

A Novel Checklist: Comparison of CBR and PBR to Inspect Use Case Specification

Asma Naveed^{1,3} and Naveed Ikram^{1,2(✉)}

¹ Riphah International University, Islamabad, Pakistan
asmanaveed@fui.edu.pk, naveed.ikram@riphah.edu.pk

² Ibn-e-Sina Empirical Software Engineering Lab, Islamabad, Pakistan

³ Foundation University Islamabad, Rawalpindi, Pakistan

Abstract. High quality and cost effective software development entails early detection of errors from requirement specification artifact/s. For this purpose, various inspection techniques have been presented to identify requirement specification errors. In most reported studies, comparison of two commonly used inspection techniques CBR (Checklist Based Reading) and PBR (Perspective Based Reading) had been conducted to identify defects from the UCS (Use Case Specification); however no comparison was done based on IEEE STD 830-1998 defects' types. Therefore, a novel checklist was developed to identify the IEEE STD 830-1998 specified defects' types namely Ambiguosness, Incorrectness, Inconsistency and Incompleteness from UCS, a major contribution of this research. This developed checklist was later validated to be utilized during this experimental research. In this study, a quasi-experiment was conducted with industrial professionals to compare the effectiveness and efficiency of CBR and PBR using the developed checklist to inspect the UCS that was specified in Use Case 2.0 format. The result of this research showed significant difference between CBR and PBR, i.e. PBR found more defects for all defects' types compared to the CBR technique, but CBR reported less False Positive defects by applying the developed checklist for all defects' types. It was also proved that CBR is more efficient (time based) than PBR for all defects' types. These findings will provide guidelines to industry practitioners for the selection of an inspection technique based on effectiveness, efficiency and false positive ratio for a particular type of defect.

Keywords: Use case specification · Inspection techniques · Defect taxonomies · Checklist based reading · Perspective based reading · Requirement testing

1 Introduction

To remain competitive, the software industry strives to develop high quality products with minimum cost and less development time. However, one major obstacle in attaining such products is late identification of defects. According to Pressman, [1] "...Some maladies, as doctors say, at the beginning are easy to cure but difficult to recognize... but in the course of time... become easy to recognize but difficult to cure". High quality software requires error detection during the requirement phase of software development,

because the errors, if not identified in the requirement phase of software development, might propagate to the next phases and result in poor quality and high cost of software product [4].

Various methods for the detection of errors from the requirement specification have been reported in literature e.g. reviews, walkthroughs and inspections [5]. Fagan [6] reported that IBM inspections detected 90% of total defects over the lifecycle of a product, and had a 9% reduction in the average project cost as compared to cases in which walkthroughs were applied. This indicates that quality can be ensured by conducting inspection to identify errors from software requirement specification. The most essential step in inspection is the defects detection phase [7], where the inspectors try to individually identify maximum defects in the document. Therefore, the present research focuses on optimization of this step.

Software requirement are divided into two categories: Functional Requirements FRs and Non-Functional Requirements NFRs [8]. FRs are commonly described as Software Requirement Specification (SRS) and UCS, mostly in the form of UCS [9]. The UCS method was initially developed by Jacobson [10]. It is mostly used in the industry to specify requirements due to adoption [9] by the Unified Modeling Language UML [11] which is a de facto standard [12].

Since the errors present in the specification are not known beforehand, it can't be claimed that any inspection technique has detected all the errors in a UCS. However, it can be ensured using the presented checklist that the UCS has been inspected against IEEE specified defects' types.

This significant endowment of this research is a tailored checklist available at: <https://www.dropbox.com/s/zoew6as6dbc4ey6/checklist-2.docx?dl=0> {Date visited: August 25, 2015} developed on the basis of defects' types that will assist practitioners in early detection of defects from UCS. It guarantees that inspected UCS has been verified for the presence of quality attributes of Unambiguousness, Correctness, Consistency and Completeness as recommended by IEEE STD 830-1998 [14].

This paper also aims to provide a thorough understanding of the inspection techniques CBR and PBR by comparing them on the basis of defects types from the UCS. Finally, it provides a guideline for the practitioners to choose a better inspection technique in terms of effectiveness and efficiency.

The rest of this paper is divided as follows: Section 2 describes the related work presented earlier, while Sections 3 elucidates experiment design and experiment execution respectively. Section 4 provides a discussion on the results of this research. Sections 5 and 6 present the conclusion and Future Work respectively.

2 Related Work

In previous researches, many experiments reported the comparison between the effectiveness and efficiency of CBR and PBR for the Requirement Specification, and UCS. At the NASA Goddard Space Flight Center, authors empirically proved a remarkable improvement in the effectiveness of the PBR (User, Designer and Tester perspective) in comparison to CBR [15]. A similar result was also reported by a replicated experiment that compared PBR to Ad-hoc and CBR giving similar results [16].

To evaluate the effectiveness and efficiency of PBR (tester perspective) and CBR, a case study was conducted with industrial professionals [17]. The outcome supported PBR (Tester perspective). However, the results of a single case study cannot be generalized. This factor motivated us to conduct an experiment with repeated trials, to compare PBR and CBR in other industrial settings, and prompted us to test the inspection techniques' effectiveness and efficiency for the defects' types in order to improve the overall defects coverage. Another issue in this study was that only the tester perspective was selected while the other perspectives like user and designer also need to be included to achieve higher defect detection rate.

