A Controlled Experiment for Evaluating a Metric–Based ReadingTechnique for Requirements Inspection

B. Bernárdez, M. Genero, A. Durán, M. Toro

Abstract

Natural language requirements documents are often verified by means of some reading technique. Some recommendations for defining a good reading technique point out that a concrete technique must not only be suitable for specific classes of defects, but also for a concrete notation in which requirements are written. Following this suggestion, we have proposed a metric–based reading (MBR) technique used for requirements inspections, whose main goal is to identify specific types of defects in use cases. The systematic approach of MBR is basically based on a set of rules as "if the metric value is too low (or high) the presence of defects of type $defType_1$,...$defType_n$ must be checked". We hypothesised that if the reviewers know these rules, the inspection process is more effective and efficient, which means that the defects detection rate is higher and the number of defects identified per unit of time increases. But this hypotheses lacks validity if it is not empirically validated. For that reason the main goal of this paper is to describe a controlled experiment we carried out to ascertain if the usage of MBR really helps in the detection of defects in comparison with a simple Checklist technique. The experiment result revealed that MBR reviewers were more effective at detecting defects than Checklist reviewers, but they were not more efficient, because MBR reviewers took longer than Checklist reviewers on average.

***Keywords**: requirements verification, reading techniques, use cases, metrics, empirical validation.*

*This work has been partially funded by the WebMade project (TIC 2003-02737-C02-01), the CALIPO project (TIC2003-07804-C05-03) and the MESSENGER project (PCC-03-003-1).

1. Introduction

The convenience of early detection of problems in requirements is widely recognized to improve the quality in the software development process [13]. One of the first studies in this sense was that presented in [6], in which Boehm concludes that the later a defect is identified in the life–cycle the more expensive it will be to repair it.

Several *reading techniques* have been proposed for requirements quality evaluation: the first proposal was adapting the code inspection technique by Fagan [10] to requirements documents. Then, as summarised in [16], several experts in requirements quality argued that a *checklist* is necessary to aid and regularise the requirements inspection process. Later, Parnas and Weiss [17] suggested that a checklist is not sufficient and commented that if the reviewers are focused on specific aspects of quality in requirements more defects will be identified. In order to achieve this goal, two main techniques for requirements inspection have been proposed, *Scenarios* (so called Defect–based Reading) [18] and Perspective–based Reading (PBR)[2]. Scenarios state that each reviewer searches for concrete types of defects in the requirements documents (classified by their nature, for example in [18] three scenarios are defined: Data Type Consistency, Incorrect Functionality and Ambiguities or Missing Functionality). PBR proposes that each reviewer verify the requirements document from a particular point of view that could be tester, developer or user of the system. As commented in [2], focusing on specific types of defects should lead to a more in–depth analysis of potential errors in the requirements document, although it provides a particular coverage of the document.

On the other hand, in [2] it is suggested that a require-

ments inspection technique should be associated with the notation in which the document is written. Since the main purpose of use cases technique is to specify part of the functional requirements of the system to be built [4], a use case defect detection technique is defined and empirically improved in our previous works [5, 9] (see section 2). This technique called *MBR* (Metric–Based Reading) is based on a set of heuristics whose usage during the requirements inspection leads this process in a systematic approach that should improve the effectiveness and efficiency of requirements inspection.

The heuristics rely on the idea that certain structural properties of use cases, which are easy to measure, could be early indicators of specific defect types in use cases, such as incompleteness, ambiguity, difficulty of understanding, lack of conciseness, triviality, etc. (see Appendix A).

The definition of software product metrics as fault–proneness indicators is widely acknowledged within the software community, for example, in [7] a set of metrics to evaluate the quality of the high–level designs with respect to its fault–proneness are defined and validated. Something similar is the idea of our heuristics, to determine which metrics values of use cases could indicate the presence of defective use cases.

But as Zelkowitz et al. [23] pointed out the proposal of a new technology lacks credibility if there is no empirical evidence of its usefulness. For that reason, we carried out a controlled experiment in order to corroborate that *MBR* really is more effective and efficient than the *Checklist* technique (see appendix A), which means that the application of *MBR* increases the rate of defects found by the reviewers and provides a greater performance of inspection time. The main objective of this paper is to describe each of the steps we followed to carry out that experiment.

The rest of the paper is organised as follows: *MBR*, the technique we propose for requirements inspection, is presented in section 2. The empirical validation of *MBR* is presented in section 3. Section 4 describes the empirical work existing in the literature related to requirements inspections techniques and finally the last section presents some concluding remarks and outlines the future work.

2. *MBR* a Metric–Based Reading Technique

MBR is a defect detection technique for use case inspection that is based on the set of heuristics that will be briefly summarised in this section. The detailed aspects of the development of this technique are collected in [8, 9]. The intuition behind these heuristics is that there are several use case metrics (see table 1) that can be used as defect–proneness indicators i.e. there exists an underlying cause–effect relationship between the metric values and the presence of certain defect types in use cases. Every heuristic defines a usual range (thresholds values, represented as $[inf, sup]$) for a use case metric ($m(uc)$), outside of which, the probability of the use case being defective increases.

