Investigating the Impact of Active Guidance on Design Inspection

Dietmar Winkler, Stefan Biffl, and Bettina Thurnher

Vienna University of Technology, Institut of Software Technology,
Karlsplatz 13, A-1040 Vienna, Austria
{Dietmar.Winkler, Stefan.Biffl,
Bettina.Thurnher}@qse.ifs.tuwien.ac.at

Abstract. Software inspection helps to improve the quality of software products early in the development process. For design inspection recent research showed that usage-based reading of documents is more effective and efficient than traditional checklists. Usage-based reading guides actively the inspector with pre-sorted use cases, while traditional checklists let the inspector figure out how best to proceed. This paper investigates the impact of active guidance on an inspection process: We introduced checklists that give the inspector a process to follow, which should be as flexible as traditional checklists but more efficient. We compared the performance of this approach in a controlled experiment in an academic environment with traditional checklist and usage-based reading. Main results of the investigation are (a) checklists with active guidance are significantly more efficient than traditional checklists for finding major defects and (b) usage-based reading is more effective and efficient than both types of checklists. These results suggest that active guidance improves the efficiency of inspectors while the upfront investment into usage-based reading pays off during inspection.

Keywords: inspection process improvement, reading techniques, software product improvement, empirical software engineering, active guidance.

1 Introduction

Software inspection is a current approach for quality improvement of software products in industrial environment, since Fagan introduced it in 1976 [5]. Inspection is a defect detection technique to reduce defects in software artifacts and to improve software product quality [4][17]. The inspection method is classified as a static verification and validation technique, which doesn't need executable software. Therefore, inspection approaches are applicable to written text documents, e.g. design documents, as well.

Inspection in our context concentrates on defect detection in early stages of software development, i.e. in design documents. The early elimination of defects leads to a higher level of product quality, due to a lower number of remaining defects and, as a consequence, to a reduction of required resources (e.g. budget, time, etc.). Therefore, inspection is one important approach for software product improvement.

F. Bomarius and S. Komi-Sirviö (Eds.): PROFES 2005, LNCS 3547, pp. 458 – 473, 2005.

In order to find defects, inspectors have to traverse the document under inspection. Reading is a key activity in defect detection processes to (1) understand the document under inspection and (2) compare the inspection artifact to a set of expectations regarding content, structure and product quality. This comparison and recognition helps to spot defects. Because of this key activity, several reading techniques (RTs) have been developed to improve inspection process quality.

In general, inspectors have to learn reading and to analyze the software artifacts applying reading techniques. Systematic reading techniques consist of series of steps that help inspectors to understand particular aspects of a document with active reading work and to use this information for defect detection. Important characteristics of RTs are [13]: usability (simplicity to follow predefined guidelines) [16], applicability to different document notation and application domains, repeatability of inspection results, document coverage, and target defects.

Therefore, a well-designed RT must achieve those requirements and uses available knowledge on the structure of a document to provide guidance through the most important parts of the document. RTs support readers while inspecting the document in an active or passive way. Readers using a passive approach inspect the artifact regarding a number of steps sequentially (e.g. given checklist items). Active guidance includes a detailed inspection process (*how to perform an inspection*) and a separation of perception (*what to inspect*) [3][14].

Empirical studies in academic environment use checklists (CBR), scenarios (SBR), use cases (UBR), or perspectives (PBR) [1] [12] to focus on different types of defects, e.g. defect severity classes, document locations, impact of individual defects, etc. to investigate the benefits of the individual RT approaches. Examples for empirical studies are: checklist-based RT (CBR) [15][27] and usage-based RT (UBR) [23][24][25][26][27]. Gilb et al. presents an overview and comparison of CBR and SBR [7]. Families of empirical studies must be performed to provide generalization of empirical findings, e.g. [6][11][18][19][21][29].

Checklist-based reading (CBR) approaches use sequentially predefined items, which lead the inspector through the document under inspection. Inspectors have to traverse the document several times for a complete coverage of the specification document and the checklist. The new checklist-based RT variant (CBR-tc) uses a tailored checklist to provide an active guidance to the inspector. Inspectors have to analyze requirements and system function and prioritize them according to their knowledge of the application domain. This proceeding is included in the inspection process. Usage-based approaches (UBR) use expert prioritized use cases and scenarios for defect detection. Inspectors follow them and traverse the document in order to find defects. The main advantage of UBR is the application of expert prioritized use cases to find the most important defects and the support of active guidance.