A few experiments, however reported contradictory results. PBR was reported to be less effective as compared to CBR using students as subjects [19][21]. Similarly, in another experiment with students as subjects, PBR (Use Case Analyst, Structured Analyst and Object-Oriented Analyst) again proved to be less effective than CBR and Ad-hoc [20]. The possible reasons for these contradictory results and failure of PBR may be improperly defining the inspectors' role, and providing insufficient instructions to subjects to apply active guidance.

In another experiment, the evaluation of PBR with client and developer perspectives was conducted using UCS as an object of study [22]. The defects were reported for both perspectives of PBR with respect to UCS format such as names of actors and use cases, flow of events, variations and so on. During the inspection, both perspectives used the same checklist. The outcome indicated that inspection effectiveness could be improved by applying different perspectives because different defects were identified by using client and developer perspectives [22].

Previously, a quasi-experiment was performed to examine the impact of active guidance [23]. The checklists and reading scenarios of PBR were designed to be identical, to examine especially the factor of active guidance. It was evident from the outcome that due to the element of active guidance, inspectors with PBR could identify more subtle defects compared to CBR. PBR proved to be more effective but less efficient as compared to CBR because the inspectors had to develop the work product during the inspection process.

Based on the above mentioned literature review, it is established that previous researches did not compare PBR and CBR on the basis of defects' types to ensure maximum coverage of defects detection from the UCS. Therefore, the present research compares CBR and PBR on the basis of defects' types to detect maximum defects from the UCS.

To detect defects during the inspection of requirement artifacts, various defect taxonomies or classifications have been presented in literature. As recommended by the IBM [24], to attain maximum defects detection rate, the inspection technique should be applied by focusing on defect taxonomies which are generic i.e. independent of process or product, so these taxonomies can be applied consistently to any deliverable product like requirement specification, design and code documents. IEEE STD 830-1998 [14] and Neill and Laplante [9] state that the presence of defects i.e. Ambiguousness, Incompleteness, Inconsistency and Incorrectness etc. in requirement specification would have an adverse impact on quality and often require major revisions, or may cause system failure entirely. According to Krogstie [25], syntax and semantics defects can be caused

by the absence of quality attributes (Unambiguousness, Correctness, Completeness and Consistency). After analyzing literature for the defect taxonomies regarding SRS, it is concluded that most of the presented defect taxonomies were generic, and applicable to all phases of software development [29]. However, a few reported defects taxonomies were not generic e.g. SRS Structure based defect taxonomy can only be applied to requirement phase, focusing only on Consistency defect type [31]. Consequently, it partially detected syntax and semantics defects.

Similarly, for Use Case Specification, many defect taxonomies for defects detection were reported to be generic [32][33][34]. However in a few studies, the defect taxonomy was not generic (structure specific) e.g. as in [23], so it can only be applied to the requirement phase. Also, most previous reported defect taxonomies considered the syntax and semantic defects [32][34][23] but in some reported studies, the defect taxonomy only focused on syntax defects [33]. After analyzing the aforementioned defect taxonomies it has been concluded that one must consider a generic defect taxonomy that is independent of phase, process and product. Therefore, in this research the defects types of Ambiguity, Incorrectness, Inconsistency and Incompleteness are being used to detect defects from UCS as recommended by IEEE STD 830-1998 [14]. Our defect taxonomy is generic and considers syntax and semantic defects.

Inspection techniques are applied to detect defects in the under inspection artifacts. In the literature, the most referred inspection techniques are CBR [33][34][23], Ad-hoc [29], Scenario Based Reading (SBR) [21], PBR [21][34][22], Usage Based Reading (UBR) [37][38], Technique for Use Case Model Construction and Construction-based Requirements Document Analysis (TUCCA) [40] and Metric Based Reading (MBR) [41]. The comparison of CBR in most reported experiments was carried out with aforementioned inspection techniques. CBR and PBR have been selected for evaluation because during the analysis of previous work, it was concluded that CBR is the most preferred inspection technique in the software industry [39]; and PBR offers multiple roles/perspectives e.g. User, Designer and Tester etc. based inspection [21][34][22][16]. Thus by applying different roles, different defects can be identified from the same document, leading to maximum coverage. SBR does not support role-based inspection whereas UBR [37][38] is similar to the PBR user perspective, except that in UBR the use cases are prioritized by the user and hence only covers one perspective of PBR. Therefore, PBR is preferred over UBR and SBR. The TUCCA [40] technique is not applicable in the present experiment to inspect UCS because it is designed to inspect the requirement document. The MBR [41] has not been applied as it was mentioned by the authors of MBR technique that during their experiment, the inspectors found it difficult to understand, and unlike PBR, the MBR technique would not result in any document/s that can be used during the later software development phases. Therefore, in the present research, the CBR and PBR inspection techniques have been selected for comparison. Both PBR and CBR are Process and Product independent i.e. they can be applied to any artifact and tailored according to any artifact for which inspection has to be carried out [23]. Unlike CBR, which only guides the inspector what to find, PBR guides the inspector how to find it and also provides role-based inspection. Thus it is established that many experiments were conducted to compare the effectiveness and efficiency of CBR and PBR for the defects identifica-

tion from the UCS but the previous researches did not compare PBR and CBR on the basis of defects' types to ensure maximum coverage of defects detection from the Use Case Specification.