Metric	Description
NOS	Number of steps of the use case (NOS=NOAS+NOSS+NOUS)
NOAS	Number of actor action steps of the use case
NOSS	Number of system action steps of the use case
NOUS	Number of use case action steps of the use case (inclusion or extension)
NOCS	Number of conditional steps of the use case
NOE	Number of exceptions of the use case
NOAS/NOS	Rate of actor action steps of the use case
NOSS/NOS	Rate of system action steps of the use case
NOUS/NOS	Rate of use case action steps of the use case
CC	Cyclomatic complexity of use case (NOCS+NOE+1)

Table 1. Use cases metrics

The heuristics and their corresponding metrics usual ranges were proposed after analyzing 414 use cases which have been developed by students, as described in [8]. A first empirical assessment for corroborating the heuristics was presented in [5]. Next, for each metric shown in table 1, its mean, standard deviation and usual range are indicated. Moreover, the rationale on which the definition of each heuristic is based on is also provided.

- **Metric NOS**: NOS $\in$ [0,∞] (μ = 5.70, σ = 2.64)
  Usual range: [4,9]
  Rationale: A use case with just a few steps is likely to be incomplete. Too many steps make the use case too complex to be understood.

- **Metric NOAS/NOS**: NOAS/NOS $\in$ [0,1] (μ = 34.52%, σ = 17.74%)
  Usual range: [30%,60%]
  Rationale: A use case describes system–actors interactions, so NOAS/NOS and NOSS/NOS should be around 50%.

- **Metric NOSS/NOS**: NOSS/NOS $\in$ [0,1] (μ = 59.71%, σ = 18.80%)
  Usual range: [40%,80%]
  Rationale: Same as NOAS/NOS metric.

- **Metric NOUS/NOS**: NOUS/NOS $\in$ [0,1] (μ = 5.77%, σ = 10.42%)
  Usual range: [0%,35%]
  Rationale: An abusive use of use case relationships makes use cases difficult to understand.

- **Metric CC**: CC $\in$ [1,∞] (μ = 2.52%, σ = 1.22%)
  Usual range: [1,5]

Rationale: A high value of CC complies too many conditional steps and exceptions, probably making the use case too complex.

These metrics have been defined for the use case model of REM [9], a requirements management tool that can automatically calculate its values. In this model, a use case is basically seen as a sequence of steps. Actions of these steps can be one of three different classes: actor–action if the action is performed by an actor; system–action if the action is performed by the system, or a use case–action, if the action consists of performing another use case (*i.e.* an *inclusion* if it is not a conditional step, an *extension* otherwise).

After carrying out a second empirical study to confirm the heuristics [3], we have identified the main types of defects in use cases in which $m(uc) \notin [inf, sup]$. As it can be seen in table 2, these causes are commonly different if the value of the metric is lower than inf or higher than sup.

On the other hand, as noted above, since a use case represents an actor–system interaction, the value of the metrics NOSS/NOS and NOAS/NOS should be around 50%. These two metrics are usually related; i.e. a use case with a low value of NOSS/NOS usually has a high value of NOAS/NOS and vice versa. Because of that, the causes that provokes high values in one of these metrics usually coincide with causes that provoke low values in the other one (see table 2).

According to [20] that stated that *different techniques find different things*, the analysis presented in [5] and summarised in table 2, has allowed us to identify the different *defects* usually detected by the heuristics. The result was a checklist (see appendix A) for use cases whose goal is to specifically determine the defect types that *MBR* helps to detect.

Metric	Value	Main causes
NOS	low (<3)	Incompleteness, too much modularity, triviality
NOS	high (>9)	Ambiguity, too many alternative steps, too much detach of actor and/or system steps
NOAS/NOS	low (<30%)	It does not include all that the system and the actors must accomplish to achieve the goal, too much detach of system steps, it express a batch process, it contains concrete references to user interface, it includes internal action of the system.
NOSS/NOS	high (>80%)	
NOAS/NOS	high (>70%)	It does not include all that the system and the actors must accomplish to achieve the goal, too much detach of actors steps, it include interactions between several actors or between actors and the environment of the system.
NOSS/NOS	low (<40%)	
NOUS/NOS	high (>2 5%)	Too much modularity, it includes concrete references to elements of the user interface, it express application menus.
CC	high (>4)	Understandability

Table 2. Main defect types in use cases outside usual range

3. Experiment Description

The main objective of the experiment is to ascertain is *MBR* is really more effective and efficient than simple *Checklist* technique (see Appendix A). Hereafter, we describe the experimental process, using the format (with minor changes) proposed by Wohlin *et al.* [22].

3.1. Definition

Based on the recommendations proposed in [13] to use the GQM template in experimentation in requirements engineering, the goal definition of our experiment can be summarised as:
Analyse the *Checklist* and *MBR* techniques
for the purpose of evaluating
with respect to their efficiency and effectiveness
from the point of view of the researcher
in the context of Undergraduate Computer Science students enrolled in the fifth-year at the Computer Science School at the University of Seville.

3.2. Planning

In the following subsections, we explain how the experiment was conducted.

3.2.1. Context selection. 146 students of Computer Science School at the University of Seville (Spain) carried out the experiment, hence the experiment was run off–line. The experiment is specific since it is focused on two requirements inspection techniques applied to two different application domains. The ability to generalize from this specific context is further elaborated below when discussing threats to the experiment. The experiment addresses a real problem, i.e. what technique is more effective an efficient to be used in requirements inspection.

3.2.2. Selection of subjects. The subject were selected for convenience i.e. they are undergraduate students who have extensive experience in use cases development. We divided the subjects during the experiments in two types of requirements reviewers:

- *Checklist* technique reviewers: subjects who reviewed the requirements document using the Checklist technique, presented in Appendix A.
- *MBR* technique reviewers: subjects who reviewed the requirements document using the MBR technique, presented in section 2.