This paper presents the results of a large-scale experiment in academic environment at Vienna University of Technology [28]. The empirical study cover a checklist-based reading technique (CBR-gc) using a generic checklist, and a usage-based reading technique approach (UBR) with use cases and a new CBR variant, a tailored checklist (CBR-tc). The aim of this paper is the investigation of the impact of active guidance on the number of defects found at different severity classes (crucial and major defects).

The remainder of this paper is structured as follows. Section 2 presents the reading techniques compared in our experiment. Section 3 describes research questions and Section 4 outlines the empirical study. Section 5 presents the study results. Section 6 discusses the results. Section 7 concludes and outlines directions for future research.

2 Checklists and Usage-Based Reading Techniques

A well-design reading technique is a structured approach to support inspectors in defect detection processes. We cover two classes of RTs, (a) the checklist-based (CBR-gc) approach using a generic checklist and (b) the usage-based RT (UBR) approach. Furthermore, we introduce a new checklist-based RT variant (CBR-tc). This new variant includes tailored checklist items, which support the reader by active guidance through the specification document.

A *checklist-based RT (CBR-gc)* is a method, which typically consists of a set of questions for general purposes, usually independent from a specific notation [7]. Inspectors traverse the document according to every checklist item several times sequentially and report defects found during inspection. CBR-gc offers little guidance on defect detection processes. Therefore, the results depend strongly on the individual inspectors and suffer from variability according to inspector capability.

RTs including active guidance aim to support inspectors during reading processes and improve disadvantages of CBR-gc. Active guidance helps inspectors to traverse the document under inspection, providing guidelines, how to perform inspection and what to inspect, e.g. according to defect types, etc. [14]. We use two different approaches, a tailored checklist (CBR-tc) and usage-based RT approach (UBR).

CBR-tc is a modified checklist providing active guidance to the inspector. A more application specific checklist lead the inspector through the inspection process, performing the following major steps:

1. Analysis of requirements and system function within the requirements document.
2. Investigation and prioritization of correlations between requirements and system functions according to the experience of the inspector.
3. Tracking of requirements and functions according to their importance through the document under inspection.
4. Report differences and defects.
5. Select the next most important requirement and proceed until the time is up or the inspector has covered all requirements and system functions.

CBR-tc leads the inspector through the specification document in an active way regarding application domain and focus on important defects, due to a prioritization task at the beginning of individual inspection. Nevertheless, it depends on the knowledge of the inspector to perform a correct ranking.

Usage-based reading (UBR) focuses on prioritized use cases and support active guidance, due to given guidelines and scenario representations [24]. Use cases represent the user view and spot defects, which are normally hard to find. Inspectors read prioritized use cases sequentially, apply them to the design specification and report defects. Because of this focus, UBR improve the understanding of inspectors and support them in finding more severe defects.

Most experiments focus on defect detection for individual inspectors and teams, concerning time variables and performance measures [2][23]. But there is very little concern for the effort of inspection preparation, i.e. the time interval before inspection within the inspection environment. Because inspection managers prefer using existing inspection material, there is a very low hurdle applying a checklist-based reading technique for individual inspection. More sophisticated reading techniques such as UBR and SBR need to be tailored upfront to the document under inspection.

Thus, there are several criteria that need to be considered when deploying a defect detection technique in practice: Effort for individual inspectors, effectiveness and efficiency of defect detection when applying a reading technique and upfront investment due to tailoring of documents.

Concerning CBR-gc the preparation phase requires very low effort because of the usage of a generic checklist, i.e. very less or no additional effort to adapt checklists to different application domains.

Experts almost need more effort to the preparation of CBR-tc reading technique approaches, because there is a context to the application domain. Experts also have to pay attention to the requirements document to provide active guidance to inspectors. Additional effort is necessary during individual inspection for analysis and prioritization of requirements and system functions.

Obviously, UBR reading technique approaches require most pre-work of experts depending on given artifacts according to their notation. Textual requirements notation requires the translation of requirements into use cases and a prioritization of use cases afterwards. In case of given use cases, i.e. the notation of requirements contains scenarios and use cases, experts have to prioritize them according to their expert knowledge.

This paper represents the empirical results of our investigations of active guidance according to effectiveness, efficiency, and effort, also regarding defect severity classes and reading technique approaches.

3 Research Questions

The main focus of this paper is the investigation of the impact of active guidance on design inspections with respect to time variables and performance measures. We use the results of an external replication of the UBR experiment as described in [27] and [28]. In addition to CBR and UBR inspection we introduce a slightly adjusted version of CBR, namely CBR-tc (CBR using tailoring approaches to prioritize requirements and corresponding system functions).

