The Impact of Time Controlled Reading on Software Inspection Effectiveness and Efficiency

A Controlled Experiment

Kai Petersen *
School of Engineering
Blekinge Institute of Technology
Box 520
SE-372 25 Ronneby, Sweden
kai.petersen@bth.se

Kari Rönkkö
School of Engineering
Blekinge Institute of Technology
Box 520
SE-372 25 Ronneby, Sweden
kari.ronkko@bth.se

Claes Wohlin
School of Engineering
Blekinge Institute of Technology
Box 520
SE-372 25 Ronneby, Sweden
claes.wohlin@bth.se

ABSTRACT

Reading techniques help to guide reviewers during individual software inspections. In this experiment, we completely transfer the principle of statistical usage testing to inspection reading techniques for the first time. Statistical usage testing relies on a usage profile to determine how intensively certain parts of the system shall be tested from the users' perspective. Usage-based reading applies statistical usage testing principles by utilizing prioritized use cases as a driver for inspecting software artifacts (e.g., design). In order to reflect how intensively certain use cases should be inspected, time budgets are introduced to usage-based reading where a maximum inspection time is assigned to each use case. High priority use cases receive more time than low priority use cases. A controlled experiment is conducted with 23 Software Engineering M.Sc. students inspecting a design document. In this experiment, usage-based reading without time budgets is compared with time controlled usage-based reading. The result of the experiment is that time budgets do not significantly improve inspection performance. In conclusion, it is sufficient to only use prioritized use cases to successfully transfer statistical usage testing to inspections.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.5 [**Software Engineering**]: Testing and Debugging—*Code inspections and walk-throughs*

General Terms

Experiment

*The author is also affiliated with Ericsson AB, Box 518, SE-37123 Karlskrona, Sweden, kai.petersen@ericsson.com

ESEM'08, October 9–10, 2008, Kaiserslautern, Germany..

Keywords

Software Inspection, Time Controlled Usage-Based Reading, Efficiency, Effectiveness, Experiment

1. INTRODUCTION

Software verification and validation techniques are divided into two major groups, namely static and dynamic [19]. Static techniques do not depend on executable software artifacts, whereas dynamic techniques require executable programs. Software inspections are static verification and validation techniques. Alongside reviews and walkthroughs, they provide a visual examination of manifold software artifacts such as requirements specifications, test plans, designs, use cases and code. They offer high potential to discover faults early because software inspections can be applied on all artifacts produced within the software development life cycle. Thus, the removal of those faults is cheaper because less rework has to be done. Aurum et al. [3] surveyed software inspection research of the last 25 years and found that researchers agree that software inspections are an efficient fault detection technique, which helps to improve software quality and reduce development costs. An important means to support software inspections is to guide reviewers through reading techniques during the individual inspection, for example using perspective based reading [4][11].

The idea of usage-based reading [21] is to transfer ideas from operational profile testing [18] and statistical usage testing [26][15] to inspections. Statistical usage testing determines how intensively certain functions of the system should be tested based on a usage profile. Functions that are important and frequently used from the users' perspective are tested more intensively. So far, usage-based reading has incorporated this by prioritizing use cases that drive the inspection. That means the reviewer starts with the use case of the highest priority and checks whether the artifact under inspection (for example design) fulfills the use case. However, this only partly transfers the principles of statistical usage testing to inspections because only the order of the use cases is considered, not how intensively they should be inspected. In order to completely transfer the concept of statistical usage testing to inspections, we propose to assign time budgets to the use cases. The higher the priority of the use case (based on frequency of use and importance to the

user) the more time is assigned to the use case. This principle is referred to as time controlled usage-based reading (TC-UBR). In addition, the time budgets assure coverage of the use cases. That is, within a limited inspection time (e.g., two hours) the reviewer cannot neglect medium and low ranked use cases.

Usage-based reading with prioritized use cases (RB-UBR) is compared with time controlled reading usage-based (TC-UBR) through a controlled experiment with 23 Software Engineering M.Sc. students at Blekinge Institute of Technology. In order to compare the two reading techniques, the students were randomly assigned to the techniques to form the treatment group (TC-UBR) and control group (RB-UBR). The following research questions are answered in this experiment:

- *Q1:* Is TC-UBR more effective than RB-UBR?
- *Q2:* Is TC-UBR more efficient than RB-UBR?
- *Q3:* Are different faults detected for TC-UBR and RB-UBR?

The performance of the groups is measured in terms of effectiveness, efficiency and fault content, defined as follows:

- Effectiveness: Number of faults found in relation to the total number of existing faults
- Efficiency: Faults found per time unit (hour)
- Fault content: Classification of faults based on severance from the users' perspective

These measurements have been used in previous reading technique experiments as well (for example [21][24][20]).

The paper is structured as follows: Section 2 provides an overview of related work. In Section 3 the reading techniques TC-UBR and RB-UBR are presented. Section 4 illustrates the design and operation of the experiment. Thereafter, Section 5 encompasses the collected data analysis. Section 6 discusses the results of the experiment. Finally, Section 7 presents the conclusions and suggestions for future work.

2. RELATED WORK

The original software inspection process was proposed by Fagan in his widely recognized IBM article from 1976 [7]. This process was the basis for further work done on software inspections. Since then two major areas of research have developed [3]. First, researchers investigate changes in structure of the original process proposed by Fagan (e.g., two person inspection [5], N-fold inspection [16], inspection without meeting [25]). Secondly, researchers explore different new ways on how to support particular phases of the inspection process (e. g., reading techniques [4][11], electronic support [17][8], re-inspection support [6][29]). Our research contributes to the second group.

Usage-based reading was evaluated in a series of experiments. In the first experiment the basic principle of this reading technique was investigated by comparing a random order of use cases with prioritized use cases [21]. The subjects were 27 Bachelor students at Lund University inspecting a design document. The experiment showed that prioritized use cases lead to significantly higher effectiveness and efficiency at a significance level of $\alpha = 0.05$. Furthermore, different sets of faults are found.