3 Research Methodology

3.1 Experiment Design

A quasi-experiment using Active inspection without meeting [36] was performed to compare the Effectiveness, Efficiency and False Positive Ratio of two inspection techniques CBR and PBR for a particular type of defect in the UCS. The inspection process of experiment is carried out on two different UCS documents, with different subjects/ persons of different organizations / places at different time.

List of Study variables is presented in Table 1.

Table 1. Study Variables

Independent Variables (IV)	Dependent Variable (DV)	Confounding Variables
Inspection techniques (PBR,CBR)	Effectiveness (number of defects)	Object: Controlled UCS Document (complexity approximately same)
	Efficiency (Time needed for the inspection of a UCS in minutes)	Controlled Environment (same working hours, project load, morning duties etc.)
	False Positive Ratio	Controlled Subject (qualification, experience) Uncontrolled (motivation level)

Research Question

Based on the discussion described in section 2, the following question hoists:

“Is there any difference in CBR and PBR inspection techniques in terms of Effectiveness, Efficiency and False Positive ratio for defects' types in Use Case Specification, or not?”

Hypothesis Development

H1₀: There is no significant difference in the ambiguity's number of defects for CBR and PBR.

H1₁: There is a significant difference in the ambiguity's number of defects for CBR and PBR.

H2₀: There is no significant difference in the ambiguity's efficiency for CBR and PBR.

H2₁: There is a significant difference in the ambiguity's efficiency for CBR and PBR.

H3₀: There is no significant difference in the ambiguity's false positive ratio for CBR and PBR.

H3₁: There is a significant difference in the ambiguity's false positive ratio for CBR and PBR.

Similarly, hypotheses were also developed to evaluate the Effectiveness, Efficiency and False Positive Ratio of other residual defects' types of Incorrectness, Inconsistency and Incompleteness.

Development of Instrument (Checklist)

In CBR, a checklist (list of questions) is provided to inspectors. A survey of the German software industry [36] showed that CBR [6] is the most frequently applied reading technique for requirement inspection. According to a survey of 117 checklists [35], most of them have twenty or more questions, and it suggests that the checklists size should not exceed one page as a longer checklist produces a wrong result due to the cognitive overload of the inspector.

In the past, most of the checklists to inspect SRS were developed by populating the checklist with questions focusing on defects types [17][21]. Similarly, to detect the defects from the UCS, checklists were also used [32][30][33][34][23]. Many of these presented checklists were developed using defect taxonomy based on the Use Case template format but the problem was that some of the detected defects were not clearly related to any particular category of use case format [30]. Due to incorrect allocation of defect into a wrong category, the distribution of defects may be reported wrongly. Consequently, this kind of incorrect reporting creates problems during the correction phase of inspection.

Most of the defects identified by applying the checklist were syntax-related rather than semantic in nature [32]. Whereas, the recommendation [36] for the requirement inspection is that the inspection technique must help to expose more semantic defects in comparison to syntactic defect, because in the later stages of software development the semantic faults are more costly and harder to fix.

A checklist was developed after reviewing theories of text comprehension presented by the discourse analysis community [34] because the use cases are written in the form of structured English. The problem with this approach is that different inspectors may report differently because most of the 7Cs are semantic in nature. Hence, subjectivity is unavoidable. Consequently, the amount of detected defects by applying a checklist would be highly dependent on the ability of the inspectors rather than on the inspection process itself. However, the guidelines recommend that the inspection process should be independent of the inspectors' ability to identify defects. In the past, tailored checklists for three different perspectives i.e. User, Designer and Tester were presented [23]. The questions of the checklists were populated by focusing on some of the subsequent defects types of Incompleteness, Inconsistency, Incorrectness, Hard-to-understand, Over-specified and Unfeasible aspects according to a particular perspective. The drawback of these checklists was that they did not consider questions related to other defects types while developing the checklists for a particular perspective e.g. for User Perspective, they should have also considered the Ambiguity's defect type because one cannot accurately find the defects of Incompleteness

and Incorrectness if an ambiguous UCS has to be inspected. Another problem with this checklist is that most of the questions were designed at an abstract level. The whole inspection process is dependent on the checklist/instrument that must be developed with an appropriate level of detail, so that the inspector can understand which defect to find and where to locate it.

Thus, it is concluded that in the past, no checklist was presented on the basis of defects' types to detect defects from the UCS. While using a checklist based on defects' types; it can be assured that inspected UCS has attained maximum coverage of IEEE quality attributes. Therefore in this research, a checklist has been developed on the basis of IEEE defects' types.

To develop this tailored checklist, the earlier presented checklists for the UCSs were reviewed [33][34][22][28], and related questions were sifted out and placed in a new checklist according to a particular defect's type. Additionally, the already-reported effective writing rules of UCS were analyzed by considering the fact that missing an effective writing rule will lead to a particular type of defect in the UCS [26][27][18][13][3]. The questions of checklist are described with details about which defects to find and where to find them in the UCS. Both syntax and semantics related defects can be identified by using developed checklist to detect defects of all types. Also, the length of the checklist for each defect type is kept up to one page to avoid both cognitive overloading and inefficiency of the instrument.