The UBR reading technique approach uses guidelines how to proceed during inspection and a predefined prioritized list of use cases [20]. The ranking of use cases is important to focus on crucial defects and to guide the inspector through the design specification. Nevertheless, this ranking requires additional effort by expert before inspection started. Further information on the importance of use-case ranking was discussed in a previous paper [28].

CBR-gc inspectors apply a generic checklist for multiple purposes to find defects in the software artifact. Inspectors have to traverse the document several times

according to every checklist item to achieve full document coverage. They do not apply any use cases or scenarios at all.

In this paper we introduce a slightly adjusted checklist to provide active guidance for checklist based reading. The initial checklist consists of a strict proceeding to identify requirements, system functions and their correlation. Additionally, the inspectors have to prioritize the requirements according to their subjective importance according to their own knowledge of the application domain. Therefore, inspectors need additional effort for this task during inspection proceeding. The inspectors use this prioritized requirements to traverse the specification document. This approach enables a deeper understanding of the specification document and the system requirements as well as system functions.

The application of an individual reading technique approach requires a different amount of effort for experiment preparation concerning RT specific tasks:

- *CBR-gc* uses checklists [15], which were developed before the experiment for generic purposes. The application of CBR-gc does not require any additional effort.
- *CBR-tc* also uses checklists, but leave the tailoring process to the inspection (as a part of the inspection process) according to the individual knowledge of the inspectors. Inspectors have to identify, classify and prioritize requirements, system functions and their dependability.
- *UBR* provide use cases and scenarios [26], which are unique to the application domain and the software document. Experts find use cases, and prioritize them according to the application domain to guide the inspector active through the inspection process.

The purpose of this paper is the investigation of *inspection effort*, *effectiveness* and *efficiency* of defect detection regarding the influence of active guidance. Inspection effort is importance in industrial environment for acceptability and applicability of inspections in general regarding deadlines and cost. Inspection effectiveness is the number of found defects in relation to all defects within the software document. One of the major goals of software inspection is the reduction of defects in software products to improve software quality [4][18]. Because different RTs focus on different defect classes (e.g. defect severity), it is interesting to point out the best applicable RT. Efficiency joins effort and effectiveness and depicts the number of found defects in a defined time interval (i.e. defects per hour).

3.1 Variables

The experiment setup contains two different types of variables [27] [28], (a) independent and (b) dependent variables. We use the reading technique used as *independent* variable, i.e. *CBR-gc*, *CBR-tc*, and *UBR*. CBR-gc applies a generic checklist for general purposes (predefined within the experiment environment). CBR-tc provides a checklist including the procedure how to proceed during inspection, i.e. finding and prioritizing requirements, classifying system functions and locating associations, etc.). UBR implies a predefined order of use cases (including expert know how) for inspection. We controlled the influence of inspector experience by randomly assigning reading techniques to inspectors.

We use *dependent variables* to capture the performance of the individual inspection procedures regarding different reading technique approaches. Following standard practice in empirical studies we focus on time variables and performance measures. Therefore, we capture *inspection time* in minutes (preparation and inspection time) and the *number of defects* found during inspection. Regarding performance measures we investigate the influence of active guidance to *inspection effectiveness* and *efficiency* (the number of defects per hour) according to the reading techniques applied. Furthermore, we pay special attention to defect severity classes. Experts ranked the defects according to three defect severity classes: critical defects *class A*), major defects (*class B*), and minor defects (*class C*). For evaluation purposes, we focus on finding important defects in the classes A or A+B (including class A or class B defects).

3.2 Hypotheses

In the experiment we observe the performance of inspectors who apply one of three reading techniques: CBR-gc, CBR-tc, or UBR. As main goal of this paper we investigate research hypothesis regarding inspection effort, effectiveness and efficiency according to the reading technique approach used and the influence of active guidance. In more detail we evaluate the following hypotheses:

Inspection Effort: Inspection effort includes inspection preparation and inspection duration as part of the individual inspection process. We do not cover effort of experts as part of the overall inspection preparation, i.e. ranking of use cases, generation of guidelines, checklists, etc.

CBR-gc inspectors have to traverse the document several times according to every checklist item to achieve full document coverage. Therefore, we expect the highest overall inspection effort. CBR-tc inspectors use active guidance but have to perform an additional tailoring session to analyze requirements and system function including prioritization. This additional session increase preparation time.

H1: Effort (CBR-tc) < Effort (CBR-gc): In summary we expect a lower effort through active guidance (CBR-tc), because the inspectors benefit from better understanding of the application domain and document under inspection.

