USING LEARNING STYLES TO IMPROVE SOFTWARE REQUIREMENTS QUALITY: AN EMPIRICAL INVESTIGATION

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Anurag Goswami

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Program:
Software Engineering

May 2017

Fargo, North Dakota

North Dakota State University
Graduate School

Title

Using Learning Styles to Improve Software Requirements Quality: An Empirical Investigation

By

Anurag Goswami

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Gursimran Singh Walia
Chair

Dr. Kendall Nygard

Dr. Saeed Salem

Dr. Mark McCourt

Approved:

05/20/2017
Date

Dr. Kendall Nygard
Department Chair

ABSTRACT

The success of a software organization depends upon its ability to deliver a quality software product within time and budget constraints. To ensure the delivery of quality software, software inspections have proven to be an effective method that aid developers to detect and remove problems from artifacts during the early stages of software lifecycle. In spite of the reported benefits of inspection, the effectiveness of the inspection process is highly dependent on the varying ability of individual inspectors. Software engineering research focused at understanding the factors (e.g., education level, experience) that can positively impact the individual's and team inspection effectiveness have met with limited success. This dissertation tries to leverage the psychology research on *Learning Styles* (LS) – a measure of an individuals' preference to perceive and process information to help understand and improve the individual and team inspection performance. To gain quantitative and qualitative insights into the LSs of software inspectors, this dissertation reports the results from a series of empirical studies in university and industry settings to evaluate the impact of LSs on individual and team inspection performance. This dissertation aims to help software managers create effective and efficient inspection teams based on LSs and reading patterns of individual inspectors thereby improving the software quality.

ACKNOWLEDGEMENTS

Firstly, I would like to express my gratitude to my advisor Dr. Gursimran Singh Walia for his timely feedbacks and continuous guidance throughout my research work. I am grateful to Dr. Kendall Nygard and Dr. Saeed Salem for continuous participation in my research process. I am also grateful Dr. Mark McCourt for his collaboration to enable eye-tracking research using Center for Visual and Cognitive Neuroscience laboratory at the Department of Psychology. Thanks to Mr. Ganesh Padmanabhan and Dr. Lynnette Leone for arranging and setting up eye-tracking instrument for conducting studies. A special mention to Dr. Urvashi Rathod for motivating and guiding when I stepped in to the world of research as well as her contribution as an international collaborator in this research.

I am also grateful to all the faculty members, staff, and students in the Department of Computer Science who were involved during my Ph.D. program at North Dakota State University.

DEDICATION

I dedicate my dissertation to my family. A special feeling of gratitude to my loving parents, who motivated me throughout the process of research. My sister, who has never left my side and is very close to my heart. A special mention to my wife, who has been a constant source of support during the challenges of life.

TABLE OF CONTENTS

ABSTRACT iii

ACKNOWLEDGEMENTS iv

DEDICATION v

LIST OF TABLES ix

LIST OF FIGURES x

LIST OF ABBREVIATIONS xii

1. INTRODUCTION 1

1.1. Dissertation Goals 6

1.2. Key Terms 8

2. BACKGROUND 10

2.1. Inspections 10

2.1.1. Inspection Cost Model 11

2.1.2. Kusumoto Cost Metric (M_k) 12

2.2. Learning Styles 12

2.3. Eye Tracking 14

3. RESEARCH WORK: EMPIRICAL STUDIES AND MAJOR RESULTS 16

3.1. Feasibility Studies: Study 1 & Study 2 16

3.1.1. Artifact 16

3.1.2. Participants 17

3.1.3. Experiment Procedure 17

3.1.4. Data Collection 18

3.2. Research Approach 19

3.2.1. Principal Component Analysis (PCA) 19

3.2.2. Cluster Analysis (CA) 20

3.2.3. Discriminant Analysis (DA) 21

3.2.4. Generation of Virtual Inspection Teams 22

3.3. Replicated Studies: With Students: Study 3, With Professional in US: Study 4, With Professionals in India: Study 5 25