3.2.3. Variables selection. In designing the experiment, we have to consider what independent variables or factors were likely to have an impact on the results. These are:

- **Defects detection technique (or reading technique (RTECH)):** This factor has two levels: the *MBR* technique and the *Checklist* technique.

- **Requirements document domain (DOC):** This factor has two levels: one of the requirements documents is of general domain (Sports Installations Reservation, SIR) and the other is related to specific and less acknowledged domain (Seeds Distinguibility Study, SDS). By means of two documents we have to avoid that specific properties of one document cloud the results of the experiment, as recommended in [20].

On the other hand, we considered three dependent variables defined according to [21]:

- **Effectiveness:** measured as the Number of defects found/Total number of defects, i.e. effectiveness means the percentage of true defects found by a reviewer with respect to the total number of defects in the inspected requirements document.

- **Efficiency:** measured as the Number of defects found/Inspection time. Where Inspection time is related to the time that subjects spent on inspecting the requirements document, measured in seconds.

- **Difficulty:** which measured how difficult it was for each reviewer to apply the corresponding reading technique. This measure was rated according to reviewers opinion, using five linguistic labels (see table 3).

Very easy	Easy	Neither	Difficult	Very difficult

Table 3. Linguistic labels for difficulty

The second dependent variable is defined in keeping with [14], that stated that although the main focus of an empirical study could be effectiveness, other performance measures like time (or efficiency) should also be analysed because they may affect the treatment.
Furthermore, a *controlled variable* was identified: *the experience with use cases technique*. We wanted to avoid that its variations cloud the results of the experiment.

3.2.4. Instrumentation. For each participant, we had prepared a folder with the experimental material[1]. Each folder contained:

- One requirements document, that could be SIR (11 pages with 21 defects) or SDS (10 pages with 13 defects). Therefore, we used two documents with a known number of defects. But inserting these defects was not necessary because the SIR and SDS documents had been written by students, i.e. the defects were made during specification of the requirements in a previous exercise done by other students. From our point of view, this situation is more realistic than artificially inserting a set of defects in the requirements documents.

- A guideline for applying the reading technique, i.e. the list of steps to carry out the search for defects. This guide is different depending on whether the subject applies the *MBR* or *Checklist* technique.

- A set of questions collected in the checklist that appears in appendix A.

- Optional, i.e. if a subject had to apply *MBR*, he was also given a summary of the heuristics presented in section 2 and the metric values for the requirements document SIR or SDS. For each metric, we indicated if it was inside the usual range or not.

[1]The experimental material is available at `http://www.lsi.us.es/~beat/ExpMater`

3.2.5. Hypotheses formulation. We want to test three groups of hypotheses, one for each dependent variable.

• Effectiveness hypotheses

$H_{0,1}$ There is no difference in effectiveness of subjects applying *MBR* technique as compared to subjects applying *Checklist* technique. //$H_{1,1} : \neg H_{0,1}$

$H_{0,2}$ There is no difference in effectiveness of subjects verifying SDS document as compared to subjects verifying SIR document. //$H_{1,2} : \neg \ _{0,2}$

$H_{0,3}$ There is no difference in effectiveness of subjects in the interaction between RTECH and DOC. //$H_{1,3} : \neg H_{0,3}$

• Efficiency hypotheses

$H_{0,4}$ There is no difference in efficiency of subjects applying *MBR* technique as compared to subjects applying *Checklist* technique. //$H_{1,4} : \neg H_{0,4}$

$H_{0,5}$ There is no difference in efficiency of subjects verifying SDS document as compared to subjects verifying SIR document. //$H_{1,5} : \neg H_{0,5}$

$H_{0,6}$ There is no difference in efficiency of subjects in the interaction between RTECH and DOC. //$H_{1,6} : \neg H_{0,6}$

Moreover we want to ascertain if any relationship exists between both effectiveness and efficiency and the difficulty when applying each technique. So we formulated the following hypotheses.

• **Difficulty hypotheses**

$H_{0,7}$ There is no relationship between difficulty and effectiveness. //$H_{1,7} : \neg H_{0,7}$

$H_{0,8}$ There is no relationship between difficulty and efficiency. //$H_{1,8} : \neg H_{0,8}$

3.2.6. Experimental Design. Taking the hypotheses into account, the experiment must consider two factors: the reading technique applied (RTECH) and the application domain of the requirements document inspected (DOC) with two levels each one. Given that we had two hours available to carry out the experiment and the number of subjects was large enough, we selected a between-subjects and blocked design (as balanced as possible because of the number of subjects available), which means that each subject was assigned only one treatment. In table 4 four groups are identified, one for each treatment.

		Technique	
		MBR	**Checklist**
Application domain	**SDS**	G1	G3
	SIR	G2	G4

Table 4. Experiment design–2x2 Factorial design

In a previous session, before the execution of the experiment the subjects were given a questionnaire in order to know their experience with the use case technique. Analysing each questionnaire the subjects were marked, and according to their marks they were assigned, using a systematic sampling, to the corresponding group. Therefore, the "*experience*" factor was, to some extent, controlled.

3.3. Operation

In this section we describe each of the steps of the operational phase: preparation, execution and data validation.

3.3.1. Preparation. We gave a seminar to the subjects of the experiment prior to the day of the experiment execution. In this seminar we explained to the subjects how to apply the technique with which they would have to review the requirements document during the experiment. The subjects of group 1 and 2 received training in the *MBR* technique, whilst the subjects of groups 3 and 4 were trained in the Checklist technique. They already had knowledge about the importance of SQA and Metrics in Software Engineering, because they had studied two themes about these disciplines three weeks previously.

