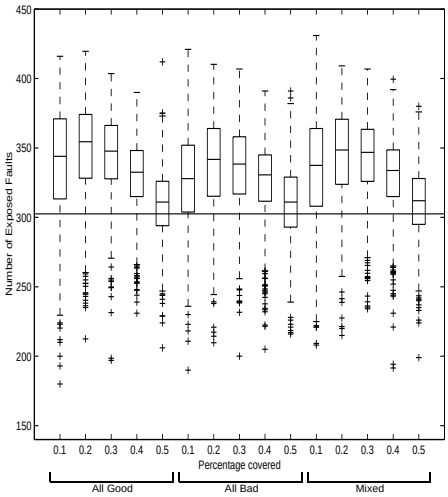

FIGURE 8. Number of exposed faults (E_i) for one reviewer, setup I, fault profiles.

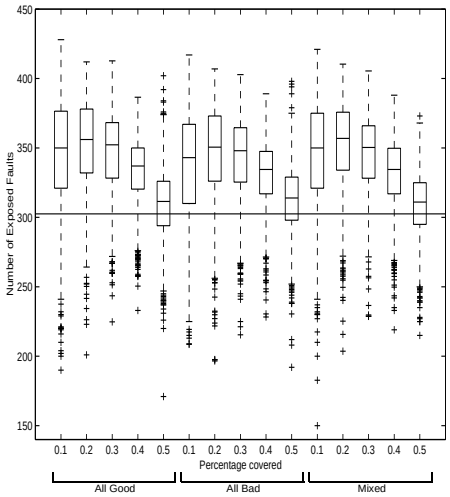

FIGURE 9. Number of exposed faults (E_i) for three reviewers, setup I, fault profiles.

The results are not as good as when using uniform profiles. However, the method is robust to deviations from the assumption of representative samples. For the mixed case, pre-inspecting 20%, 63% of the faults are exposed (using 55% of the effort). For some cases the reduction of exposed faults are larger. However, in neither of the cases the median of the sampling approach is less than the median value when not using sampling.

The simulation of the non-uniform within document profiles shows that sampling is important. Although SDI provides smaller profit when the samples are less representative, it is still better compared to selecting the documents by random.

4.2 Main Inspection

The effect of the abilities of the reviewers is not large when investigating the number of exposed faults. This is a consequence of the robustness of SDI, i.e. as long as the reviewers have equal abilities for all documents, the order of the sorting will not be affected.

However, the main inspection is affected by the abilities of the reviewer. This effect is shown in Figure 10 and Figure 11, based on the setup defined in Section 3.3.1. In these figures, the total number of faults found from both the pre-inspection and the main inspection is shown. Hence, the ability of the reviewers is very important for the success of an inspection. However, the ability does not affect the pre-inspection part of SDI much.

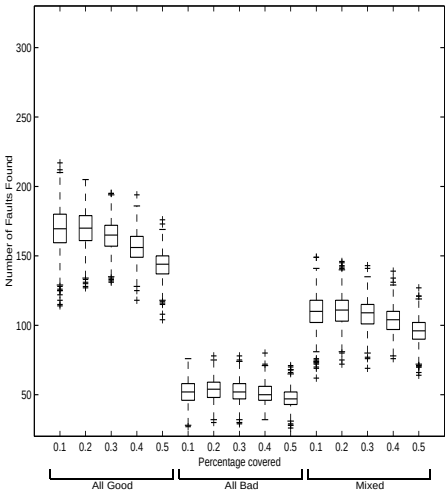

FIGURE 10. Number of found faults for one reviewer **(D_i)**, setup I, uniform faults.

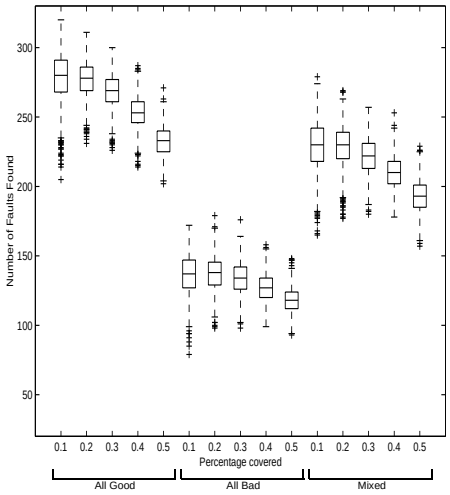

FIGURE 11. Number of found faults for three reviewers **(D_i)**, setup I, uniform faults.

4.3 Fault-Prone Documents

The aim of SDI is to focus inspections on the documents that contain most faults. Sometimes documents under inspections contain many faults and should be corrected before further inspections. Furthermore, some software modules may contain much more faults than others and thereby cause problem later on during inspection, development and maintenance. The simulation setup II corresponds to this case.