3.3.1. Artifact 25

3.3.2. Participants 26

3.3.3. Experiment Design 26

3.3.4. Data Collection 28

3.4. Tracking Eye Movements during Inspections: Study 6 and Study 7 28

3.4.1. Artifact 28

3.4.2. Eye Tracking Apparatus 29

3.4.3. Participants 30

3.4.4. Experiment Design 30

3.4.5. Data Collection 36

3.5. Results 40

3.5.1. Inspectors with certain LS have positive impact on inspections? 40

3.5.2. In each LS dimension, which LS category favors inspections? 46

3.5.3. How inspection performance is affected as dissimilarity in teams increased? 47

3.5.4. Are dissimilar teams more cost-effective as compared to random or similar teams? 58

3.5.5. Are results from previous studies validated when combined? 63

3.5.6. Is overall inspection performance affected by the way inspectors read requirements document? 65

3.5.7. Does inspection teams, based on LSs have a particular reading pattern that impacts their inspection performance? 67

3.5.8. Does inspectors belonging to a particular LS category have a reading pattern which supports inspection outcome positively? 68

3.5.9. What insights can be gained from the eye tracking and inspection results to help improve the readability of requirements development? 71

3.5.10. How does the reading patterns of high-performing inspectors varies across different LS categories and dimensions? 73

4. RELEVANCE TO SOFTWARE ENGINEERING 76

5. DISSEMINATION 77

6. CONCLUSION AND FUTURE WORK 79

REFERENCES 81

APPENDIX A. PRE-STUDY SURVEY 89

APPENDIX B. FAULT REPORTING FORM 91

LIST OF TABLES

Table		Page
1.	Grouping inspectors for team size 4 into clusters	23
2.	Group membership for 4 clusters	24
3.	Sample of reflection form for LAS document	33
4.	Six LS categories vs. Effectiveness and efficiency	46
5.	Multiple regression results for effect of each LS category	47
6.	Correlation between LS and inspection effectiveness for study 1	51
7.	Correlation between LS and inspection efficiency for study 1	51
8.	Correlation between LS and inspection effectiveness for study 3	53
9.	Correlation between LS and inspection effectiveness for study 4	54
10.	LS vs. Inspection performance of virtual team	58
11.	Paired sample test result	63
12.	Eye tracking vs. Inspection performance	66
13.	Eye movement vs. Virtual inspection teams of size 2-10	68

LIST OF FIGURES

Figure		Page
1.	Research framework	7
2.	Example result of the questionnaire on the ILS	14
3.	Feasibility study procedure	17
4.	Example to show statistical techniques (PCA, CA, and DA) used for virtual inspection team creation	21
5.	Tool approach	22
6.	Virtual inspection teams created from 4 clusters	25
7.	EyeLink 1000 desktop mount	30
8.	Inspection task in eye-tracking laboratory	34
9.	A sample fixation, scanpath, and heatmap	37
10.	Number of faults found by inspectors of different LSs for study 2	41
11.	Number of faults found by inspectors of different LSs for study 3	43
12.	Faults type found by participants in different LS clusters for study 2	44
13.	Faults type found by participants in different LS clusters for study 3	45
14.	Inspection effectiveness and efficiency of team size two to ten inspectors for study 1	49
15.	Inspection effectiveness of team size two to ten inspectors for study 3	52
16.	Inspection effectiveness of team size two to ten inspectors for study 4	54
17.	LS dissimilarity vs. Inspection effectiveness for varying team sizes for study 5	55
18.	LSs dissimilarity vs. Inspection effectiveness and efficiency for team size 2 to 10 for study 6 & 7	57
19.	Calculation of Kusumoto metric for team size 6	61
20.	Comparison of LS on cost-effectiveness for team size 2 – 9	61
21.	Meta-Analysis: comparing and combining results from four experiments	64