3.3.2. Execution. The experiment was carried out in two classrooms. In the first classroom *MBR* technique was applied (groups 1 and 2), and in the other the *Checklist* technique (groups 3 and 4) was applied.

The students worked under examination conditions, without speaking mutually and asking the professors, who supervised the experiment, any doubt that appeared during the inspection process. The subjects had to perform the following experimental tasks:

•To manually fill out the form registering the defects in a table. For each use case in the document, the subjects had to mark those questions of the checklist that the use case did not fulfil with a cross.

•To write down the start and end time of the reviewing process.

• To rate, using a five linguistic labels scale (see table 3), the difficulty of the application of the reading technique.

• To fill out a debriefing questionnaire, which included personal details and experience.

3.3.3. Data validation. We collected the forms filled out by the subjects, checking if they were complete. When the experiment was run we realized that four subjects who had done the experiment with the *MBR* technique had previously attended to the seminar were the *Checklist* technique was explained, and not the *MBR* technique seminar. That fact must be taken into account before statistically analysing the data. The subjects had experience with use cases and approximately the same little experience in working outside the university. The average age was 22 years old and 67% were male. This information was collected from the debriefing questionnaire.

3.4. Analysis and Interpretation

The purpose is to analyse how the independent variables have influence on dependent variables. The independent variables are on a nominal scale, i.e. the reading technique used for inspection (RTECH) has two levels *MBR* and *Checklist*, the requirements document (DOC) also has two levels SDS and SIR. On the other hand, effectiveness and efficiency are on a ratio scale and difficulty is on an ordinal scale.

3.4.1. Testing effectiveness hypotheses. The purpose is to determine:

- whether *MBR* technique is more effective than *Checklist* technique.
- whether the specific or general application domain has

influenced effectiveness.

- whether the interaction between the technique applied and the type of requirements document inspected had influenced effectiveness.

Figure 1 visualises the graphical dispersion of effectiveness according to the level of the independent variables RTECH and DOC.

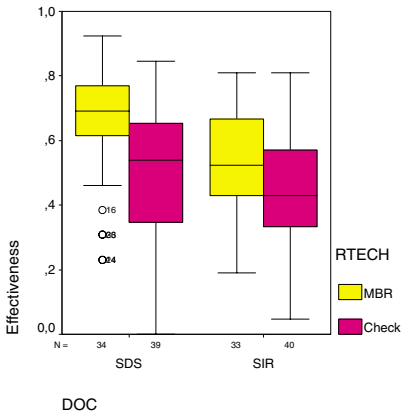

Figure 1. Box plot of effectiveness of the inspection process

Five outliers have been identified (the subjects 14, 16, 24, 26 and 33). We investigated why the effectiveness of these subjects was so low. Four of them were the subjects which had done the experiment with *MBR* but had had training in *Checklist*, i.e. they were in the wrong classroom. The other one (subject number 24) has no apparent causes to have such a low performance. Therefore, to carry out the data analysis the subjects which were in the wrong classroom were left out.

The mean of effectiveness obtained for each level of the independent variables RTECH and DOC are shown in figure 2.

For both requirements documents (SDS and SIR), the mean for effectiveness for the *MBR* technique is higher than for reviewers using the *Checklist* technique. To test the effectiveness hypothesis, firstly we evaluated if the data followed a normal distribution or not (see Shapiro–Wilk results in table 5). Even though the data of group 1 was not normal at the level of 0.05, we decided to carry out an ANOVA, considering that most of the data was normal and the statistical test robust, and could not invalidate the findings.

In table 6 the results obtained by means of ANOVA for effectiveness are shown. The first column represents the source of variation, the second column shows the sum squared, the third one represents the degrees of freedom, the

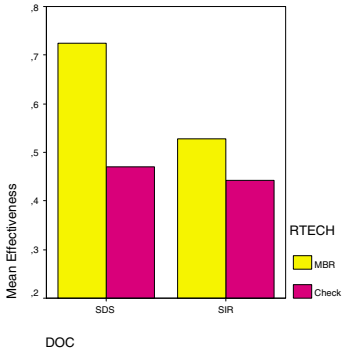

Figure 2. Bar plot of mean of effectiveness of the inspection process

Dependent variable	**Group**	**Shapiro - Wilk Significance level**
Effectiveness	Group 1	0.030
	Group 2	0.174
	Group 3	0.059
	Group 4	0.764

Table 5. Shapiro–Wilk normality test results to effectiveness

fourth column represents the mean squared, the fifth column indicates the F-ratio. The last column represents the level of significance. In each row of the table we have the two independent variables of the experiment, the interaction, the error, the total and the total corrected.

Dependent variable: Effectiveness

Source	Sum of squared	df	Mean squared	F	Significance level
Corrected model	1,456[a]	3	,485	14,119	,000
Intersection	40,527	1	40,527	1178,591	,000
RTECH	,952	1	,952	27,690	,000
DOC	,395	1	,395	11,485	,001
RTECH * DOC	,215	1	,215	6,266	,013
Error	4,745	138	3,439E-02		
Total	45,590	142			
Total corrected	6,202	141			

a. R squared = ,235 (R squared corrected = ,218)

Table 6. ANOVA of effectiveness of the inspection process

The analysis summarised in table 6 revealed a significant effect ($\alpha = 0.01$) for reading technique, i.e. the *MBR* groups have a statistically better effectiveness measured by

the percentage of defects that were found by the students. At this significance level ($\alpha = 0.01$), the variable DOC is also significant but, the interaction between RTECH and DOC is not significant at 0.01 level. This means that, although the mean effectiveness varied significantly for both documents, the effect of the reading technique is not linked to these differences. Therefore, we reject the null hypotheses H_{01} and H_{02} at the $\alpha = 0.01$ significance level. ($R^2 = 0.235$ indicates what percentage of variance in the effectiveness is accounted for by RTECH and DOC).