Effectiveness: In this paper we consider effectiveness as the number of defects found at three different severity classes (critical, major, and minor) regarding the overall number of seeded defects of every severity class.

H21: Effectiveness (UBR) > Effectiveness (CBR-gc): UBR inspectors benefit from active guidance using prioritized use cases and scenario. Therefore, we expect a higher effectiveness in relation to CBR-gc inspectors, who have to read the document several times sequentially. Additionally, UBR readers focus on important defects due to an expert ranked approach of prioritization. Experts are familiar with the design specification including background knowledge of seeded defects.

H22: Effectiveness (UBR) > Effectiveness (CBR-tc): The argument is similar to H21 for UBR inspectors. CBR-tc inspectors perform tailoring and prioritization of requirements and system functions without background knowledge. We expect an advantage of expert ranking effects for important defects according to inspection effectiveness regarding UBR approaches.

H23: Effectiveness (CBR-tc) > Effectiveness (CBR-gc): The argument is similar to H22 under the assumption that the analysis of requirements and system functions during the preparation phase improve defect detection and effectiveness. CBR-gc inspectors traverse the design specification several times using a generic checklist without special attention on the application domain.

Efficiency: Efficiency combines effort and effectiveness and is defined as the number of detected defects per time interval, i.e. per hour. To investigate efficiency we summarize overall inspection effort (including preparation and inspection time) and total number of matched (seeded) defects.

H31: Efficiency (UBR) > Efficiency (CBR-gc): We expect a higher efficiency for UBR inspectors because of active guidance of inspectors using use cases and scenarios. CBR-gc inspectors traverse the specification document several times without active support by the reading technique. Therefore, UBR inspectors will find more defects and need less effort.

H32: Efficiency (UBR) > Efficiency (CBR-tc): The argument is similar to H31. Additionally CBR-tc is classified as a method with active guidance with focus on prioritized requirements and system functions. Because of an additional effort for this analysis and prioritization task, inspection effort will increase. Obviously, efficiency will decrease.

H33: Efficiency (CBR-tc) > Efficiency (CBR-gc): Active guidance and prioritized requirements and system functions improve efficiency of CBR-tc in contrast to CBR-gc, where the inspectors have to traverse the document under inspection several times sequentially. The expected additional effort for prioritization compensates CBR-gc approaches.

4 Experiment Description

The experiment was conducted at Vienna University of Technology in academic environment in December 2003. This study is a replicated experiment as described in [27] and [23] involving 127 inspection participants. We leave the details to the original developers of UBR and the experiment environment. In this section we will briefly describe key aspects of the experiment and point out the basic experiment proceeding, used artifacts, and involved subjects.

4.1 Experiment Proceeding

The empirical study consists of three major phases, a preparation phase, the inspection execution, and the evaluation phase:

Experiment preparation: Experts had to prepare inspection artifacts, e.g. requirements document, design specification, guidelines for reading techniques, and questionnaires. Using 3 different RT approaches, corresponding tasks had to be performed by experts before conducting the inspection within the experiment environment. We do not cover organizational aspects in this paper.

Additional to the preparation of guidelines and questionnaires, this preparation phase included:

- Conductance and prioritization of use cases by experts for UBR.
- Preparation of checklist items for tailoring tasks to support inspectors for CBR-tc. Inspectors have to perform analysis and prioritization of requirements and system functions during the inspection process using those guidelines.
- Preparation of a generic checklist for CBR-gc. This approach did not require much additional effort because of the generic structure of the checklist (re-use of this checklist).

Inspection execution: The experiment execution included three steps: a *training and preparation* session, *individual inspection*, and *data submission*. Inspectors got an overview of the inspections process and learned basics about inspection and different reading technique approaches during a short training session. Individual inspection included (a) the inspection preparation, i.e. reading the document under inspection, analysis and prioritization of requirements and system functions (CBR-tc) etc., (b) the defect detection and reporting process, and (c) data submission. During data submission step, all inspectors had to log their candidate defects electronically for quality assurance purposes. This data submission task also included questionnaire results and time stamps for processes and defects findings.

Data evaluation: After individual inspection, experts mapped candidate defects, i.e. defects noted by individual inspectors, to reference defects, i.e. real defects seeded by experts. The data, derived from the database were checked for consistency and correctness. We excluded the data from subjects, who delivered inconsistent data or did not follow the experiment process properly.

Also note, that candidate defects that refer to one true seeded defects were counted only once at the first clock time of defect detection.

For statistical evaluation we use the Mann-Whitney test to investigate *effort* and *efficiency* and a chi square test to test *effectiveness*. The significance level of rejecting the hypotheses is set to 0.05 for all tests.