22. Comparison of eye movements of LS categories 69

23. Multiple regression results comparing inspection performance of LS categories based on eye movement factors 70

24. Page transition vs. Inspection performance 71

25. Heatmaps for introduction, general description and functional requirements page 72

26. Comparison of eye tracking data and inspection fault count between ACT/REF, SEN/INT, and SEQ/GLO LS category 74

27. Comparison of eye tracking data and inspection fault rate between ACT/REF, SEN/INT, and SEQ/GLO LS category 74

LIST OF ABBREVIATIONS

SDLC ..... Software Development Lifecycle

FC ..... Fault Count

FR ..... Fault Rate

PBR ..... Perspective Based Reading

NL ..... Natural Language

LS ..... Learning Styles

MBTI ..... Myers Briggs Type Indicator

FSLSM ..... Felder Silverman Learning Style Model

ACT ..... Active

REF ..... Reflective

SEN ..... Sensing

INT ..... Intuitive

VIS ..... Visual

VER ..... Verbal

SEQ ..... Sequential

GLO ..... Global

NDSU ..... North Dakota State University

PCA ..... Principal Component Analysis

CA ..... Cluster Analysis

DA ..... Discriminant Analysis

RIM ..... Restaurant Interactive Menu

LAS ..... Loan Arranger System

PGCS ..... Parking Garage Control System

CVCN ..... Center for Visual and Cognitive Neuroscience

1. INTRODUCTION

Software organizations focused on delivering quality software product within allocated budget and time [55]. Software industries follow various quality improvement approaches ranging from informal walkthroughs [27, 60] to formal checklist based inspections [45, 56] to prototyping [66] to testing [68]. All of these approaches aid developers to detect faults by inspecting software artifacts produced at different stages of development. While these methods are effective, evidence suggests that methods focused towards detecting faults introduced in the early software artifacts (i.e. requirements and design documents) of *Software Development Lifecycle* (SDLC) have most impact on the rework cost savings (i.e., the cost that are otherwise spent on fixing the faults) [16, 32]. Faults if left undetected are harder to find and fix [15] at the later stages of development. As a result, leading software organizations focus their attention on developing methods to aid developers in finding and fixing faults at the early stages of software development [24, 46].

To have most impact on software quality and rework cost savings, this research is focused on detecting faults committed during the development of requirement document (used to establish a problem space). Requirements are typically written in *Natural Language* (NL) where customer's needs for the software (to be developed) are documented in requirements document and is formally known as Software Requirements Specification (SRS). SRS is often written in *Natural Language* (NL) which is a means of communication among different stakeholders (e.g., technical and non-technical, end users and developers). However, due to the inherently flexible and ambiguous nature of NL, SRS development is prone to faults (e.g., incompleteness, ambiguity, inconsistency, correctness) [13, 14, 26].

Among different approaches (e.g., NL to State transitions [1, 2], Checklist based inspections [56], Scenario based reading [63], Ad Hoc inspections [59]) for detecting NL requirement faults, software inspection are widely recognized as most effective verification technique. During an inspection process, a team of inspectors is selected (depending upon their prior experience, or their technical background, or their domain knowledge) to review a software artifact by reading through it and recording faults during the process. The output of this process is a list of faults present in the artifact that can be fixed by the artifacts' author to avoid costly rework at the later stages [3, 27, 28]. While inspections have been reported to be widely beneficial, the evidence shows that, the effectiveness (i.e. Fault Count – FC) during the individual review significantly impacts overall inspection performance [58].

On that end, researchers [20, 21] have tried to investigate the factors (e.g., effect of educational background, level of technical degree of inspectors) that may impact the performance of individual inspectors and in turn improve the team efficiency (Fault Rate – FR). Research results at major software development organizations showed that inspection performance does not depends on the level of educational background [5, 21]. Contrary to the expectations, Software Engineers with a non-technical degree performed better inspections of a requirements document compared to the technical degree holders [21]. Even when inspectors utilize the same inspection technique, and receive same training, their effectiveness varies significantly. Hence, the results from the studies report that the *higher level of technical degree* and the *technical education background* had no impact on the inspection performance as compared to individuals with non-technical education background.