3.4.2. Testing efficiency hypotheses. The purpose is to determine:

- whether *MBR* technique is more efficient than *Checklist* technique.
- whether the specific or general domain has influence on efficiency.
- whether the interaction between reading technique and document has influence on efficiency.

Figure 3 visualises the graphical dispersion of efficiency according to the levels of the independent variables RTECH and DOC. As figure 3 shows there are three outliers (subjects 21,107 and 110) whose efficiency is greater than the rest of subjects. We excluded their data for the data analysis because they could have had more experience than the rest of subjects on working using the *Checklist* technique.

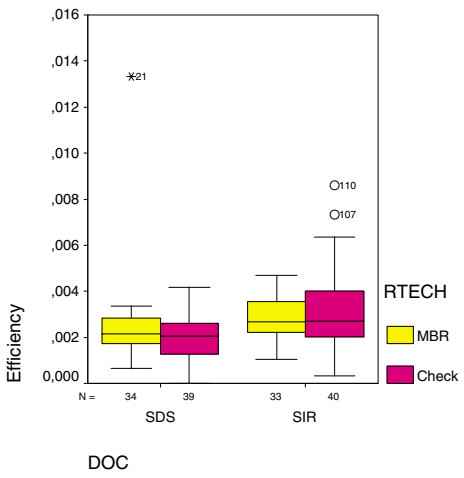

Figure 3. Box plot of efficiency of the inspection process

The mean of efficiency obtained for each level of the independent variables RTECH and DOC is shown in figure 4. The mean of the efficiency seems to be higher in SIR document than in SDS document. Nevertheless, there seems not to be any difference with respect to the reading technique.

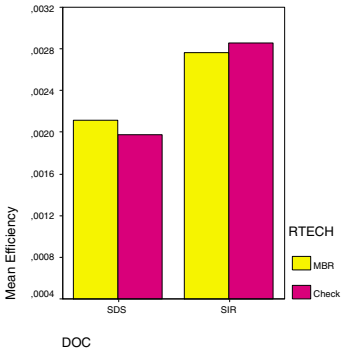

Figure 4. Bar plot of mean of efficiency of the inspection process

As commented above, ANOVA assumes that the data points of the sample group are normally distributed. In order to check this situation, four Shapiro–Wilk tests have been performed (one for each group with respect to Efficiency variable). The result of these tests (see table 7) indicates that data are normally distributed at the 0.05 level.

Dependent variable	Group	Shapiro - Wilk Significance level
Efficiency	Group 1	0.297
	Group 2	0.662
	Group 3	0.515
	Group 4	0.776

Table 7. Shapiro–Wilk normality test results to Efficiency

By means of an ANOVA we obtained the results shown in table 8, that reveals that only there is a significant difference between documents domain and this difference is not linked to the interaction between both independent variables, i.e. we can reject the hypothesis $H_{0,5}$.

Searching the causes of the difference in efficiency with respect to requirements document we have studied a new variable: the time spent in inspection process by the subject (variable Time).

The result of Shapiro–Wilk tests (see table 9) indicates that data are normally distributed.

Dependent variable: Efficiency

Source	Sum of square	df	Mean square	F	Significance level
Corrected model	2,206E-05[a]	3	7,352E-06	6,392	,000
Intersection	8,382E-04	1	8,382E-04	728,733	,000
RTECH	2,542E-08	1	2,542E-08	,022	,882
DOC	2,089E-05	1	2,089E-05	18,166	,000
RTECH * DOC	4,990E-07	1	4,990E-07	,434	,511
Error	1,599E-04	139	1,150E-06		
Total	1,022E-03	143			
Total corrected	1,819E-04	142			

a. R squared = ,121 (R squared corrected= ,102)

Table 8. ANOVA of efficiency of the inspection process

Dependent variable	Group	Shapiro - Wilk Significance level
Time	Group 1	0.799
	Group 2	0.464
	Group 3	0.441
	Group 4	0.492

Table 9. Shapiro–Wilk normality test results to Time

The means of time according to different levels of independent variables can be seen in figure 5.

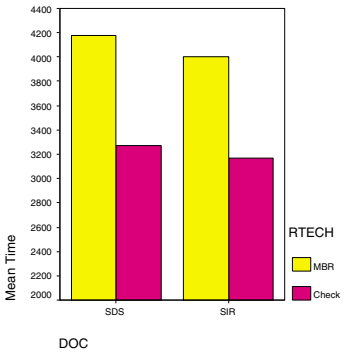

Figure 5. Bar plot of Time of the inspection process

The ANOVA (see table 10) reveals the following:

- The difference in time is statistically significant at the $\alpha = 0.01$, i.e. using *MBR* more time is spent for both documents (SDS and SIR). Since mean Time(*MBR*) is equal to 4089.62 seconds and mean Time(*Checklist*) is equal to 3266 seconds, the reviewers using *MBR* spent a mean of 25.21% more time than using *Checklist* to search for defects in use cases. Nevertheless, we believe that time used with *MBR* could be decreased if reviewers have more experience in *MBR*. Obviously *MBR* is more complex to apply than *Checklist*. However, we believe that if MBR reviewers get more experience in the technique, they could improve their efficiency. In this experiment there were really only two hours of training, and so we can not say that the subjects were experts in the *MBR* technique. For that reason for the replica of this experiment we try to consider more experts reviewers.