Instrument Validation

Practitioners of the Quality Assurance Departments of software companies B, O and I were requested to review the instrument in order to assess its validity. The inspectors pointed out some redundant questions. The feedback of the pilot study was implemented accordingly i.e. redundant questions were eliminated to improve the efficiency and reliability of the instrument for the defects detection.

Pilot Study

Prior to the execution of the actual experiment, a pilot study was carried out to assess the research design and adequacy of the experimental material. This study was conducted with the help of a Software Development Company P. Three inspectors were nominated by the Quality Assurance Department with similar qualifications and years of professional experience. They were assigned the inspection task randomly. A one hour session was arranged for the inspectors to be acquainted with the inspection process requirements. Three defects of each defect's type were injected to determine the validity and reliability of the instrument/checklist.

The inspectors pointed out some redundant questions in the checklist. They also suggested that some questions must be mentioned with examples for more clarity. Therefore, the presented instrument/checklist was revised based on their feedback.

Experiment Procedure

Sampling

True representative of software industry, software companies E and T have been selected on the basis of high-level maturity of CMMI (Capability Maturity Model Integration) i.e. Company E with CMMI level 3 and Company T at level 5. The experiment was carried out with these 2 companies; 3 participants and 10 UCSs from each company that leads to a total of 60 trails.

Selection of Subjects

The population of the present experimental research is professionals from the software development sector. Three participants with similar qualifications and same years of professional experience were nominated by the Quality Assurance (QA) Department of two Software Companies E and T. These nominated participants also had similar experiences of inspection activities. Afterward, these inspectors were assigned their tasks randomly.

Selection of Object

After the discussion with the software companies, ten use cases were finalized considering the inspectors other project commitments. Companies were requested to provide those documents which have inter-dependent UCSs because they may cause subtle defects e.g. inconsistency defects etc. We put in our best efforts to collect UCSs with almost same complexity level.

Both provided UCSs were written for the database management projects. The validity of document can be judged as the Company E's software is operational in the domain of Hospital Information Management based on this specification. The same criterion is valid for Software Company T's UCSs.

Table 2. Severity Levels of Injected Defects

Level of Severity	Example
Less Severity (Internal to use case)	Ambiguity: The Pronoun 'he/she' is used instead of User Subject. Inconsistency in sequence number of alternative.
Moderate Severity (Intra use cases)	Inconsistency: Use Case 2 and Use Case 3 have the same name i.e. Register a patient.
More Severity (Requirement Specification level)	Incompleteness: Missing exceptional flow/ Incomplete precondition. Incorrectness: Incorrect post condition

After receiving the document, three defects of each defect type were injected into each Use Case. So overall in both software companies provided UCSs, 120 defects of different severity levels (examples of which are mentioned in Table 2) for each defect type were injected.

3.2 Experiment Execution

The experiment was performed within four consecutive days. The time taken by each inspector differed depending upon the inspection technique and his personal capability. On an average, 2-3 hours per day were spent on the experiment. A total of six participants constituting Group I and II executed the inspection at their own work places for their convenience. Like the pilot study the inspectors from Groups I and II were given separate sessions of approximately one hour about the inspection process. This session also included a question-answer part to resolve the participants' confusions and queries. The inspectors from both companies attended the same session conducted by the same instructor (researcher) to ensure the same understanding level. Additionally, the same material i.e. different procedures to inspect documents with CBR, PBR user and designer perspective, defect injected UCS, different defect logging forms according to defects' types and the same instrument i.e. a checklist, was provided. The participants from both companies were asked to specify the description of the defect for confirmation and evidence. The inspectors were also requested to log/register the total time to identify the defects from a UCS to calculate the efficiency of an inspection technique.

3.3 Validity

Internal Validity

In the present study, best efforts were made in order to control and eliminate threats to internal validity, but threats to internal validity usually cannot be completely avoided. In this experiment, as mentioned earlier, the selection of the software companies was not random but based on the maturity level and the availability of requirements documentation in the UCS format. Similarly, the selection of inspectors was not random but was done by the software company itself considering the availability and relevance with the area of requirement testing. To avoid threats to internal validity, the allocation of inspection techniques was done randomly within the delegated teams from both companies.

The inspectors were given ten different UCSs with different injected defects, to eliminate the threats of the learning effect that can influence the internal validity of the experiment. Another threat to the internal validity was the previous knowledge of the participants, which was controlled in the present experiment by the selection of inspectors/participants having almost the same qualifications and years of experience in the field of requirement testing. For the participants' convenience, the study was conducted in their companies and they were requested to avoid sharing the experiment's outcome with each other. Also, as mentioned earlier, the UCS document with the same seeded defects, same instrument i.e. checklist and same training by the same resource person was provided to avoid threats to internal validity.