Research [9] summarizing the results from inspection studies of 25 years stated the open question: *What is the best way to staff software inspections?* One of the methods (e.g.,

Perspective Based Reading (PBR) [63]) utilized to staff inspectors relies on having inspectors assume different perspectives and review the document using a reading method for that particular perspective. In PBR technique, each inspector reviews an artifact from one perspective (e.g., user, designer, or tester). The idea of this method is to reduce the fault overlap by inspectors (by creating heterogeneous perspectives of reading) and to increase number of faults logged during inspections. The downside of PBR is, the number of perspectives of software developers are limited (only few in number) and the inspectors need to be matched (or else trained) to fit the perspective that would yield the best inspection result.

Research results [10, 34], regarding the correlation (or lack thereof) between technical knowledge vs. inspection performance led us to hypothesize that inspector's ability of detecting faults in requirements document may be affected by the ways with which an individual characteristically acquires, retains and retrieves the information known as *Learning Styles* (LS). Cognitive psychologists have studied LSs for a long time to gain insights into the learning strengths of individuals [6, 22, 53]. For example, some individual tends to work in group while some prefer to think about the information and work alone. Cognitive psychologists have been successful at developing an instrument for measuring the LS of an individual [31]. Research [4, 29] in cognitive psychology that uses LS has successfully crossed over to academia where LS of students are taken into account to improve scores in their performance in the course(s). Results show that if information is presented in the preferred LS of an individual, they perceive and process it better [6]. This in turn helps faculty members to design their course in a way that matches with the learning preference of students. LSs are more complex and measure user characteristics along different dimensions which could be used as different perspectives for inspections. Therefore, like PBR, we were motivated towards creating heterogeneous inspection

teams with inspectors of diverse LSs which could yield high inspection output. This dissertation tries to answer this question by borrowing the research from psychology to bear on the task of software inspection team development.

Project managers employing software inspections in their organizations need a reliable strategy that can aid them to determine effective inspection team from a pool of available inspectors. Using psychology measures to improve software teams is not a novel idea. On that note, Software Engineering researchers have borrowed cognitive and social psychology research to improve inspection team performance in past [52]. Using *Myers-Briggs Type Indicator* (MBTI) instrument [54] authors created heterogeneous inspection teams by maximizing cognitive disparity between team members. Despite these efforts, they have met with limited success because unlike LS instruments (that measure the cognitive learning preferences), MBTI is a personality inventory [53]. In software engineering domain, we were able to find one research [8] which takes in LS of software engineers in a geographically distributed team and suggests requirements elicitation tool which matches their LS preference. The results suggest that LSs of stakeholders varies significantly and they should be taken into account while selecting requirements elicitation methods to improve the quality of elicitation task in geographically distributed teams. This motivated us to evaluate whether LS could be used on improving individual and team performance in software requirements inspection.

After getting significant positive results from our experiments, where inspection team with inspectors of diverse LSs are more effective and efficient as compared to similar teams, we believed, we believe that high performing inspectors and inspection teams (who detect more number of faults in less time) have certain LSs and that, they tend to read inspection document in a certain fashion, to comprehend information depending on their LS preference.