The evaluation was carried out by counting the number of times the documents with 60 and 90 faults are selected (T_i). For the mixed case, pre-inspection of 30% shows best result. One reviewer pre-inspecting 30% selects the document with 90 faults 98% of the times and the document with 60 faults 80% of the times. These figures become slightly better when using 3 and 5 reviewers.

For the same prerequisites as above and using the profiles, the documents with 90 and 60 faults are selected 88% and 74% of the times, respectively. In summary, SDI performs very well when the differences in quality between the documents are large. If the differences are small, it is more difficult to sort the documents in a correct order. On the other hand, it is also much less critical. Thus, SDI works well when it is most needed.

5. Discussion

5.1 Research Questions

In Section 2.6, four research questions were posed for this study.

- How large part of the document has to be pre-inspected in order to make a good enough sorting?
- How does the number of reviewers affect the result?
- How do the abilities of the reviewers affect the result?
- How does the representativeness of the samples affect the result?

As shown in Section 4, almost all values lie above the mean of what would be the case if the same effort was spent randomly among the documents. This leads to the conclusion that the SDI method does work.

The largest variation in the simulation results comes from the choice of the sample size. All of the investigated sample sizes give on average better results than the random approach. Even inspecting only 10% gives a fairly good estimate of the number of faults in each document and is better than choosing documents randomly. The choice of sample size is also a matter of trade-off between mean and dispersion. Pre-inspecting 50% results in a very small dispersion but not much improvement in mean compared to the 10% case.

Regarding questions two and three, neither the number of reviewers nor their abilities affect the outcome in terms of number of exposed faults very much. The dispersion in the three-reviewer case is smaller and the median is somewhat better compared to the one-reviewer case. The same can be seen when comparing the *all good* reviewer case with the *all bad* reviewer case. This is expected since the risk of missing a fault is smaller the more and better reviewers that participate in the pre-inspection.

The last question is more difficult to answer. It is obvious when disturbing the sampling that the yield from SDI becomes smaller. In the case study the data is well in the conservative range of the simulated data, i.e. are more evenly distributed. Nevertheless, SDI pays off. In a real-world inspection, the yield of SDI will be a tussle between how the faults are distributed and how well the sampling technique works. However, with the circumstances used in our simulation and in the case study, SDI still delivers better results than merely selecting the documents by random.

5.2 Applying SDI

One aspect contributing to the robustness of SDI is that neither the exact estimate of the faults nor the complete ranking is used. The only important issue is that the correct documents are selected for the main inspection. For example, in the 10% simulation case, five documents are chosen for the main inspection, which means that the selection is only a matter of dividing the documents into two equally sized groups, one to be inspected and one not.

Even if the number of reviewers and their abilities do not affect the scores of exposed faults very much, it does affect the actual outcome of the inspection, as can be seen in Section 4.2. When it comes to actually finding the faults, it is very important how many and how good the reviewers are. It must be remembered that SDI does not improve the reviewers' chance of finding faults; it makes the best out of the given situation.

As mentioned earlier, the key to success lies in the sampling. The application of SDI relies on that organizations learn how to select a representative sample. This could be made either through random sampling or through selecting an appropriate part of the document, where appropriate means a part that is close to the mean quality of the document. A related issue is to study the effect of first deciding on what sample to pre-inspect and then determine the size of that sample. These matters ought to be further investigated in a series of controlled experiments where different approaches of sampling as well as the SDI approach as such can be evaluated. It is likely that

the sampling method will be highly dependent on the type of document as well as the organization using the method. The actual sampling procedure probably has to be calibrated for the given situation.

An early recommendation, based on the results in this study, is to use one reviewer to pre-inspect 20-30% of the document and then decide which documents to focus on in the main inspection. In the simulation, 20-30% managed to expose in average 70% of the total number faults. This yield is dependent on the distribution of faults among the documents as shown in Section 4.1. A rather common case in software development is the 20-80 rule. This rule means that often 20% of the modules contain 80% of the faults. The Pareto effect has been shown empirically in, for example by Ohlsson et al. (1996), although in this study 20% of the modules accounted for 60% of the faults. In our study of particularly fault-prone documents, SDI is very successful in the case when the quality differences are large.

This simulation study in combination with the study by Kusumoto et al. (1996) show that sampling in software inspections is a promising method for software projects where the time or quality factor is important. The result from these investigations show that the detection rate can be increased. Applying the method also give an opportunity for inspection moderators to control the lead-time and effort for an inspection session and thereby make it easier to plan. Hence, sampling techniques are useful for software organizations. MacFarland et al. (2001) describe sampling of inspections as one improvement factor for software inspections in a large telecommunications company.