External Validity

In this research, the external validity is achieved by using an independent design of the experiment. During the experiment, the sample of objects and subjects was taken from two different software companies. So the inspection process of experiment was

carried out on two different UCSs documents with almost the same complexity, but with different subjects/persons of different organizations/places at different times. The reporting of the defects was done by using the self-inventory method of test administration. The participants were asked to specify the description of the defect for confirmation and evidence. By analyzing the description of defects, it can be identified that the detected defects are actual defects and that the inspectors had followed the inspection process as required. The confirmation that the inspection process was rightly applied can also be done by participants/inspectors of experiments who were assigned inspection technique PBR (user and designer perspective), since they write the UCSs and draw the state diagrams respectively, as a procedural step.

Conclusion Validity

To avoid threats to the conclusion validity, the received results, after the execution of the experiment were compiled and carefully checked. The reliability and understandability of the instrument is essential for the conclusion validity, therefore after the execution of the pilot study, redundant questions were eliminated. To remove the threats of “poor reliability of treatment implementation” [48], a self-inventory of defect logging with question numbers along with the description of detected defects was carried out. In addition to this, reliability of treatment implementation was confirmed by artifacts of UCSs and state diagrams developed as a part of PBR inspection technique. The participants were also asked to conduct the experiment in their own work environment for their convenience. In this way, external disturbances were avoided to eliminate the threats of “random irrelevancies in the settings” [2]. Also, by considering 0.05 as a value of significance, credibility of the results was assured.

Construct Validity

The practitioners of QA Department of Software Companies (B, O, P and I) carried out the validation of construct/instrument/checklist. The instrument was later revised by implementing their recommendation.

4 Results and Analysis

4.1 Data Preparation

The Inspectors-submitted results of each inspection technique were matched with the seeded defects. The inspectors also detected unseeded defects. These unseeded defects were separated and analyzed with the help of a domain expert to determine if they were true or false positive defects. The final defects for the PBR were obtained by taking the union of detected defects for the user and designer perspectives of PBR in the case of Seeded, Unseeded and False Positive defects as a procedural requirement of inspection technique. While calculating the efficiency of the PBR inspection technique, time taken by both perspectives was compared and the PBR perspective that consumes more time for defect detection was considered.

4.2 Data Analysis

The objective of this study is to compare the Effectiveness, Efficiency and False Positive Ratio between CBR and PBR on the basis of defects' types. To compare the defect detection rate (effectiveness) of CBR and PBR for the defects' types, 30 defects of each defect's type were seeded in each UCS.

Table 3. Comparison of CBR and PBR for Seeded, Unseeded True and False Positive Defects for companies E and T

		Ambiguity	Incorrectness	Inconsistency	Incompleteness	
Seeded Defects	Total Seeded Defects for both companies E and T	30	30	30	30	
	Company E	CBR	19	18	14	13
		PBR	23	28	26	13
	Company T	PBR	21	25	27	19
		CBR	20	24	24	13
Unseeded Defects	Total True Un-seeded Defects for Company E	61	52	32	100	
	Company E	CBR	47	28	16	39
		PBR	54	43	30	85
	Total True Un-seeded Defects for Company T	26	14	14	35	
	Company T	PBR	26	14	14	35
		CBR	12	7	5	22
False Positive Defects	Total False Positive Defects for Company E	58	83	45	57	
	Company E	CBR	29	18	15	19
		PBR	42	72	41	50
	Total False Positive Defects for Company T	18	28	20	30	
	Company T	PBR	18	28	20	30
		CBR	7	16	6	13

The results of Table 3 show that PBR found more Seeded defects for both companies E and T.

The Unseeded True defects identified by experts for all defects' types were checked against inspectors' reported CBR and PBR results as shown in above Table 3. A noteworthy verdict is that PBR reported more defects than CBR for all defect types and the PBR detected defects included all the CBR detected defects besides other detected defects. Interestingly, the number of Incompleteness type of defects is very high. A possible reason for this increase in Incompleteness defect' type may be Incorrectness and Inconsistency types of defects which were also interpreted by inspectors as Incompleteness' type of defects.

The false positive defects identified by expert for all defects' types were checked against inspectors' reported CBR and PBR results. Compiled results shown in Table 3 also indicate that PBR identified more false positive defects as compared to CBR which is not a supportive argument for an inspection technique. The possible reason for PBR detecting more false positive defects is that as a part of required procedure of PBR, two inspectors with different perspectives were inspecting the UCSs.

Table 4. Effectiveness of Inspection techniques for Companies E and T

		Ambiguity	Incorrectness	Inconsistency	Incompleteness
Company E	CBR	73%	56%	48%	40%
	PBR	85%	87%	90%	75%
Company T	CBR	57%	70%	66%	54%
	PBR	84%	89%	93%	83%

During the analysis of results, effectiveness was calculated for CBR and PBR separately. The results of Table 4 show that PBR is more effective than CBR to identify all types of defects.

Table 5. Efficiency of Inspection Techniques for Companies E and T

		Ambiguity	Incorrectness	Inconsistency	Incompleteness
Company E	CBR	55	51	25	43
	PBR	22	20	12	18
Company T	CBR	38	34	33	43
	PBR	21	17	18	21

Efficiency was calculated as number of true defects identified per hour, listed in Table 5. It can be concluded from results that PBR is less efficient. The reason may be that the inspectors applying the PBR need more time because they have to develop artifacts and model too.