4.2 Software Artifacts

The artifacts describe a taxi management system and include (a) a textual requirements definition, (b) a design document, (c) use case documents, (d) several guidelines for reading technique approaches, and (d) questionnaires for capturing experience and feedback information.

- The *textual requirements document* was used as a reference document and was assumed to be accurate.
- The *design document* describes an overview of the software modules and their individual context. The document includes the internal representation (between two or more modules) as well as the external representation (between the user and the taxi management system). The design document consists of 9 pages, including 2500 words, 2 sequence charts, and 39 known defects, i.e. seeded defects.

- Defect reports were linked to 39 *seeded defects* at different severity classes (13 crucial (class A), 15 major (class B), and 11 minor (class C) defects) spread all over the document, which were seeded before inspection by experts during experiment preparation. *Class A* means a heavy adverse affect on functionality which will appear very often (highest risk). *Class B* contains important rarely used defects or unimportant often used defects (medium risk). *Class C* defects are neither crucial nor very important (low risk).
- The *use case document* contained 24 use cases from user view-point in task notation. Experts prioritized those use cases according to their importance for the usage-based reading technique application.
- RT specific *guidelines* support the individual inspectors and lead them through the inspection process while applying the assigned reading technique.
- We use two types of questionnaires to achieve (a) background information of the individual inspectors (experience questionnaire) at the beginning, and (b) feedback on the RT applied (feedback questionnaire) at the end of inspection.

4.3 Subjects

127 software engineering students were taking part in the experiment using CBR-gc, CBR-tc, or UBR reading technique approaches. The experiment was fully integrated into the course as a practical part to practice software inspection and learn key aspects of software product improvement in early stages of software development. We controlled the influence of inspection capability by randomly assigning reading techniques to inspectors.

To our knowledge this is the largest empirical study comparing UBR and CBR RT approaches. Similar experiments involved a total number 12 [3], 23 [27], 62 [23], and 42 participants [14].

Table 1. Number of inspectors by RT

RT	Number of Insp.	Percentage
CBR-gc	24	19%
CBR-tc	48	38%
UBR	55	43%
Total	127	100%

Table 1 displays the distribution of inspectors with respect to reading technique roles. We used the CBR-gc reading technique as control group involving 19% of inspectors.

5 Experiment Results

In this section we present the empirical results of the study. We pay attention to inspection effort, effectiveness (number of defects found), and efficiency (defects found per hour).

5.1 Effort

Inspection effort includes preparation time and inspection duration. The inspectors logged the clock time for each individual task; reading the specification and design document, tailoring and prioritizing of requirements – summarized in preparation time and inspection time. Table 2 displays the mean and standard deviation for the three reading techniques.

Table 2. Individual preparation and inspection time in minutes

		CBR-gc	CBR-tc	UBR
Mean	Preparation	43.3	46.0	42.8
	Inspection	120.3	110.0	117.7
	Total	163.5	155.9	160.6
Std.Dev	Preparation	15.7	19.0	22.5
	Inspection	27.9	30.8	28.1
	Total	25.1	34.6	29.5

All three RTs have a similar total effort on average. Concerning preparation time, CBR-tc inspectors need somewhat longer because of an additional tailoring and prioritization task in contrast to CBR-gc and UBR inspectors, but the subsequent inspection duration is shorter. In summary, CBR-tc inspectors need somewhat less effort for overall inspection.

Concerning our research hypothesis we observe that there is no significant difference of inspection effort between all three RT approaches.

5.2 Effectiveness

In the context of this paper we define effectiveness as the number of seeded defects found by an individual inspector in relation to the overall number of seeded defects within the design document. The experiment setup contains 39 seeded defects, summarized by 13 crucial, 15 major and 11 minor defects. To figure out the benefits of the individual RTs we investigate crucial defects (class A), important defects (class A and class B), and all defects found by inspectors.

Table 3 summarized mean values and standard deviations of effectiveness according to defect classes and reading techniques.

Table 3. Effectiveness by defect class and reading technique

	Defect	CBR-gc	CBR-tc	UBR
Mean	Class A	24.4	29,2	35.8
	Class A+B	26.3	28,1	33.1
	All defects	24.9	25,2	29.8
Std.Dev	Class A	19.2	13,5	14.4
	Class A+B	14.4	10,0	11.0
	All defects	10.9	8,8	11.9

On average there is a notable difference between the effectiveness of CBR-gc, CBR-tc, and UBR, that is consistent for all classes of defects investigated. UBR inspectors found more defects than CBR-tc and CBR-gc inspectors for every defect class. Also CBR-tc dominates CBR-gc for every subset of defects. The performance advantage of UBR is greatest for defects belonging to class A. Note the smallest standard deviation at CBR-tc with respect to all defects.