Thereafter, usage-based reading was compared with checklist based reading in an experiment with 23 Master students at Blekinge Institute of Technology inspecting the same design document as in the first experiment [22]. In terms of effectiveness and efficiency, usage-based reading is significantly better. Furthermore, different faults are found by both techniques. The experiment presented in [22] has been replicated two times. The first replication was done at Blekinge Institute of Technology with 62 Master students using the same experimental package [20]. The results of the first replication also show significantly better results for usage-based reading in terms of effectiveness, efficiency and fault content. A second replication of the experiment was conducted at Vienna University with 131 students, including Master and Bachelor students [27]. The results also significantly point in favor of usage-based reading.

Another experiment evaluated usage-based reading in comparison to usage-based testing [2]. The findings are that usage-based reading is more efficient and effective, but does not find different faults. Furthermore, modifications to usage-based reading have been investigated [24][27]. However, these modifications (like students have to prioritize the use cases by themselves in the study of [27]) did not lead to significant improvements.

3. TIME CONTROLLED READING

Usage-based reading was created out of the testing methods usage-based testing [28] and operational profile testing [18], i. e., the principles applied by those techniques were transferred to inspections [21]. Traditional reading techniques, like checklist based reading [12] or perspective based reading [4][12], aimed at finding as many faults as possible while neglecting the relative significance of faults. Reviewers applying usage-based reading focus their inspection effort on those parts of the document that are most crucial from the users' perspective [24]. However, since usage-based reading requires a set of prioritized use cases Thelin et al. [24] propose to use this technique in projects where the use cases are already created.

How usage-based reading works in detail is described in the following. In order to focus the inspection effort on the most important parts of the document from the users' perspective, use cases are utilized as shown in Figure 1. The circles 1 to 5 represent the steps to be conducted during the RB-UBR reading process. Step 1 is related to pre-inspection activities (where the use cases are prioritized) while step 2 to 5 are conducted by the reviewer during the individual inspection.

The steps cover the following activities for RB-UBR [23].

1. The use cases are prioritized from the perspective of the user (see circle no. 1 in Figure 1). The prioritization of the use cases could be achieved using prioritization techniques like the Analytical Hierarchy Process (AHP) [9] or the cumulative voting approach [14].
2. After prioritizing the use cases the reviewer starts with the use case of highest priority.
3. The reviewer traces the parts that are related to the specific use case within the document under inspection (in this example the design of software X, see Figure

1) by following the tasks provided by the use case. A task is either an action (of the system or the user) or a certain condition (for example, driver in state "available").

4. While following the tasks through the document under inspection, the reviewer checks if the software artifact provides complete and correct information so that the goals of the use case are fulfilled. For the design example, the reviewer might check if the interfaces are correct (e. g. parameters of a signal) and if all necessary signals are provided so that the goal of the use case could be achieved.

5. When the reviewer thinks he has completed the inspection of the use case, he selects the use case with the next highest priority and repeats the steps 3 to 5.

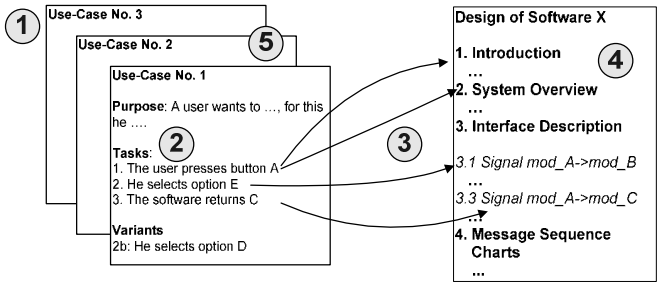

Figure 1: Applying RB-UBR

TC-UBR requires a time budget assigned to each of the different use cases where the more important use cases get assigned more time than the less important ones. Hence, one does not just require the prioritized order of the use cases, but also must derive the time from the prioritization methods. Consequently, changes have to be made to the different steps outlined before so that the time budget is taken into consideration. In the case of TC-UBR the steps are:

1. When using AHP as a prioritization method, the times are calculated in the following way: Derive the relative importance p_i where $(0 \leq p_i \leq 1)$ and $\sum p_i = 1$ for each use case u_i from the prioritization data collected during AHP. In practice, the relative importance is known because people using AHP compare all unique pairs of requirements with each other and judge which one is of higher importance and also to what extend. After that, determine the total time available for the inspection (T). Based on this, calculate the time spent on each use case (t_i) by multiplying p_i with the total inspection time T. In the case of cumulative voting, the same approach can be followed. That is, multiply T with the number of dollars d_i assigned to each use case u_i and divide them by D (total mount of total dollars available).

2. as RB-UBR

3. The reviewer traces the parts related to the use case within the document under inspection for no longer than the duration assigned to the use case.

4. as RB-UBR

5. When the time is up or the reviewer has completely inspected the use case before that, she selects the use case with the next highest priority and repeats steps 3 to 5.

4. EXPERIMENT DESIGN

In this section, the design of the experiment (definition, variables, treatments, and instruments) as well as the operation of the experiment are explained.

4.1 Experiment Definition

The experiment is defined based on the template proposed in [30] as:

- Analyze *TC-UBR and RB-UBR* for the purpose of *evaluation*,
- with respect to *effectiveness, efficiency and fault content*,
- from the point of view of *the researcher*,
- in the context of *Software Engineering M.Sc. students inspecting a design document*.

4.2 Preparation and Planning of the Experiment

The experiment plan comprises sample selection and commitment, description of the experimental package, definition of variables, statement of hypotheses and description of design principles.

4.2.1 Sample Selection and Commitment

The subjects of the study are selected using convenience sampling, i.e., easily accessible persons are selected [30]. The sample consisted of 23 Software Engineering M.Sc. students at Blekinge Institute of Technology in Sweden. The experiment was conducted in the context of a course on software project management. An experience questionnaire was given to the students prior to the experiment in order to determine how experienced they were in the topics relevant to the experiment. Examples are experience in the topic areas design inspections, programming and use cases. The questions were asked on an ordinal scale. For example, the scale for use cases was 1) Never used use cases, 2) Used use cases in courses and 3) Industrial experience in using use cases. The result of the questionnaire showed that the experience of the students can be characterized as heterogeneous because most of them are foreign students from all over the world. Consequently, they have gained different educational as well as different practical experience. However, they all have at least one year experience from software projects. This is a prerequisite to be accepted to the master program. The students are classified into three groups, 1) highly experienced (experience from industry in at least two relevant topic areas), 2) experienced (experience from industry in one relevant topic area) and 3) less experienced (experiences gained in courses). The number of students in each group are shown in Table 1.