Researchers in software engineering domain have characterized eye movements of software engineers during program comprehension [12], source code reviews [71], UML class diagrams [73], computer interface evaluation [35], user behavior in www search [37] to understand the reading patterns in the past. Some researchers [18, 51] have tried to understand reading patterns of individuals with different LS preference. Authors used an eye-tracking device to track the eye movements of participants while they were presented with one page information on screen. Results helped them to understand how individuals with different LS focus on the information presented (i.e. pictures vs. written sentences) and how they read them (i.e. sequentially vs. randomly). This directs our attention towards investigating the relationship between eye movement of inspectors during inspection and their inspection performance. Hence, we were interested to explore reading techniques followed by inspectors of different LSs during inspection. Tracking eye movements of inspectors where inspectors' eye movements will be recorded while inspecting requirements document (with multiple pages) during inspection on a computer monitor. An eye-tracking device was used to record eye movements of participants as they inspect a NL requirements document. During the experiment, the participants reported their LSs and performed an individual inspection of a requirements document (on a computer monitor) in an eye tracking laboratory settings. We analyzed the effect of LSs by measuring eye movement data of inspectors belonging to different LS groups with respect to their inspection effectiveness and efficiency. The results show that eye movement are significantly correlated with the inspection outcome in general and more positively correlated for inspectors with certain LS preferences.

1.1. Dissertation Goals

While LS have successfully been used in academia [4, 6, 29] to improve student's performance in their coursework, no concrete empirical evidence was found that focused on the reading strategies of inspectors with different LSs during inspection. Using these results as our motivation, we propose a systematic framework (Figure 1) where LSs of individual inspectors is taken into account and high performance inspection teams are generated based on LS input. This technique would guide software industries to detect more unique faults in early artifact (i.e. requirements document) of SDLC and thereby, leads to more fault coverage. Using these results as our motivation, we propose a systematic framework (Figure 1) where LSs of individual inspectors is taken into account and high performance inspection teams are generated based on LS input. The research framework describes a series of studies starting with the feasibility study that evaluated the impact of individual LS on inspection performance. Based on the promising results from feasibility studies, we investigated the impact the LSs had on team based inspections in terms of increasing LS diversity among inspection team members. Next, we replicated studies with academic as well as industrial participants to validate our previous results. The results of individual studies were also combined using meta-analysis technique to validate and generalize our results. Using eye tracking studies, the last phase of our framework investigated the relation between the reading pattern and inspection performance of inspectors in general as well as in teams. This technique would guide software industries to detect more unique faults in early artifact (i.e. requirements document) of SDLC and thereby, leads to more fault coverage. Inspection teams with inspectors of diverse LS preference would look an artifact from different perspectives and leads to less fault overlap during inspection. Creating inspection teams with diverse inspectors would lead to less number of faults to propagate in later stages of SDLC and

assures the quality software development. Hence, saves the amount of re-work, time, and cost involved in it.