Dependent variable: Time

Source	Sum of squared	df	Mean Square	F	Significance level
Corrected model	27405214,6[a]	3	9135071,5	17,089	,000
Intersection	1877076100	1	1,88E+09	3511,503	,000
RTECH	26565052,5	1	26565052	49,696	,000
DOC	650981,986	1	650981,986	1,218	,272
RTECH * DOC	37878,173	1	37878,173	,071	,790
Error	73767982,4	138	534550,597		
Total	1954685555	142			
Total corrected	101173197	141			

a. R squared = ,271 (R squared corrected = ,255)

Table 10. ANOVA of Time of the inspection process

- The variable DOC and the interaction between variables DOC and RTECH are not significant. This result reveals that time is not the cause for the difference in efficiency between SIR and SDS. From our point of view, it could be because of several reasons:
  – Since SDS is a specific domain requirements document and SIR is a general domain one, it is possible that reviewers have less problems inspecting SIR than inspecting SDS. However this situation should be reflected in figure 6. We believe that the commitment level of the subjects for estimating the degree of difficulty of each technique was very low.
  – Perhaps some types of defects that contain SIR document have seemed more obvious than the ones in SDS document. If this situation is confirmed we have to be careful when selecting the objects of the experiments in the future replications of this experiment.

3.4.3. Testing difficulty hypotheses. The purpose is to determine:

- whether any relationship exists between difficulty and effectiveness.
- whether any relationship exists between difficulty and efficiency.

Figures 6 and 7 illustrate how the subjects have evaluated the difficulty for both reading technique and requirements document, respectively. From our point of view, too many subjects responded *neither*. As commented above, we believe this is because the subjects presented a certain lack of commitment when they ranked the difficulty.

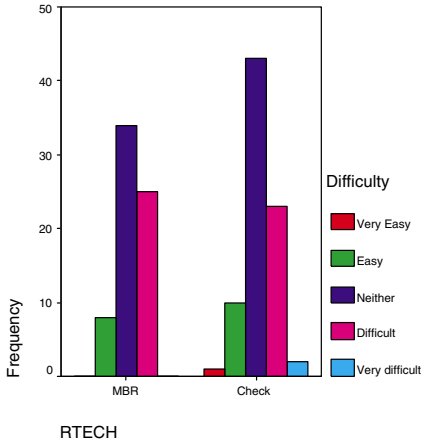

Figure 6. The reviewers' opinion about difficulty of reading technique

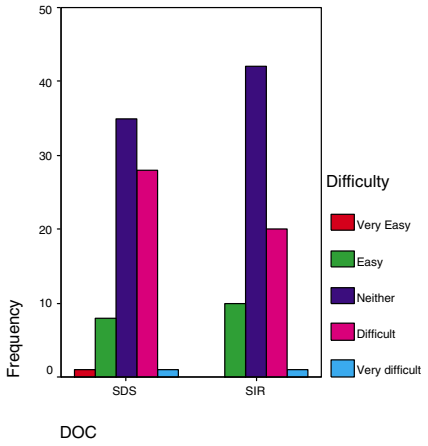

Figure 7. The reviewers' opinion about difficulty of document

Since the difficulty variable is on a ordinal scale, in order to study if it was related to effectiveness and efficiency, we used τ_b Kendall and ρ Spearman correlation coefficients. We obtained that there is no relationship between effectiveness or difficulty and between efficiency and difficulty.

3.5. Threats to Validity

In [22] a list of issues that threaten the validity of the empirical studies are identified and defined: conclusion validity, internal validity, construct validity and external validity. In this section we try to analyse those issues that could threaten the validity of the experiment.

Conclusion validity: the size of the sample it is enough for the statistical tests we applied to analyse the data. Moreover, the statistical assumptions of each statistical tests were verified. So, the conclusion validity was fulfilled.

Internal validity: this threat type concerns if there are some elements different to independent variables that could cause the outcome of the experiment. We think that the degree of knowledge of the use cases technique could have influenced the outcome. Because of that we identified this element as a controlled variable and we blocked it. On the other hand, since there are not repeated measures (between–subjects design) we eliminated the learning effect, controlling the *maturation* and avoiding the carry–over effect. Because the experiment took only two hours we avoid the effect of the *history*. According to [21], the threat of *selection* is also under control, as the experiment is a mandatory part of a course. With respect to *instrumentation*, the experimental material has been thoroughly prepared and the unique difference in it is because the under–studied independent variables.

Construct validity: this threats type concerns to the theory behind the experiment. Since the experiment includes two requirements documents (SDS and SIR) as objects we reduced the *mono–operation bias* because the cause construct is completely represented. The *mono–method bias* is also reduced by the analysis of effectiveness and efficiency dependent variables.

External validity: the greater the external validity, the more the results of an empirical study can be generalized to current software engineering practice. Three threats to validity have been identified which limit the possibility of applying any such generalization:

- **Materials used**: in the experiment we intended that the subjects were familiar with the application domains of the requirements documents. For that reason a detailed description ot them was written as introduction in the document. Nevertheless, we

believe that requirements documents might not be representative of industrial problems. Both SDS and SIR are smaller and less complex than industrial requirements documents.

• **Subjects**: to solve the difficulty of obtaining professional subjects, we used students from software engineering courses. We are aware that more experiments with practitioners and professionals must be carried out in order to be able to generalize these results. However, in this case, the tasks to be performed do not require high levels of industrial experience, so, experiments with students could be appropriate [19]. Moreover, students are the next generation of professionals, so they are close to the population under study [14].