An introductory lecture was provided to motivate the study, describe the content of the study, introduce inspections in general and the specific techniques used in particular. The

students were offered to participate in the study. The students voted whether to participate or not to participate in the study. Before the voting, all students agreed that they would follow the majority vote (i.e., they were given the opportunity of not participating). The majority of students voted in favor of the experiment. To further increase the benefit for the students in the context of the project management course, they were given an assignment to reflect on the strengths and weaknesses of experiments as a basis for management decisions in the context of software engineering.

Table 1: Experience of Students

	Number of Students
Highly Experienced	8
Experienced	6
Less Experienced	9

4.2.2 Experimental Package

The experimental package used was developed at Lund University by several researchers working at the Department of Communication Systems and comprised a taxi management system. The goal of the system is to manage customer orders submitted either to the taxi central or to the taxi driver. Moreover, the system is able to automatically dispatch orders to available taxis [2].

The following documents are provided in the experiment package. More details can be found in [2].

- *Textual requirements:* The requirements are formulated in natural language (English). None of the participants had English as their native language, but the course course in which the experiment was conducted was held in English. In the context of this experiment, the requirements are assumed to be correct, i.e., if there are inconsistencies between the design document under inspection and the textual requirements, then the reviewer shall assume that the design is incorrect.

- *Use case document:* The use case document originally consisted of 24 use cases. However, since time was limited to two hours for this experiment, the last five use cases (i.e., use cases with the lowest priority) were removed which increases the likelihood that the reviewer is able to utilize all of them within the given time. For TC-UBR, time budgets need to be assigned to every use case. Time is assigned based on the prioritization of the use cases. More time is assigned to use cases with higher priority. The use cases are formulated in task notation [13], including purpose of the use case, tasks and variants of the use case.

- *Design document:* The size of the design document is 2300 words on 9 pages. In the beginning of the document, the reviewer gets an overview describing the components of the system and how they communicate with each other through signals. The signals and their parameters are listed as well. After that, two scenarios (one special case and one normal case) are described in the notation of Message Sequence Charts (MSC) [31]. Within the document a total of 38 faults exist. The faults are divided into three classes, namely A (13 faults), B (14 faults) and C (11 faults). The characteristics of the fault classes are defined as follows: A-faults are crucial from the users' perspective, they concern functionality that is often used and important to the user. B-faults are important, that means they are either often used or of high importance. C-faults are related to functionality that is not often used and not considered as important. The faults are the same as in previous experiments on usage based reading, the faults being real faults made during the design of the system.

- *Inspection record form:* In this form, the reviewer documents the individual inspection process. The reviewer logs time spent for different activities, faults found and use case used to find the fault. Furthermore, the location of the fault is logged, the time when the fault was found and a short description of the fault so that the researcher can judge whether it is a false positive or an existing fault.

- *Guidelines:* The guidelines are a concise description of the reading process. They state the maximum time assigned to overview reading. Moreover they provide a short description of the technique. The guidelines also provide instructions of what to do when the inspection is finished. For TC-UBR, the introductions have to be slightly changed, so that the maximum time budgets for each use case are taken into consideration.

- *Feedback Questionnaire:* In the questionnaire the reviewer is asked to provide feedback about the technique. Moreover, the reviewer shall rate process conformance and difficulty to follow the process. This helps to determine whether the reviewer executed the technique as intended. The reviewer is also encouraged to make improvement suggestions and to comment on the quality of the documents in order to further improve the experimental package.

4.2.3 Variables

The independent variable (factor) controlled by the researcher is the reading technique. The values assigned to this variable (treatments) are TC-UBR and RB-UBR.

Dependent variables are effected by the treatment and they have to be measured to say something about the cause effect relationship between treatment and dependent variable [30]. The following dependent variables are determined by the indirect and direct measurements [24]:

- Time spent on overview reading in minutes
- Time spent on inspection in minutes
- Time when each fault was found by the reviewer in minutes from the beginning of the inspection

From these direct measurements, the indirect measurements for effectiveness and efficiency are calculated according to the following formulas [24]:

$$Effectiveness = \frac{Number\ of\ Faults\ Found}{Total\ Number\ of\ Faults} \quad (1)$$

$$Efficiency = 60 * \frac{Number\ of\ Faults\ Found}{Reading\ Time + Inspection\ Time} \quad (2)$$

Moreover, the variable experience might effect the dependent variables. If the variable experience is not taken into consideration, one group might consist of much more experienced reviewers. Thus, the result of the experiment is not primarily affected by the factor reading technique, but also by the experience of the reviewer. This was handled through the questionnaire handed out before running the experiment.

4.2.4 Hypotheses

To answer the research questions, a set of hypotheses is evaluated. The research questions and the related hypotheses are presented in the following:

Q1: Is TC-UBR more effective than RB-UBR? The null hypotheses (H_{0_S}) for effectiveness states that the share of faults (S) is the same for both techniques while the alternative hypotheses states that the share of faults is higher for TC-UBR (H_{a_S}). The hypotheses are checked for *all* faults, $A - faults$ and $A\&B - faults$.

$$H_{0_{S_{all}}} := S_{all}(TC - UBR) = S_{all}(RB - UBR)$$
$$H_{a_{S_{all}}} := S_{all}(TC - UBR) > S_{all}(RB - UBR)$$
$$H_{0_{S_A}} := S_A(TC - UBR) = S_A(RB - UBR)$$
$$H_{a_{S_A}} := S_A(TC - UBR) > S_A(RB - UBR)$$
$$H_{0_{S_{A\&B}}} := S_{A\&B}(TC - UBR) = S_{A\&B}(RB - UBR)$$
$$H_{a_{S_{A\&B}}} := S_{A\&B}(TC - UBR) > S_{A\&B}(RB - UBR)$$