Table 6. False positive defect ratio for Companies E and T

		Ambiguity	Incorrectness	Inconsistency	Incompleteness
Company E	CBR	50%	22%	33%	33%
	PBR	72%	87%	91%	88%
Company T	CBR	39%	57%	30%	43%
	PBR	100%	100%	100%	100%

False positive ratio was calculated as: (false positive defects identified by inspection technique) ÷ (expert's reported false positive defects) presented in Table 6. Both companies' presented results indicated more false positive defects ratio for PBR. Moreover, PBR reported 100% false positive defects for all defects' types in company T because PBR found same defects which were also found by CBR, in addition to other defects.

Analysis on the Basis of Statistical Results

For the generalization of results, statistical analysis was performed. The difference between effectiveness of CBR and PBR is found by applying non-parametric chi-square test at 1 degree of freedom as our experiment design has one factor (inspection technique) with two treatments (CBR and PBR) and the data of defects' types is nominal. The chi-square results for the Seeded defects, Unseeded defects, Overall true defects and False Positive defects are presented in below Table 7. Whereas Parametric t-test is applied to find the difference between efficiency of CBR and PBR on the basis of time spent in detecting defects presented in Table 8, as experiment design has one factor (inspection technique) with two treatments (CBR and PBR), the data of time spent in detecting defects is ratio and also distribution of data is normal.

Table 7. Overall Comparison of CBR and PBR effectiveness for Companies E and T for developed Checklist

		Seeded Defects			Unseeded True defects			Overall True Defects			False positive Defects						
Defects' types	Chi-Square	Company E	Company T	Better inspection Technique	Company E	Company T	Better inspection Technique	Company E	Company T	Better inspection Technique	Company E	Company T	Better inspection Technique				
	Ambiguity	χ^2 df p	1.27 1 .260		.077 1 .781	PBR		2.82 1 .093	19.2 1 .000		PBR	3.95 1 .047		20.35 1 .002	PBR	6.14 1 .013	6.76 1 .009
Incorrectness	χ^2 df p	9.32 1 .002	.111 1 .739	PBR	9.99 1 .002		9.33 1 .002	PBR	18.7 1 .000	4.47 1 .034		PBR	70.77 1 .000	6.87 1 .009		CBR	
Inconsistency	χ^2 df p	10.8 1 .001	1.17 6 .278		PBR		15.15 1 .000		13.3 1 .000	PBR			25.7 1 .000	10.06 1 .002			PBR
Incompleteness	χ^2 df p	.000 1 1	2.41 1 .121			PBR	44.91 1 .000		15.9 1 .000		PBR		33.3 1 .000	12.86 1 .000	PBR		









Chi-square test values for the seeded defects of both companies E and T to compare the effectiveness of CBR and PBR are presented in Table 7. The Null hypothesis is rejected at (0.05) level of significance for two defects' types of Incorrectness and Inconsistency ($p=.002$ and $p=.001$) of company E because the calculated values of chi-square are less than 0.05. Therefore, it is concluded that there is significant

difference in the effectiveness of CBR and PBR for the said defects' types. However, for Ambiguity and Incompleteness defects' types of company E and for all defects' types of company T, we accept Null hypothesis at (0.05) level of significance as the computed value of p are greater than 0.05 ($p > 0.05$). Therefore, it is concluded that there is no difference in PBR and CBR. The possible reason may be that Company T is at CMMI level 5, so the inspectors from these companies are more experienced than those of Company E. The inspector has also detected Unseeded True Defects other than seeded defects for all defects types, we reject the Null Hypothesis at 0.05 level of significance as the ($p < 0.05$) except for ambiguity defect type.

Thus for Unseeded True defects, there is difference in effectiveness of CBR and PBR; PBR proved to be more effective for these defects. To compare effectiveness of CBR and PBR for the Overall true defects for both companies E and T, chi-square results were computed that rejected the null hypothesis at (0.05) level of significance for all defects' types because the calculated values of p are less than 0.05. Therefore, it is concluded that there is a significant difference in the effectiveness (true defects) of inspection techniques for all defects' types. PBR detected more defects of each particular defect's type in comparison to CBR. The reason for this difference may be that during execution of active inspection technique PBR, the inspector gets better understanding of defects by developing artifacts that is a part of its inspection procedure and also by inspecting it with different perspective resulting in more defects.

The chi-square values of false positive defects show significant difference in PBR and CBR for every defects' types at (0.05) level of significance as ($p < 0.05$). It can be seen that PBR identified more false positive defects as compared to CBR, which is not a supportive argument for an inspection technique. The possible reason is that as a part of procedure, two inspectors with different perspectives were inspecting the UCSs. Thus, the inspection by two inspectors of PBR resulted in reporting of more false positive defects as compared to CBR with one inspector.

Table 8. Comparison of CBR and PBR Efficiency for Companies E and T

Defects' Types	Inspection Techniques		T-test(p-values)	
	PBR	CBR	Company E	Company T
Ambiguity			8.06E-05	6.12E-14
Incorrectness			1.24E-05	1.02E-14
Inconsistency			1.13E-06	2.78E-12
Incompleteness			0.0010	1.59E-16

The time difference between CBR and PBR to detect a particular type of defect is described in above mentioned TABLE 8. On the basis of t-test results, the null hypothesis is rejected at (0.05) level of significance for all defects' types because the computed values of t-test are less than 0.05. Therefore, it is concluded that CBR is more efficient than PBR for detection of all defects' types. The reason for this is that each inspector of PBR (user, designer) spent more time in comparison of CBR inspector to detect defects because they have to develop model as a part of inspection procedure. The result of this study confirms the past reported studies' results [16][23] where the CBR proved to be more efficient than PBR.