• **Time**: the reviewers had two hours to carry out the experiment. This time is sufficient to detect defects in SDS and SIR. But we think that two hours is not a realistic period to review any requirements document of real systems. For this reason, in the future we will consider real requirements documents.

4. Related Work

In the area of requirements engineering, it has been claimed that there is a lack of experimentation to validate research results. As Kamsties and Rombach commented in [13], *empirical researches can contribute to requirements engineering by evaluating the effectiveness of techniques, methods and tools*. In this sense, some empirical researches have been done in the requirements inspection area, all of them designed to compare two or more reading techniques in order to know which of them is *better* than the other. These approaches have different characteristics, such as the notation of the requirements document under study, the scope in which experiment is realised, the experimental design that has been applied, etc. We summarise in this section the main proposals that have influenced our work.

Basili et al.'s experiment [2] studies the possible cause–effect relationship between both PBR technique and requirements documents and the defect detection rate. Furthermore [2] provides the experimental process developed in industrial environment by means of both a pilot study and a controlled experiment. This analysis concludes that teams using PBR are more effective than teams using their usual technique for requirements inspection, confirming the assumption that, focusing on specific classes of defects, the performance of inspections is improved.

In *Empirical Software Engineering* journal, two replications of a experiment conduced by Porter and published in [18] have been collected, first [11] and later [16]. This family of experiments analysed how both individual reading techniques (Scenarios, Checklist or Ad Hoc) and meetings between reviewers affect the defects detection rate in requirements documents for control systems, written using a tabular requirements notation [12]. The only result in which the three versions of the experiment coincide is that collection meetings contributed nothing to fault detection effectiveness. About the effectiveness of Scenarios against Checklist and Ad Hoc, the results are ambiguous, but on balance it seems that Scenarios is more effective than Checklist or Ad Hoc. The ambiguous results of this family of experiments revealed how important the realization of replications of the experiments is and that it is convenient to change some design aspects, trying to solve some designs errors committed in the original experiment.

In [15], an abstraction mechanism applied during PBR inspections is empirically analysed in order to study its effectiveness. The idea behind *error abstraction* is to group similar defects (i.e. defects with the same cause) that are joined to identify an error category. The three perspectives used by the reviewers were designer, tester and use case creator. The experiment was developed in an academic environment and on average, students found 1.7 additional true defects as a result of applying error abstraction process and reported an average of 3.7 faults/error.

Related to empirical validation of use case inspection techniques, in [1] a checklist–based inspection technique for detecting defects in use cases is proposed. That inspection technique was evaluated in two studies with students as subjects. The findings of the experiments revealed that inspections are useful for detecting defects in use cases.

5. Conclusions and Future Work

The requirement inspection is a critical activity throughout the software development process, whose success is fundamental for the quality of the software product that it is finally delivered. With this idea in mind we have proposed a new reading technique called *MBR* technique which is based on some metric–based heuristics. For gathering empirical evidence that really *MBR* is more efficient and effective than a simple *Checklist* technique we carried out a controlled experiment. Moreover, the experiment studied how the application domain of the requirements documents could influence the effectiveness and efficiency when detecting defects.

With respect to effectiveness, from this experiment it is concluded that reviewers using *MBR* are more effective than those using *Checklist* in two different application domains, both related to management applications. With respect to efficiency, it is concluded that there are no significant differences between the efficiency of the *MBR* technique and a simple *Checklist* technique, however reviewers using *MBR* spent more time on inspecting the requirements document

than *Checklist* reviewers. This fact is, to some extent, normal given that *MBR* could be a bit more complex, rather than a simple checklist. But, in general, software developers prefer a requirements document without defects, even if it takes more time to identify them, because requirements documents constitute the backbone of the software products that are finally delivered.

Even if the obtained results are encouraging, we must consider them as preliminaries. The empirical results that have been presented in this study must be interpreted with caution. Several threats to the study's validity have been outlined and discussed. It is our belief that it is necessary to make a family of experiments to increase the external validity of the results to the extent that the conclusions currently presented can be generalized. Such a family of experiments should also have to use professionals of the software development process as subjects and requirements documents taken from real systems. Besides we are conscious of the necessity to make laboratory packages with the information of the empirical studies, to encourage their external and independent replication and obtain a body of knowledge about the utility of the *MBR* technique. This will eventually contribute to reviewers making better decisions in the early phases of software development. After all, this is the most important goal we pursued when we proposed a new technique for inspecting requirements documents.

6. Acknowledgments

We want to thanks Isabel Ramos from the University of Seville for allowing us to perform the experiment with their students, Esperanza Manso from University of Valladolid by her aid in statistics and Mari Carmen Otero from the University of Basque Country for her valuable comments related to the experimental design.

A Checklist to use cases inspection

Completeness:

- Does every requests of actor to the system obtain system response and vice versa?
- Does the use case contain all the complete set of possible scenarios for achieving a goal?
- Do alternatives of the use case specify all different results to ordinary sequence?
- Does the use case consider all the possible exceptions to ordinary sequence?

No ambiguity:

- Has the use case more than one possible interpretation?

Understandability:

- Is it possible to read the use case without excessive re–reading?
- Is difficult to follow the use case sequence because of its *include* or *extend* relationships?
- Is difficult to follow the use case sequence because of its alternative steps?
- Are the actor or system steps too detached?