Q2: Is TC-UBR more efficient than RB-UBR? The null hypotheses (H_{0_E}) assumes equal efficiency (E) for both techniques while the alternative hypotheses (H_{a_E}) are formulated in favor for TC-UBR. The efficiency is evaluated for *all* faults, $A - faults$ and $A\&B - faults$.

$$H_{0_{E_{all}}} := E_{all}(TC - UBR) = E_{all}(RB - UBR)$$
$$H_{a_{E_{all}}} := E_{all}(TC - UBR) > E_{all}(RB - UBR)$$
$$H_{0_{E_A}} := E_A(TC - UBR) = E_A(RB - UBR)$$
$$H_{a_{E_A}} := E_A(TC - UBR) > E_A(RB - UBR)$$
$$H_{0_{E_{A\&B}}} := E_{A\&B}(TC - UBR) = E_{A\&B}(RB - UBR)$$
$$H_{a_{E_{A\&B}}} := E_{A\&B}(TC - UBR) > E_{A\&B}(RB - UBR)$$

Q3: Are different faults detected for TC-UBR and RB-UBR? The null hypotheses for fault content assumes that the same faults are found while the alternative hypotheses assumes that both techniques discover different faults.

$$H_{0_{Fault}} := Fault(TC - UBR) = Fault(RB - UBR)$$
$$H_{a_{Fault}} := Fault(TC - UBR) \neq Fault(RB - UBR)$$

4.2.5 Design Principle

In order to have an equal level of experience in the treatment and control group, three groups of students are formed. The first group has high experience in programming, inspections and use cases which are considered to be the most important areas of expertise to execute the experiment successfully (as determined by the experience questionnaire introduced before). The second group has a moderate level of experience and the third group has quite limited experience (see Table 1). From each group, the students are assigned randomly to the treatments, reducing the risk that experience strongly effects the dependent variables.

The standard design types for an experiment can be selected based on the number of factors and treatments [30]. This study has one factor (reading technique) and two treatments (TC-UBR and RB-UBR). Based on this, the Mann-Whitney test is selected to compare the median values of for effectiveness and efficiency. Furthermore, the test is selected due to its robustness as it does not make assumptions on the distribution of data. Whether the fault content is the same or not for the groups is determined through the chi-square test. The significance level is set to $\alpha = 0.05$ for all tests.

4.3 Operation of the Experiment

The operation is divided into different session. An introduction is provided in session 1 (as described above). Moreover, anonymity of the subjects is guaranteed and the students are informed that their performance in the experiment is not part of the grading in the course. After that, the students are provided with information of the documents they are going to work with. This includes an explanation of the taxi management system. Also the notations used in the document are explained so that the students are able to read and interpret the information provided in the documents correctly.

In session 2, the students of the treatment group (TC-UBR) and control group (RB-UBR) are distributed to two different classrooms. Thereafter, each group gets an introduction to their treatment. In the presentations the students are informed that they are not allowed to discuss their technique with members of the other group until after the experiment. The students receive the following instructions:

- Read the documents (overview reading) in order to understand them for no longer than 45 minutes.
- Inspect the document guided by the use cases using your designated reading technique.
- Use the requirements document as reference. The requirements document is assumed to be correct. That means, if you identify inconsistencies between requirements and the design document under inspection, you have to document this as a fault in the design.
- Record the time spent and faults found in the inspection record form.
- When finished, log the clock time and last use case utilized. Fill in the feedback questionnaire and hand in everything.

After this introduction the students apply their respective technique on a small example learning how to use the technique and the inspection record form. Moreover, misunderstandings between the presenter and student can be discovered during the lunch break after session 2 as the presenter reviews the forms. The small example uses the same structure and notation as the ones intended for the inspection experiment.

In session 3, both groups conduct the inspection on the taxi management system in the rooms where the instruction to the techniques was provided. Before starting the inspection, the students are informed about mistaken issues from the instructions which were identified during the lunch

break. After that the students take part in the inspection for two hours. When they are finished, they fill in the feedback questionnaire and hand in everything. In Table 2 the whole experiment operation including times and themes for each session is summarized.

Table 2: Overview of Experiment Operation

	Time (min)	Group 1	Group 2
10:00-10:45 (Session 1)	45	System Introduction	
11:00-12:00 (Session 2)	60	Intro UBR-TC	Intro UBR-RB
13:00-16:45 (Session 3)	120	Individual Inspection	
17:00-17:15 (Session 3)	15	Feedback Questionnaire	

4.4 Threats to Validity

Conclusion validity is concerned *"with issues that effect the ability to draw the correct conclusions about relations between the treatment and the outcome of an experiment"* [30]. One threat is that the treatment implementation differs between the subjects. For example, the difference in treatments can be caused by discussions between subjects, i.e., the discussion might change the view of the subject on the treatment and thus an unintended difference occurs. This threat is minimized by separating the students into two groups and forbidding them to talk with each other. Another threat is the heterogeneous characteristic of the subjects who are all graduate students from different countries, i.e., they have quite different experience and backgrounds. As Wohlin et al. [30] point out undergraduate students are more likely to be heterogeneous, but it is harder to generalize the results from such a sample to a population. Therefore, the threat of heterogeneity of subjects is accepted in this study in favor of external validity, i.e., the ability to generalize the results of the study to industry practice. In summary, the threats to conclusion validity can be considered under control.

Threats to internal validity are concerned with *"influences affecting the independent variable with respect to causality without the researcher's knowledge"* [30]. One possible threat to internal validity is a lack of quality of the instruments used, for example quality of collection forms, documents to be inspected and so forth. Since the documents were used in several former inspections experiments they are thoroughly tested and thus this threat is minimized. A further threat is that the experience of the subjects effects the result of the experiment besides the treatments. This threat is controlled by the experience questionnaire and by putting the students into three groups of highly experienced, median experienced and less experienced students within the relevant fields of expertise and assigning them randomly to the treatments. Hence, it can be assumed that possible threats to internal validity are also under control.