Fig. 1 depicts effectiveness of three reading technique with respect to defect severity classes.

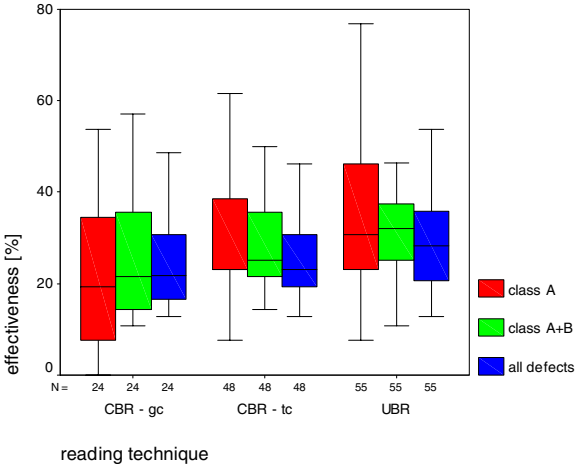

Fig. 1. Effectiveness of three reading techniques

Table 4 shows the results of our investigation according to significance values. There is a notable difference concerning all defect severity classes and all couples of reading techniques investigated.

Table 4. p-values for effectiveness according to defect classes and RT

	Class A	**Class A+B**	**All Defects**
CBR-gc / UBR	<0,001(S)	0,021(S)	0,018 (S)
CBR-tc / UBR	<0,001(S)	<0,001(S)	0,029(S)
CBR-gc / CBR-tc	<0,001(S)	0,022(S)	0,043(S)

Therefore, we can actually confirm our research hypotheses: UBR with expert ranking significantly outperforms all other reading techniques, due to active guidance including the prioritized use cases applied. Thus, the performance advantage of UBR is greatest for all defects. CBR-tc also dominates CBR-gc for every subset of defect severity classes, because of the impact of active guidance for CBR-tc inspection.

Additionally we investigated the ratio of *false positives*, i.e. the ratio of candidate defects that do not match to true defects regarding all defects found be individual

inspectors. Because the inspectors did not know the number of real defects, they reported all possible defects, which were matched to real defects by experts. We did not recognize significant differences concerning false positives at noted defects and false positives. We did register significant differences with respect to non reference defects. CBR-gc inspectors received the highest number of wrong defects because of a non-active guidance through the inspection process using a generic checklist.

5.3 Efficiency

We combine the measures of effort and effectiveness to investigate inspection efficiency, i.e. defect detection rate per hour. Effort is concerned as the total amount of preparation and inspection time in minutes. We also use the total number of matched defects to derive efficiency.

Table 5. Inspection efficiency (number of defects found per hour)

	Defect	CBR-gc	CBR-tc	UBR
Mean	Class A	1.1	1,5	1.8
	Class A+B	2.7	3,1	3.6
	All defects	3.6	3,9	4.5
Std.Dev	Class A	0.8	0,7	0.8
	Class A+B	1.4	1,2	1.3
	All defects	1.5	1,4	1.7

Table 5 displays mean values and standard deviations for three defect classes with respect to the reading technique approaches. The results show that UBR inspectors achieve the highest efficiency according to all defect severity classes. We measured the lowest efficiency for CBR-gc inspectors, and somewhat between for CBR-tc inspectors.

Table 6. P-values for efficiency according to defect severity and reading techniques

	Class A	Class A+B	All defects
CBR-gc / UBR	0,002(S)	0,023 (S)	0,053(-)
CBR-tc / UBR	0,059(-)	0,098(-)	0,079(-)
CBR-gc / CBR-tc	0,039(S)	0,166(-)	0,400(-)

Table 6 shows a summary of p-values, derived by the Mann-Whitney test, for defect classes and all three reading techniques used. We notice significant differences for crucial defects (class A) by comparing RTs using active guidance (UBR and CBR-tc) with respect to CBR-gc. Furthermore, we recognize significant differences for important (class A or class B defects) defects concerning CBR-gc and UBR but not for all defects.

One interesting finding is that there is no significant difference at efficiency at CBR-tc and UBR (both RTs with active guidance). This implies that active guidance focus the inspectors efficiently on the more important defect classes.