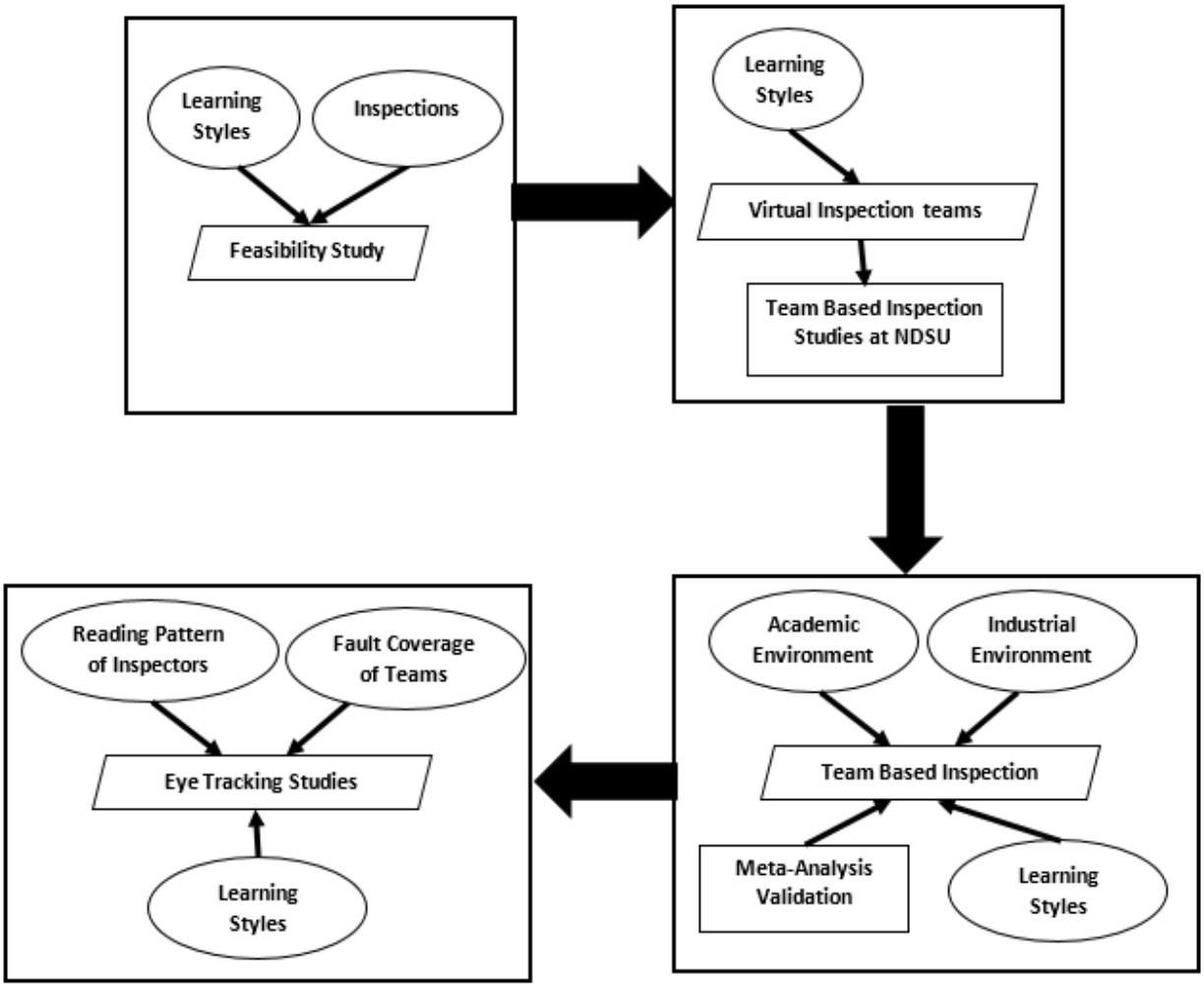


Figure 1. Research framework

Following are the major dissertation goals and research questions under each goal:

G1: Evaluate the impact of individual LSs on inspection.

RQ 1: Whether inspectors with certain LSs have positive impact on inspections?

RQ 2: In each LS dimension, which LS category favors inspections?

G2: Improving the effectiveness of inspection team performance based on LSs.

RQ 3: How inspection performance of inspection teams is affected when dissimilarity in LSs of inspectors is increased?

RQ 4: Are dissimilar teams are more cost-effective during inspection as compared to random or similar teams?

RQ 5: Are results from replicated studies validate previous results when combined?

G3: Characterization of reading pattern of inspectors.

RQ 6: Is overall inspection performance affected by the way inspectors read requirements document?

RQ 7: Does inspection teams, ranging from dissimilar to similar, based on LSs have a particular reading pattern that impacts their inspection performance?

RQ 8: Does inspectors belonging to a particular LS category have a reading pattern which supports inspection outcome positively?

RQ 9: What insights can be gained from the eye tracking and inspection results to help improve the readability of requirements development?

RQ 10: How does the reading patterns of high-performing inspectors varies across different LS categories and dimensions?

1.2. Key Terms

This subsection describes major terms used throughout the dissertation.

- *Learning Styles:* the characteristic strengths and preferences in the ways individuals take in and process information.
- *Fixations:* a point where eyes are relatively stationary and an individual in taking in the information.