Construct validity is the ability to *"generalize the result of the experiment to the concept or theory behind the experiment"* [30]. Biased judgment might threaten the objectivity of the prioritization of the use cases. To minimize this threat, the prioritization was done by different researchers at Lund University with sound domain knowledge in the taxi management domain. Thus, the prioritization result becomes more objective. In addition, the expectations of the experimenter could be a threat to conclusion validity because the researcher might tweak the results in order to fulfill some specific expectations of the experiment, consciously or unconsciously. However, this threat can also be considered as quite low because the main experimenter did not invent the technique and thus has a more objective view on the experiment. A further validity threat is hypothesis guessing. This threat is (as former threats concerning conclusion and internal validity) addressed by separating the subjects according to their treatments and by forbidding them to talk about the experiment. Summarized one can say that the threats to construct validity are also under control.

External validity is the ability to *"generalize the results of an experiment to industrial practice"* [30]. It is always a threat to external validity to have students as subjects. However, when using students it is desirable to consider Masters students for the experiment because they have more experience and thus are more similar to people working in industry than undergraduate students. In this particular case, the students all have practical experience from projects because this is a prerequisite to get admitted to the Master Program in Software Engineering at Blekinge Institute of Technology. Thus, the students participating in this study can be considered as similar to people with two years of industrial experience. Another threat is the use of toy systems, as such material is not representative for industrial practice. The system used in this study is based on a real world problem, but it is scaled down to a manageable size. However, the study is comparative and the researchers have been unable to identify any reason why one of the methods would work better than the other when scaling to a larger system. In other words, the relative difference between the methods ought to be the same for a larger system. Having said this, a potential threat is the use of the same experimental package in all experiments. The results may be a result of the package. This is best addressed with a replication not using the same experimental package. The threats to external validity can be considered as the most crucial ones in this study.

5. EXPERIMENT RESULTS

Four subjects had to be excluded before the data is evaluated because two students withdrew from the experiment and two students did not follow the reading process (discussed in the context of process conformance). Thus, a total of 19 subjects remained and are included in the statistical analysis.

5.1 Time Data

In session 2 the students were instructed to log the time they needed for overview reading and inspection. Overview reading means that they read through the requirements, design and use cases in order to familiarize themselves with the system before starting the real inspection. Table 3 summarizes the average times for overview reading and individual inspection as well as the standard deviations. The group that applied RB-UBR used a little more time for overview reading (1.2 minutes) and inspection (3.9 minutes). For the inspection as a whole the group using RB-UBR took on average 5 minutes longer time. The high value of the standard deviation in regard to individual inspection times can be explained by the fact that some subjects inspected the doc-

uments for quite a short time. In the case of RB-UBR two subjects only inspected the design for the duration of 78 and 90 minutes respectively. In the case of TC-UBR three subjects utilized even less time, i.e. 74, 82 and 82 minutes respectively.

Table 3: Overview Reading and Inspection Times

	Mean (in Minutes)		Standard Deviation	
	TC-UBR	RB-UBR	TC-UBR	RB-UBR
Overview	35.4	36.6	7.83	5.43
Inspection	104.8	108.7	15.8	15.7
Total	140.2	145.2	5.5	18.3

5.2 Individual Reviewer Performance

The subjects logged the time when they discovered a fault within the design document. Based on this data, the cumulative number of faults is plotted in a cumulative histogram for an average reviewer in the TC-UBR and the RB-UBR group for the most critical faults, i.e., A-faults and A&B-faults as well as for all faults. For each point in time on the time line the total number of defects discovered so far is standardized by the number of reviewers. In Figure 2 the individual performance of an average reviewer in both groups is shown for all faults (A, B, C). It becomes apparent that the number of faults found by an average reviewer in both groups is similar until minute 80. After that minute, the slope of the TC-UBR curve increases and thus the TC-UBR group found more faults than the RB-UBR group in the end of the inspection (after 160 minutes including overview reading). In particular, the TC-UBR group found 17 % more faults than the RB-UBR group. Consequently one can claim that the average reviewer applying time controlled reading performed moderately better.

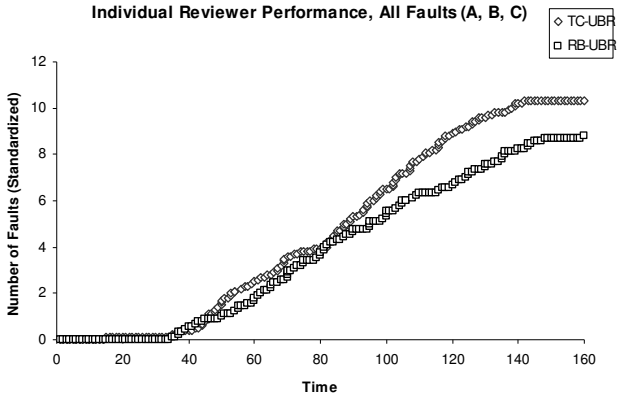

Figure 2: Cumulative Number of all Faults found by an Average Reviewer

Separate analysis have been made for only A-faults and A&B-faults. The outcome is similar. For A faults, the TC-UBR group performs slightly better. For example, the average reviewer applying time controlled reading found 2 % more A-faults and thus performed insignificantly better. For A&B-faults, an average reviewer in the TC-UBR group found 13 % more A&B-faults. In summary, the individual TC-UBR reviewer performed better in finding all faults, A- faults and A&B-faults. However, the difference is quite small between TC-UBR and RB-UBR concerning all faults and A&B-faults.

5.3 Fault Content

The design document includes a total of 13 A-faults, 14 B-faults and 11 C-faults. This section shows the share of reviewers that discovered specific A-faults and A&B-faults by applying the different treatments. The results are illustrated in form of bar plots where the x-coordinate shows the identifier of the faults seeded in the design document and the y-coordinate represents the share of reviewers finding the faults. One could also use the absolute number of faults found, but in this case it is reasonable to use the share of reviewers because the number of reviewers differs between the groups. Even though the average reviewer in both groups found a similar number of class A-faults, the histogram in Figure 3 indicates that the groups found moderately different A-faults. It attracts attention that the group that applied time controlled reading was able to discover all class A-faults. In contrast to this, the group applying rank based reading missed two A-faults (number 4 and 8). However, fault 4 was only discovered by one reviewer and fault 8 by two reviewers. No common pattern can be found and it is hard to draw any conclusions based on the usage of the different inspection methods based on the histogram.