6 Discussion

In this section we summarize the empirical results from our experiment concerning the comparison of CBR-gc, CBR-tc, and UBR. Analyzing our empirical results we derive the following implications for this comparison:

Effort of inspection duration: Inspectors from all reading techniques required a similar overall amount of time for inspection. Although CBR-tc inspectors needed longer for preparation due to their additional classification and prioritization phase of requirements and system functions, but they used also less time for the subsequent inspection phase. They benefit from better understanding of the application domain and the document under inspection.

We used the Mann-Whitney test for statistical evaluation, but we do not register any significant differences. One possible reason might be some kind of group effect, because we did not specify any upper limit for overall inspection duration.

Effectiveness: The analysis of the results supports all hypotheses (*H21-H23*) nominally for reading technique approaches. We found significantly differences according to all defect severity classes, i.e. crucial, major, and minor defects.

The impact of active guidance due to expert ranking of use cases influences effectiveness at any class of defects. The reasons might be (a) a well performed prioritization process by experts during inspection preparation phase, (b) the active guidance of use cases and scenarios during inspection, and (c) a well-formed presentation of the application domain due to use-case notation. Therefore UBR outperforms all CBR RT approaches. One disadvantage of this approach is the effort, spent by experts during the prioritization phase, because these activities were executed in the before the inspection process started at all.

The comparison of CBR-tc and CBR-gc also shows significant differences according to inspection effectiveness for all defect severity classes. Inspection effectiveness also benefits from the additional task of analysis and prioritization of requirements and system functions, using the CBR-tc approach, due to a better understanding and structuring of the application domain and the active guidance through the inspection process.

Efficiency: Our hypotheses are confirmed as well. UBR is most efficient and CBR-gc is less efficient. Concerning all defects, there is no significant difference at any comparison of RTs.

The results also show significant differences at RTs using active guidance in comparison to CBR-gc (which does not use active guidance) according to crucial defects (class A). We assume the influence of use cases and scenarios as the main reason for those findings. Nevertheless, active guidance improves inspection efficiency in comparison to generic checklist approaches. Tailoring of requirements (CBR-tc) improve efficiency due a structured approach and a deeper understanding of

the document under inspection in comparison to a generic checklist approach. Nevertheless, use cases and scenarios approaches exceed efficiency of CBR-tc.

Analyzing the data from our experiment we conclude that UBR with expert ranking shows highest effectiveness and efficiency and therefore best overall performance.

The comparison to CBR-tc implies that the analysis, classification and prioritization of requirements and system functions have some influence on the performance of inspection. Active guidance, as provided by CBR-tc and UBR improve efficiency due to a reduction of overall inspection effort. Concerning effectiveness, the UBR reading technique approach, outperforms the tailored checklist approach and the generic checklist approach. The usage of use cases and scenarios lead to a deeper understanding of the application domain and improve effectiveness.

7 Conclusion and Further Work

Inspection is an important approach to reduce defects in software engineering artifacts. Reading techniques such as UBR can focus inspector attention on specific types of defects, i.e. crucial defects. Active guidance lead to a better understanding and a more structured approach of inspection proceeding and, therefore, improves inspection effectiveness and efficiency.

This paper presented a large-scale external experiment replication in the UBR family of experiments [2] in an academic environment. In addition to previous empirical studies, we introduced a new reading technique variant, a tailored checklist approach (CBR-tc) to investigate the impact of active guidance on inspection performance.

Main results of the study were: (a) active guidance improves inspection effectiveness and efficiency, (b) UBR expert know-how has significant effects of on defect detection rates; (c) the application of use cases and scenarios improve document understanding and inspection performance, and (d) both RTs with active guidance perform significantly better than CBR-gc.

Further work is to investigate the impact of inspector capability on defect detection: The empirical studies in this family of experiments have investigated defect detection according to different UBR and CBR approaches without regarding inspector capability in detail. Whereas we aware that inspector capability is an important factor to achieve well-performed inspection results.

References

1. Basili V.R., S. Green, O. Laitenberger, F. Lanubile, F. Shull, S. Soerumgaard, and M. Zelkowitz, “The Empirical Investigation of Perspective-Based Reading”, *Empirical Software Engineering Journal*, vol. 1, no. 2, pp. 133-164, 1996.
2. Basili V.R., F. Shull, and F. Lanubile, “Building Knowledge through Families of Experiments,” *IEEE Trans. Software Eng*., vol. 25, no. 4, pp. 456-473, July/Aug. 1999.
3. Denger C., Ciolkowsky M., Lanubile F. *Investigation the Active Guidance Factor in Reading Techniques for Defect Detection*. ISESE 2004.
4. Ebenau R.G. and S.H. Strauss, *Software Inspection Process*. McGraw-Hill, 1994.