5 Conclusion and Contribution

Based on the Results and Analysis presented in Section 4, we conclude that PBR detected more true defects against seeded as well as unseeded defects. However, for applying PBR, the effort (person hours) as a procedural demand, is more than CBR. CBR is better than PBR for the aspects of efficiency and false positive ratio. We also calculated the ratio of true and false defects to find comparison of actual effectiveness for both inspection techniques. It proved that CBR found more true defects against false positive defects. Therefore CBR is recommended to be used in industry because it is more efficient, reports less false positive defects, has a higher true against false ratio and also needs fewer resources (person hours) in comparison to PBR. The outcome provides the guidelines below to practitioners. Practitioners can now apply our novel developed checklist to detect the defects during early phase of software development.

- For medium to small companies, where resources and affordability are limited, CBR is recommended for inspection. However, it is suggested that in companies where the resources are sufficient, PBR is a better choice for inspection as it finds more true defects (seeded and unseeded) but it consumes more resources in terms of effort (person hours)
- Another significant finding for the industry is that the choice of both inspection techniques is not affected by defects' types since they are not sensitive towards defects' type.
- Results also guides that the practitioners must observe care while authoring and inspecting the UCSs for the Incompleteness and Ambiguity defects' types because their proportion is very high as compared to other defects' types.
- In addition, the result of both companies E and T indicates that the PBR inspectors reported less overlapping as compared to the CBR inspectors.

6 Future Work

Due to limitation of resources like practitioners' time, the scope of this research was only restricted to the comparison of two inspection techniques CBR and PBR. In the future, the other inspection techniques can be compared for guiding the software industry in finding the most effective and efficient inspection technique based on defects type in the UCS. As it was found that both PBR and CBR are not sensitive to defects types, it is to be checked in future if other inspection techniques are sensitive to defects' types or not. PBR can be analyzed by increasing the number of perspectives to find the increase in number of defects. This will guide the industry about the optimal level of perspective for optimal coverage of defects.

Due to prevalence of Agile software development ASD, the new tailored checklist on the basis of defects' types can be developed for user stories and the same experiment to compare the effectiveness and efficiency of PBR and CBR may be conducted to assist the industry to provide maximum defects detection at early phase of ASD in order to ensure high quality of software products in a cost effective way.

Acknowledgment. Thanks to Barbara Peach, for her guidance on the checklist. We also thank all the industry professionals who gave their precious time to complete this experiment.

References

1. Pressman, R.S.: Software Engineering. system **5**, 47B (1999)
2. Trochim, W.M., Donnelly, J.P.: Research Methods Knowledge Base. Atomic Dog /Cengage Learning, Mason, OH (2008)
3. Rolland, C., Achour, C.B.: Guiding the construction of textual use case specifications. *Data Knowl. Eng.* **25**(1), 125–160 (1998)
4. Boehm, B., Basili, V.R.: Software Defect Reduction Top 10 List, *Softw. Eng. Barry W Boehms Lifetime Contrib. Softw. Dev. Manag. Res.*, 69(1), 75 (2007)
5. Aurum, A., Petersson, H., Wohlin, C.: State-of-the-art: software inspections after 25 years. *Softw. Test. Verification Reliab.* **12**(3), 133–154 (2002)
6. Fagan, M.E.: Advances in software inspections. In: *Pioneers and Their Contributions to Software Engineering*, Springer, pp. 335–360 (2001)
7. Gilb, T., Graham, D., Finzi, S.: *Software inspection*. Addison-Wesley Longman Publishing Co., Inc. (1993)
8. Nuseibeh, B., Easterbrook, S.: Requirements engineering: a roadmap. In: *Proceedings of the Conference on the Future of Software Engineering*, pp. 35–46 (2000)
9. Neill, C.J., Laplante, P.A.: Requirements engineering: the state of the practice. *Softw. IEEE* **20**(6), 40–45 (2003)
10. Jacobson, I.: *Object-oriented software engineering: a use case driven approach*. Pearson Education India (1992)
11. Rumbaugh, J., Jacobson, I., Booch, G.: *The Unified Modeling Language Reference Manual 1998*, Harlow Addison-Wesley (1990)
12. Lange, C.F., Chaudron, M.R.: Effects of defects in UML models: an experimental investigation. In: *Proceedings of the 28th international conference on Software engineering*, pp. 401–411 (2006)
13. Somé, S.S.: Supporting use case based requirements engineering. *Inf. Softw. Technol.* **48**(1), 43–58 (2006)
14. I.C.S.S.E.S. Committee, I. Electronics Engineers, and I.-S. S. Board: IEEE recommended practice for software requirements specifications: approved June 25 1998, vol. 830. IEEE (1998)
15. Basili, V.R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sørungård, S., Zelkowitz, M.V.: The empirical investigation of perspective-based reading. *Empir. Softw. Eng.* **1**(2), 133–164 (1996)
16. Ciolkowski, M.: S.E.G.S. mit Generischen, Empirical investigation of perspective-based reading: A replicated experiment. *Fraunhofer-IESE* (1997)
17. Berling, T., Runeson, P.: Evaluation of a perspective based review method applied in an industrial setting. *IEE Proc.-Softw.*, 150(3), 177–184 (2003)
18. Cockburn, A.: *Writing effective use cases*, vol. 1. Addison-Wesley Boston (2001)
19. Lanubile, F., Mallardo, T., Calefato, F., Denger, C., Ciolkowski, M.: Assessing the impact of active guidance for defect detection: a replicated experiment, In: *Proceedings of the 10th International Symposium on Software Metrics 2004*, pp. 269–278 (2004)
20. Lanubile, F., Visaggio, G.: Evaluating defect detection techniques for software requirements inspections, *ISERN Rep. No 00-08* (2000)
21. Halling, M., Biffel, S., Grechenig, T., Kohle, M.: Using reading techniques to focus inspection performance, In: *Proceedings of the 27th Euromicro Conference 2001*, pp. 248–257 (2001)