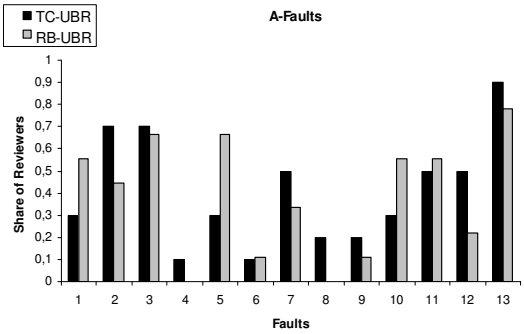

Figure 3: Share of Reviewers that Found each A-Fault

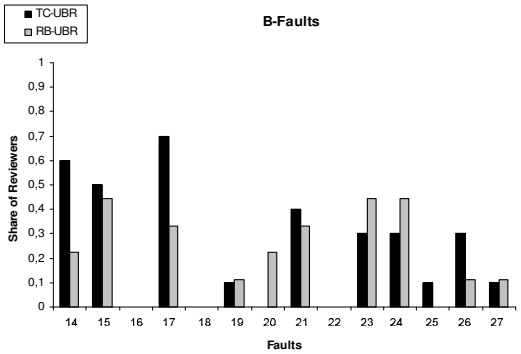

Figure 4: Share of Reviewers that Found each B-Fault

When studying B-faults, a similar outcome is obtained as for A-faults. In other words, no real pattern is observed. To evaluate the differences for both A-faults and B-faults

statistically, the chi-square test is applied on the frequencies of how often the 38 faults included in the design were found by the two groups of subjects applying TC-UBR and RB-UBR. The p-value of the chi-square test is $p = 0.818$ which is larger than $\alpha = 0.05$. Thus, the following hypothesis cannot be statistically rejected in this study: $H_{0_{Fault}}$

5.4 Effectiveness and Efficiency

Figures 5 and 6 show box plots illustrating effectiveness and efficiency for all faults, A-faults and A&B-faults. The plots illustrate how the data points are distributed. For example, in Figure 6 the top line of the box plot represents the maximum value in the data set (e.g., the data set for TC-UBR and all faults has a maximum value of 6.32). In contrast to this, the bottom line shows the lowest value within the data set, here 2.79. The line in the middle of the box plots breaks the whole data set into two equally large halves, i.e., the line represents the median value. The boxes around the median value illustrate the upper and lower quartile.

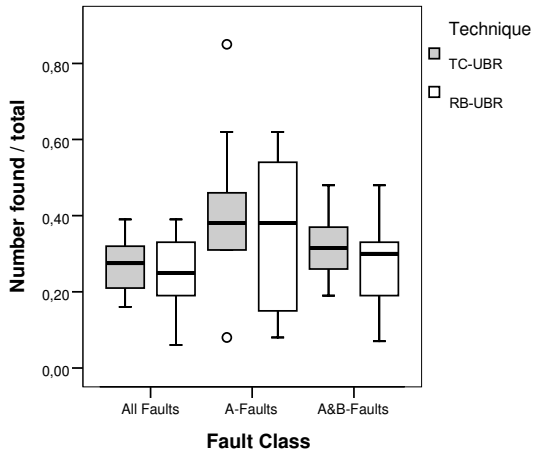

Figure 5: Effectiveness for all Faults, A-Faults and A&B-Faults

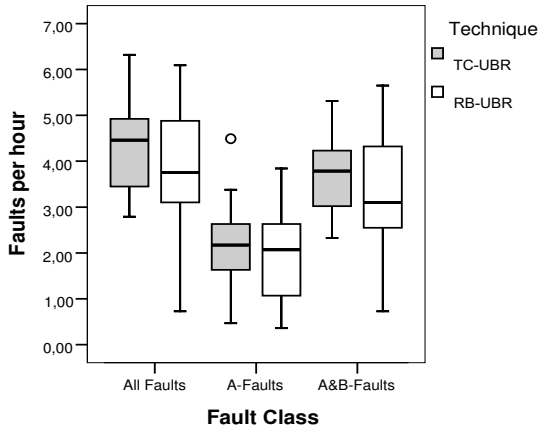

Figure 6: Efficiency for all Faults, A-Faults and A&B-Faults

Figure 5 shows that the box plots lie on the same level regarding the share of faults found for the different types of faults. The median values are nearly the same for both techniques. Hence, it seems that both techniques are similar with respect to effectiveness. Furthermore, two outliers can be found (A-faults). Outliers can be excluded when they are due to a reason that does not occur again. On the other hand, as suggested in [30], the outlier should not be excluded if it exists due to rare events. In this case, the reason for the outliers were the capabilities of the reviewers. This has also been observed in previous experiments [20][24][22]. Hence, the statistical tests include the outlier. The p-values calculated with the Mann-Whitney test are $p = 0.795$ for all faults, $p = 0.951$ for A-faults and $p = 0.532$ for A&B-faults. Consequently, the following stated hypotheses cannot be rejected: $H_{0_{S_{all}}}$, $H_{0_{S_A}}$, and $H_{0_{S_{A\&B}}}$.

Figure 6 shows that the box plots lie on the same level of faults found per hour regarding all faults, A-faults and A&B-faults. Only the median values of all faults and A&B-faults are moderately higher. Therefore, the figure indicates that time controlled and rank based reading have about the same efficiency. Moreover, one outlier is identified which was due to the good capabilities of the reviewer and hence is included in the analysis. For efficiency, the Mann-Whitney test gives the following p-values: For all faults the p-value is $p = 0.367$, for A-faults $p = 0.765$ and for A&B-faults $p = 0.535$. Since all these p-values are larger than p=0.05 the following hypotheses cannot be rejected: $H_{0_{E_{all}}}$, $H_{0_{E_A}}$, and $H_{0_{E_{A\&B}}}$.