5. Fagan M., "Design and Code Inspections To Reduce Errors In Program Development", *IBM Systems J*., vol. 15, no. 3, 1976, pp. 182-211.
6. Fusaro P., F. Lanubile, and G. Visaggio, "A Replicated Experiment to Assess Requirements Inspection Techniques," *Empirical Software Eng.: An Int'l J.,* vol. 2, no. 1, pp. 39-57, 1997.
7. Gilb T., Graham D., *Software Inspection*, Addison-Wesley, 1993.
8. ITU-T Z.100, Specification and Description Language, SDL, ITU-T Recommendation Z.100, 1993.
9. ITU-T Z.120, Message Sequence Charts, MSC, ITU-T Recommendation Z.120, 1996.
10. Jeffery, R.; Scott, L.; Has twenty-five years of empirical software engineering made a difference? *Software Engineering Conference, 2002. Ninth Asia-Pacific , 4-6 Dec*. 2002 pp. 539 -546
11. Juristo N. and A.M. Moreno, *Basics of Software Engineering Experimentation*. Kluwer Academic, 2001.
12. Laitenberger O., Atkinson C. "Generalizing Perspective-based Inspection to handle Object-Oriented Development Artifacts", *Proc. of the Int. Conf. on Software Engineering*, 1999.
13. Laitenberger O., DeBaud J.-M., "An encompassing life cycle centric survey of software inspection", *Journal of Systems and Software*, vol. 50, no. 1, 2000, pp. 5-31.
14. Lanubile F., Mallardo T., Calefato F., Denger C., Ciolkowksi M. *Assessing the Impact of Active Guidance for Defect Detection: A Replicated Experiment*. METRICS 2004.
15. Miller J., M. Wood, and M. Roper; "Further experiences with scenarios and checklists", *Empirical Software Engineering Journal,* vol. 3, no. 1, pp. 37-64, 1998.
16. Nielson, J.; "*Usability Engineering*", San Diego: Academic Press, 1993.
17. Parnas D., Lawford M., "The role of inspection in software quality assurance", *IEEE Trans. on SE, vol. 29(8), August 2003*, pp. 674-676.
18. Porter A., L. Votta, "Comparing Detection Methods for Software Requirements Inspections: a Replicated Experiment using professional subjects", *Empirical Software Engineering Journal,* vol. 3, no. 4, 1998, pp. 355-379.
19. Porter A., L. Votta, and V. Basili, "Comparing Detection Methods for Software Requirements Inspections: a Replicated Experiment", *IEEE Transactions on Software Engineering vol. 21, no. 6*, pp. 563-575, June 1995.
20. Saaty T.L., Vargas L.G.; "*Models, Methods, Concepts & Applications of the Analytic Hierarchy Process*"; Kluver Academic, 2001.
21. Sandahl K., Blomkvist O., Karlsson J., Krysander C., Lindvall M., Ohlsson N.: *An Extended Replication of an Experiment for Assessing Methods for Software Requirements Inspections*, Kluwer Academic Publishers, 1998
22. Shull F.J., "*Developing Techniques for using Software Documents: A Series of Empirical Studies*", PhD thesis, University of Maryland, College Park, www.cs.umd.edu/~fshull/pubs, 1998.
23. Thelin T., C. Andersson, P. Runeson, N. Dzamashvili-Fogelström "A Replicated Experiment of Usage-Based and Checklist-Based Reading", *Metrics 2004*.
24. Thelin T., P. Runeson, and B. Regnell, "Usage-Based Reading—An Experiment to Guide Reviewers with Use Cases," *Information and Software Technology*, vol. 43, no. 15, pp. 925-938, 2001.
25. Thelin T., P. Runeson, C. Wohlin, T. Olsson, and C. Andersson, "How Much Information Is Needed for Usage-Based Reading? —A Series of Experiments," *Proc. First Int'l Symp. Empirical Software Eng*., pp. 127-138, 2002.

26. Thelin T., Runeson P., Wohlin C., "Prioritized Use Cases as a Vehicle for Software Inspections", *IEEE Software*, vol. 20(4), July/August 2003, pp. 30-33.
27. Thelin T., Runeson P., Wohlin C., "An experimental comparison of usage-based and checklist-based reading", *IEEE Trans. on SE*, vol. 29(8), August 2003, pp. 687-704.
28. Winkler D., Halling M., Biffl St. "*Investigating the Effect of Expert Ranking of Use Cases for Design Inspections*", Euromicro 2004.
29. Wohlin C., P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering - An Introduction*, The Kluwer International Series in Software Engineering, Kluwer Academic Publishers, 2000.