Conciseness:

- Could its meaning be expressed in fewer words?
- Is it written with too much detail?
- Are there elements that could be obviated?
- Are unnecessary remarks that make reading difficult the lecture included?
- Are interactions between actors and elements of the system environment or between primary actors and secondary actors included?

No triviality:

- Is the use case named as user goal that the system will support?
- Does the use case reflect a *result of value* to actor, not a set of trivial interactions?

Proper use of use cases technique:

- Does the use case represent an external processing that requires any user participation?

Design independence:

- Does the use case use the customer language, no words that reveal detail of design or implementation?
- Does the use case contain concrete references to user interface elements?
- Does the use case describe what is to be accomplished by the system but not how?
- Is the use case not focused on internal processing?
- Does the use case try to anticipate how application menus will be like?

References

[1] B. Anda and D. Sjøberg. Towards an inspection technique for use case models. In *Proceedings of the 14th Software Engineering Knowledge Engineering (SEKE'02)*, Ischia, Italy, 2002.

[2] V. R. Basili, S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Sørumgård, and M. V. Zelkowitz. The Empirical Insvestigation of Perspective–Based Reading. *Empirical Software Engineering*, 1(2):133–164, 1996.

[3] B. Bernárdez, A. Durán, and M. Genero. Empirical Assessment of a Defect Detection Technique for Use Cases. In *Proceedings of the 2nd Workshop on Software Quality of the 26 th International Conference on Software Engineering*, Edinburgh, 2004.

[4] B. Bernárdez, A. Durán, and M. Genero. *Metrics for Software Conceptual Models*, chapter Metrics for Use Cases: A Survey of Current Proposals. Imperial College, 2004. *To be published.*

[5] B. Bernárdez, A. Durán, and M. Genero. Empirical Evaluation and Review of a Metrics–Based Approach for Use Case Verification. *Journal of Research and Practice in Information Technology, Special Collection on Requirements Engineering*, *To be published*.

[6] B. W. Boehm. *Software Engineering Economics*. Prentice–Hall, 1981.

[7] L. C. Briand, S. Morasca, and V. R. Basili. Defining and Validating Measures for Object–Based High–Level Design. *IEEE Transactions on Software Engineering*, 25(5):722–743, September–October 1999.

[8] A. Durán, A. Ruiz, B. Bernárdez, and M. Toro. Verifying Software Requirements with XSLT. *ACM Software Engineering Notes*, 29(1):39–44, 2002.

[9] A. Durán, A. Ruiz-Cortés, R. Corchuelo, and M. Toro. Supporting Requirements Verification using XSLT. In *Proceedings of the IEEE Joint International Requirements Engineering Conference (RE'02)*, Essen, Germany, 2002.

[10] M. Fagan. Design and code inspections to reduce errors un program development. *IBM System Journal*, 15(3):182–211, 1976.

[11] P. Fusaro, F. Lanubile, and G. Visaggio. A Replicated Experiment to Assess Requirements Inspection Techniques. *Empirical Software Engineering*, 2(1):39–57, 1997. Available in`http://www.kluweronline.com/issn/1382-3256`.

[12] K. L. Heninger. Specifiying Software for Complex Systems: New Techniques and Their Application. *IEEE Transactions on Software Engineering*, SE-6(1):2–13, June 1980.

[13] E. Kamsties and H. D. Rombach. A Framework for Evaluating System and Software Requirements Specification Approaches. *Lecture Notes in Computer Science*, 1526, 1997.

[14] B. A. Kitchenham, S. L. Pfleeger, D. Hoaglin., K. E. Emam, and J. Rosenberg. Preliminary Guidelines for Empirical Research in Software Engineering. *IEEE Transactions on Software Engineering*, 28(8):721–734, August 2002.

[15] F. Lanubile, F. Shull, and V. R. Basili. Experimenting with Error Abstraction in Requirements Documents. In *Proceeding of the 5th International Symposium on Software Metrics*, Bethesda, Maryland (USA), 1998.

[16] J. Miller, M. Wood, and M. Roper. Further Experiences with Scenarios and Checklists. *Empirical Software Engineering*, 3(1):37–94, 1998. Available in`http://www.kluweronline.com/issn/1382-3256`.

[17] D. L. Parnas and D. Weiss. Active Design Reviews: Principles and Practices. In *Proceedings of the 8th International Conference on Software Engineering*, London, UK, 1985.

[18] A. A. Porter, L. G. Votta, and V. R. Basili. Comparing Detection Methods for Sofware Requirements Inspections: A Replicated Experiment. *IEEE Transactions on Software Engineering*, 21(6):563–575, June 1995.

[19] A. A. Porter, L. G. Votta, and V. R. Basili. Building Knowledge through Families of Experiments. *IEEE Transactions on Software Engineering*, 25(4):456–473, July 1999.

[20] F. Shull, J.Carver, G.Travassos, J. Maldonado, R. Conradi, and V.Basili. *Lecture Notes on Empirical Software Engineering*, chapter Building a Body of Knowledge about Software Reading Techniques, pages 39–84. A World Scientific Singapore, 2003. Eds. Juristo N. and Moreno A.

[21] T. Thelin, P. Runeson, C. Wohlin, T. Olsson, and C. Andersson. Evaluation of Usage–Based Reading–Conclusions after Three Experiments. *Empirical Software Engineering*, 9(1–2):77–110, 2004. Available in `http://www.kluweronline.com/issn/1382-3256`.

[22] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, 2000.

[23] M. Zelkowitz and D. Wallace. Experimental validation in software engineering. *Information and Software Technology*, 39(11):735–743, 1997.