5.5 Perceived Difficulties and Process Conformance

In Figure 7, the results from the question of how difficult the students perceived the different techniques are plotted. It should be observed that this is the perceived difficulty of the techniques, i.e. it shows the subjective impression of the students. It is possible to see that both groups perceived the difficulty of TC-UBR and RB-UBR in a similar way. The majority of students (about 50 %) perceived the techniques as neither easy nor difficult. One quarter perceived the techniques as easy and the last quarter found that they are somewhat difficult. Since both techniques are similar and the fact that the perception of difficulty is influenced by prior knowledge and experience it can be assumed that the groups are balanced quite well in respect of their experience.

Even though the techniques are similar, the subjects applying TC-UBR perceived their process conformance as lower than the subjects using RB-UBR did (see Figure 8). This becomes apparent because many more of the subjects using RB-UBR think that they followed the process all the time (40 %) while only 10 % of the subjects using TC-UBR felt the same way. Moreover, only subjects in the TC-UBR group claim that they followed the reading process sometimes (20 %) or seldom (10 %). In fact, two of the subjects using TC-UBR admit that they did not follow the process properly, and hence they had to be removed from the statistical evaluation due to lack of process conformance. One possible reason is a loss of interest during the course of the inspection. In order to verify process conformance of the remaining subjects the review forms were checked for completeness. Furthermore, in the TC-UBR group the time frame in which each use case is supposed to be inspected was checked to verifiy that the students stayed within the time budgets assigned to the use cases. Overall, the inspection materials handed in indicate that the process was followed quite well by all remaining subjects.

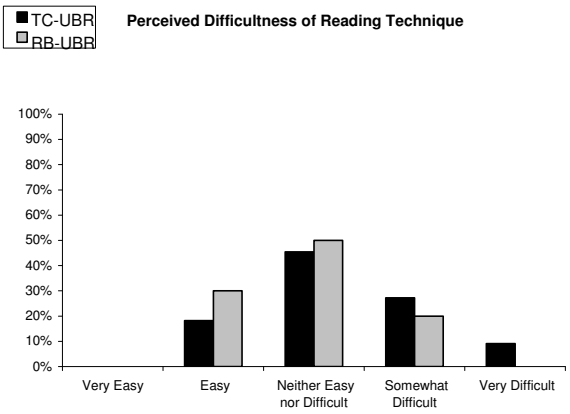

Figure 7: Perceived Difficulty

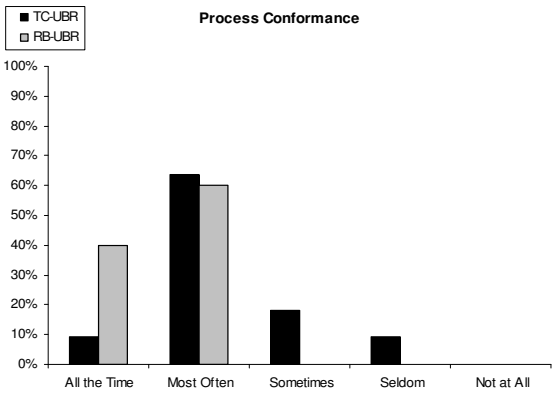

Figure 8: Perceived Process Conformance

6. DISCUSSION

Based on our hypothesis tests, we were not able to reject the null hypotheses posed for this experiment. In the following table, an overview of the p-values is provided. It becomes apparent that the p-values are all quite high. This is an indication for high similarities between both techniques in terms of effectiveness, efficiency and fault content. Thus, no claims can be made whether one technique performs better than the other.

Table 4: p-values of Hypotheses Tests

	All (A, B, C)	A	A&B
Efficiency	0.367	0.765	0.535
Effectiveness	0.795	0.951	0.532
Fault Content	0.818		

We suspect three main reasons why TC-UBR did not lead to significant improvements.

- The first reason being that the time budgets do not take the experience of the reviewer into consideration as they were the same for all reviewers. An experienced reviewer might be able to thoroughly inspect a highly prioritized use case in shorter time, while a less experienced reviewer feels pressured. This is also indicated in the data, Figures 5 and 6 show that the variance in the data set seems to be higher for TC-UBR than for RB-UBR.

- Secondly, the complexity of the use cases could also negatively influenced the performance. In fact, researchers established a relationship between use cases and system complexity when developing use case points [1][10]. One factor that is used to predict the complexity of the system with use case points is the number of transaction steps within a use case. In this study, the use cases differ in their number of transaction steps, ranging from three to twelve transaction steps. The more complex the use case, the harder it is for the reviewer to inspect it within a given time frame.

- A third possible reason is that the students feel stressed by the time constraints when compared to inspectors that have years of industrial experience.

Overall, the finding in this study does not support moving from usage-based reading to the time-controlled version. Thus, it seems sufficient to just prioritize the use cases.

7. CONCLUSION AND FURTHER WORK

In this experiment, we applied the principles of statistical usage testing to software inspections. The usage profiles determine how intensively certain parts of the system should be tested from the users' perspective. In order to use this in inspections, the reviewers use prioritized use cases as a driver to inspect a software artifact. How intensively a use case should be inspected is reflected in the time budged for the use case (referred to as time controlled usage-based reading). In order to evaluate the performance of this technique, we compared it with usage-based reading without time budgets (i.e., the use cases are only inspected in prioritized order). That is, it is not pre-defined how intensively the use case should be inspected. The comparison was made using a controlled experiment with 23 Software Engineering M.Sc. students. The outcome of the experiment is that no significant improvements are achieved when assigning time budgets based on usage profiles. The two main reasons for the result are that 1) the experience of reviewers is not reflected when only considering usage profiles and 2) the complexity of use cases is not taken into account. The finding in this study does not support moving from usage-based reading to the time controlled version. Thus, it seems sufficient to just prioritize the use cases.

In future work the time budgets should be assigned by combining factors like usage profile and use case complexity. For example, by deciding that a specific use case complexity requires a minimum time budget while the maximum budget is decided based on the profile. Thereafter, complexity and priorities have to be balanced against each other.