6. Summary

In this paper, a method to allocate the inspection resources efficiently is presented and evaluated. The method deals with the case of having a number of documents to inspect but not the resources available to cover them all. The main parts of the method are the pre-inspection phase and the main inspection phase. During the pre-inspection, a sample of a number of documents is inspected in order to sort them in terms of fault-proneness. Then, during the main inspection, the documents predicted to be most fault-prone are inspected. The number of documents inspected during the main inspection is determined by the available resources. The method, however, does require neither the limitation of resources to be decided beforehand nor any special technique for the inspection.

The method was evaluated using Monte Carlo simulations and the simulation results were validated in a requirements inspection case study. The results can be summarized as follow:

- The method works and gives an advantage over merely choosing software documents by random.
- The method is robust, simple to use and gives reliable results.
- The recommendation is to use one reviewer to pre-inspect 20-30% of the software documents.
- The method shows even more promising results when it comes to identify fault-prone software documents.

To summarize, SDI works better when the differences in the quality between the documents are large and worse when the differences are smaller. Thus, SDI is best when it is needed the most.

Further work includes running controlled experiments in order to evaluate the method without the limitations of the simulation or case study. The focus of such experiments would be to investigate if the SDI approach works in a real setting with controllable noise factors as well as investigate different ways of solving the representative sample problem.

Acknowledgements

Thanks to Mr. Tomas Berling for his work digging out the data from the case study. This work was partly funded by The Swedish National Agency for Innovation Systems (VINNOVA), under a grant for the Center for Applied Software Research at Lund University (LUCAS). The work was also in part supported by The Fulbright Commission, the Royal Swedish Academy of Sciences and The Royal Physiographic Society in Lund.

References

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

Berling, T. and Runeson, P., 2001. Evaluation of a Perspective Based Review Method Performed in an Industrial Setting, *Proc. of the 5th International Conference on Empirical Assessment & Evaluation in Software Engineering*.

Burr, A. and Owen, M., 1996. *Statistical Methods for Software Quality – Using Metrics for Process Improvements*, International Thomson Computer Press, UK.

Chao, A, 1989. Estimating Population Size for Sparse Data in Capture-Recapture Experiments, *Biometrics* 45, pp. 427-438.

Ebenau, R.G. and Strauss, S. H., 1994. *Software Inspection Process*, McGraw-Hill, New York.

Eick, S. G., Loader, C. R., Long, M. D., Votta, L. G. and Vander Wiel, S. A.,1992. Estimating Software Fault Content Before Coding, *Proc. of the 14th International Conference on Software Engineering*, pp. 59-65.

El Emam, K. and Laitenberger, O., 2001. Evaluating Capture-Recapture Models With Two Inspectors, *IEEE Transactions on Software Engineering*, 27 (9): 851 -864.

Fagan, M. E., 1976. Design and Code Inspections to Reduce Errors in Program Development, *IBM System Journal*, 15(3):182-211.

Gilb, T. and Graham, D., 1993. *Software Inspections*, Addison-Wesley, UK.

Kellner, M. I., Madachy, R. J., Raffo, D., M., 1999. Software Process Simulation Modeling: Why? What? How?, *Journal of Systems and Software*, 46(2/3):91-105.

Kit, E., 1995. *Software Testing in the Real World – Improving the Process*, Addison-Wesley, USA.

Kusumoto, S., Kikuno, T., Matsumoto, K-I. and Torii, K., 1996. Experimental Evaluation of Time Allocation Procedure for Technical Reviews, *Journal of Systems and Software*, 35(2):119-126.

MacFarland, R., Golubic, S. and Grenner, D., 2001. An Improvement Program Spreading Effective Use of Reviews and Inspections throughout a Multinational Telecommunications Company, *Proc. of Workshop on Inspection in Software Engineering*, pp. 127-135.

Miller, J., 1999. Estimating the Number of Remaining Defects after Inspection, *Software Testing, Verification and Reliability*, 9(4):167-189.

Ohlsson, N. Helander, M. and Wohlin, C., 1996. Quality Improvement by Identification of Fault-Prone Modules using Software Design Metrics, *Proc. of the 6th International Conference on Software Quality*, pp. 1-13.

Otis, D. L., Burnham, K. P., White, G. C. and Anderson, D. R., 1978. Statistical Inference from Capture Data on Closed Animal Populations, *Wildlife Monographs* 62.

Petersson, H. and Wohlin, C.,1999. An Empirical Study of Experience-based Software Defect Content Estimation Methods, *Proc. of the 10:th International Symposium on Software Reliability Engineering*, pp. 126-135.

Weller, E. F.,1993. Lessons from Three Years of Inspection Data, *IEEE Software*, 10(5):38-45.