22. Anda, B., Hansen, K., Sand, G.: An investigation of use case quality in a large safety-critical software development project. *Inf. Softw. Technol.* **51**(12), 1699–1711 (2009)
23. Denger, C., Ciolkowski, M., Lanubile, F.: Does active guidance improve software inspections? A preliminary empirical study. In: *Proceedings of the IASTED International Conference on Software Engineering (SE)* (2004)
24. Chillarege, R., Bhandari, I.S., Chaar, J.K., Halliday, M.J., Moebus, D.S., Ray, B.K., Wong, M.-Y.: Orthogonal defect classification—a concept for in-process measurements. *IEEE Trans. Softw. Eng.* **18**(11), 943–956 (1992)
25. Krogstie, J.: A semiotic approach to quality in requirements specifications. In: *Organizational Semiotics*, Springer, pp. 231–249 (2002)
26. Adolph, U.S., Bramble, P., Cockburn, A., Pols, A.: *Patterns for effective use cases*. Addison-Wesley Professional (2002)
27. Achour, C.B., Rolland, C., Maiden, N.A.M., Souveyet, C.: Guiding use case authoring: Results of an empirical study. In: *Proceedings IEEE International Symposium on Requirements Engineering 1999*, pp. 36–43 (1999)
28. Denger, C., Paech, B.: An Integrated Quality Assurance Approach for Use Case Based Requirements. In: *Modellierung*, pp. 59–74 (2004)
29. Sandahl, K., Blomkvist, O., Karlsson, J., Krysanter, C., Lindvall, M., Ohlsson, N.: An extended replication of an experiment for assessing methods for software requirements inspections. *Empir. Softw. Eng.* **3**(4), 327–354 (1998)
30. Shull, F., Rus, I., Basili, V.: How perspective-based reading can improve requirements inspections. *Computer* **33**(7), 73–79 (2000)
31. Halling, M., Biffel, S., Grechenig, T., Kohle, M.: Using reading techniques to focus inspection performance. In: *Proceedings of the 27th Euromicro Conference 2001*, pp. 248–257 (2001)
32. Anda, B., Sjøberg, D.I.: Towards an inspection technique for use case models. In: *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, pp. 127–134 (2002)
33. Cox, K., Aurum, A., Jeffery, R.: An experiment in inspecting the quality of use case descriptions. *J. Res. Pract. Inf. Technol.* **36**(4), 211–229 (2004)
34. Phalp, K.T., Vincent, J., Cox, K.: Assessing the quality of use case descriptions. *Softw. Qual. J.* **15**(1), 69–97 (2007)
35. Brykczynski, B.: A survey of software inspection checklists. *ACM SIGSOFT Softw. Eng. Notes* **24**(1), 82 (1999)
36. Laitenberger, O., Vegas, S., Ciolkowskoi, M.: The state of the practice of review and inspection technologies in germany, Tech Report Number: ViSEK/011/E (2002)
37. Thelin, T., Runeson, P., Wohlin, C.: An experimental comparison of usage-based and checklist-based reading. *IEEE Trans. Softw. Eng.* **29**(8), 687–704 (2003a)
38. Thelin, T., Runeson, P., Wohlin, C.: Prioritized use cases as a vehicle for software inspections. *Softw. IEEE* **20**(4), 30–33 (2003b)
39. Fogelström, N.D., Gorschek, T.: Test-case driven versus checklist-based inspections of software requirements—an experimental evaluation. In: *WER07-Workshop em Engenharia de Requisitos*, pp. 116–126. Toronto, Canada (2007)
40. Belgamo, A., Fabbri, S., Maldonado, J.C.: TUCCA: improving the effectiveness of use case construction and requirement analysis. In: *International Symposium on Empirical Software Engineering 2005*, p. 10 (2005)
41. Bernardes, B., Genero, M., Duran, A., Toro, M.: A controlled experiment for evaluating a metric-based reading technique for requirements inspection. In: *Proceedings of the 10th International Symposium on Software Metrics 2004*, pp. 257–268 (2004)