8. ACKNOWLEDGMENTS

We would like to thank all students who participated in the experiment, and Stephan Petersen for support during the operation of the experiment. This work was partly funded by The Knowledge Foundation in Sweden under a research grant for the project "Blekinge - Engineering Software Qualities (BESQ)" (http://www.bth.se/besq).

9. REFERENCES

[1] B. Anda, H. C. Benestad, and S. E. Hove. A multiple-case study of software effort estimation based on use case points. In *Proc. of the fourth International Symposium on Empirical Software Engineering (ISESE 2005)*, pages 407–416, 2005.

[2] C. Andersson, T. Thelin, P. Runeson, and N. Dzamashvili. An experimental evaluation of inspection and testing for detection of design faults. In *Proc. of the 2nd International Symposium on Empirical Software Engineering (ISESE 2003)*, pages 174–184, 2003.

[3] A. Aurum, H. Petersson, and C. Wohlin. State-of-the-art: software inspections after 25 years. *Softw. Test., Verif. Reliab.*, 12(3):133–154, 2002.

[4] V. R. Basili, S. Green, O. Laitenberger, F. Shull, S. S., and M. V. Zelkowitz. The empirical investigation of perspective-based reading. Technical report, University of Maryland at College Park, College Park, MD, USA, 1995.

[5] D. B. Bisant and J. R. Lyle. A two-person inspection method to improve programming productivity. *IEEE Transactions on Software Engineering*, 15(10):1294–1304, 1989.

[6] S. G. Eick, C. R. Loader, M. D. Long, L. G. Votta, and S. A. V. Wiel. Estimating software fault content before coding. In *Proc. of the 14th International Conference on Software Engineering (ICSE 1992)*, pages 59–65, 1992.

[7] M. E. Fagan. Design and code inspections to reduce errors in program development. *IBM Systems Journal*, 15(3):182–211, 1976.

[8] P. M. Johnson. An instrumented approach to improving software quality through formal technical review. In *Proc. of the 16th International Conference on Software Engineering (ICSE 1994)*, pages 113–122, 1994.

[9] J. Karlsson and K. Ryan. A cost-value approach for prioritizing requirements. *IEEE Software*, 14(5):67–74, 1997.

[10] S. Kusumoto, F. Matsukawa, K. Inoue, S. Hanabusa, and Y. Maegawa. Estimating effort by use case points: Method, tool and case study. In *Proc. of the 10th IEEE International Software Metrics Symposium (METRICS 2004)*, pages 292–299, 2004.

[11] O. Laitenberger, C. Atkinson, M. Schlich, and K. El-Emam. An experimental comparison of reading techniques for defect detection in uml design documents. *Journal of Systems and Software*, 53(2):183–204, 2000.

[12] O. Laitenberger, K. El-Emam, and T. G. Harbich. An internally replicated quasi-experimental comparison of checklist and perspective-based reading of code documents. *IEEE Transactions on Software Engineering*, 27(5):387–421, 2001.

[13] S. Lauesen. *Software Requirements: Styles and Techniques*. Addison-Wesley Professional, 2002.

[14] D. Leffingwell and D. Widrig. *Managing Software Requirements: A Use Case Approach (2nd Edition) (The Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2003.

[15] R. C. Linger. Cleanroom process model. *IEEE Software*, 11(2):50–58, 1994.

[16] J. Martin and W. T. Tsai. N-fold inspection: A requirements analysis technique. *Communications of the ACM*, 33(2):225–232, 1990.

[17] V. Mashayekhi, J. M. Drake, W.-T. Tsai, and J. Riedl. Distributed, collaborative software inspection. *IEEE Software*, 10(5):66–75, 1993.

[18] J. D. Musa. *Software Reliability Engineering: More Reliable Software Faster and Cheaper 2nd Edition*. AuthorHouse, 2004.

[19] I. Sommerville. *Software Engineering: (Update) (8th Edition) (International Computer Science Series)*. Addison Wesley, 2006.

[20] T. Thelin, C. Andersson, P. Runeson, and N. Dzamashvili-Fogelström. A replicated experiment of usage-based and checklist-based reading. In *Proc. of the 10th IEEE International Software Metrics Symposium (METRICS 2004)*, pages 246–256, 2004.

[21] T. Thelin, P. Runeson, and B. Regnell. Usage-based readingan experiment to guide reviewers with use cases. *Information & Software Technology*, 43(15):925–938, 2001.

[22] T. Thelin, P. Runeson, and C. Wohlin. An experimental comparison of usage-based and checklist-based reading. *IEEE Transactions on Software Engineering*, 29(8):687–704, 2003.

[23] T. Thelin, P. Runeson, and C. Wohlin. Prioritized use cases as a vehicle for software inspections. *IEEE Software*, 20(4):30–33, 2003.

[24] T. Thelin, P. Runeson, C. Wohlin, T. Olsson, and C. Andersson. How much information is needed for usage-based reading? - a series of experiments. In *Proc. of the 1st International Symposium on Empirical Software Engineering (ISESE 2002)*, pages 127–138, 2002.

[25] L. G. Votta. Does every inspection need a meeting? In *Proc. of the 1st ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE 1993)*, pages 107–114, 1993.

[26] G. H. Walton, J. H. Poore, and C. J. Trammell. Statistical testing of software based on a usage model. *Software: Practice and Experience*, 25(1):97–108, 1995.

[27] D. Winkler, M. Halling, and S. Biffl. Investigating the effect of expert ranking of use cases for design inspection. In *Proc. of the 30th EUROMICRO Conference 2004*, pages 362–371, 2004.

[28] C. Wohlin, B. Regnell, A. Wesslén, and H. Cosmo. User-centered software engineering: A comprehensive view of software development. In *Proc. of the Nordic Seminar on Dependable Computing Systems*, pages 229–240, 1994.

[29] C. Wohlin and P. Runeson. Defect content estimations from review data. In *Proc. of the 20th International Conference on Software Engineering (ICSE 1998)*, pages 400–409, 1998.

[30] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction (International Series in Software Engineering)*. Springer, 2000.

[31] I.-T. Z.120. Message sequence charts, msc, itu-t recommendation z.120, 1996.