- *Saccades:* quick eye movements between fixations.
- *Scanpath*: Series of fixations and saccades is known as a scanpath (i.e. Complete saccade – fixate – saccade sequence).
- *Gaze*: is the sum of fixations durations in an area. They are also known as "dwell", "fixation cluster", or "fixation cycle".
- *Region of Interest* (ROI): is an analysis method where eye movements that fall under certain area is evaluated (ROI in this study is the area where fault exist in the document).

Rest of the document is organized into following sections: Section 2 covers details about the concepts of inspections, LSs, and eye-tracking techniques. Section 3 elaborates completes research work on the grounds of research goals and research questions in it. Section 4 describes importance of our work to software engineering domain. Section 5 consists of publications with our research and Section 6 covers our conclusion and future work with our research.

2. BACKGROUND

This section describes the background terms that we have used in our experiment and how they are related to our research. Section 2.1 describes inspection and inspection cost model which was used to construct inspection teams of high performance which are also cost-effective in nature. Section 2.2 discusses about Learning Styles and its instrument (i.e. Index of Learning Styles) which is used to derive LS of individuals for our different studies. Section 2.3 describes background of eye-tracking in research and studies that used LS of individuals to track eye movements.

2.1. Inspections

The concept of inspection was introduced by Fagan [28] to detect and report faults in a software artifact. Inspection is widely used and is empirically validated [25, 27, 61] for early detection and elimination of fault in software artifacts. Researchers introduced many versions [49, 57] of inspection that emphasize different parts of the inspection process (e.g., placing more emphasis on the individual preparation phase and less emphasis on the team meeting phase). Inspection is carried out in the following steps:

- Inspection manager selects an inspection team of from a pool of skilled inspectors who are provided with the requirements document to be inspected.
- The requirements author provides a brief overview and background of the document.
- Each inspector performs an individual review by reading and reporting faults detected in a fault form followed by a meeting to create a master fault list.
- The master fault list is then handed back to the author to fix faults or explains why they are not faults.

We have used inspection data from fault-checklist based inspection technique to measure inspection performance of individual inspectors and inspection teams.

2.1.1. Inspection Cost Model

Inspection cost model [44] determines, how much cost is saved by inspections as compared to software testing. It has following components:

- C_r – cost spent on an inspection, is the sum of total time taken to inspect a document by each inspector.
- c_t – Average cost to detect a fault in testing, is not available during inspection. Therefore, it is measured as a factor of an average cost to detect a fault during an inspection. If a defect introduced at the earlier stage passes to the later stages, it requires rework which involves huge cost. Hence, it is always cost-effective to detect a fault as early as possible.
- D_{total} – total number of faults present in the software product, can be determined if a document seeded with faults or the number of faults found by multiple inspectors.
- D_r – number of faults detected: unique faults found during the inspection by all inspectors.
- C_t – testing cost: cost to detect remaining faults in testing, (i.e. $D_{total} – D_r$) after inspection. If we consider c_t as the average cost to detect a fault in testing, then the cost can be measured as the product of total number of faults remaining after inspection and the average cost to detect a fault during testing. This is, $C_t = (D_{total} – D_r) * c_t$
- ΔC_t – testing cost saved by inspection: by spending cost C_r during inspection, the cost ΔC_t is saved during the testing. It is calculated as the product of the total number

of unique faults found during the inspection (D_r) and the average cost to detect a fault in testing (c_t). That is, $\Delta C_t = D_r * c_t$

- C_{vt} – virtual testing cost, (i.e. testing cost if no inspections are performed) is the total of the cost required to detect faults left after inspection (C_t) and the testing cost saved by inspection (ΔC_t). That is, $C_{vt} = (C_t + \Delta C_t)$.

2.1.2. Kusumoto Cost Metric (M_k)

Kusumoto cost metric (M_k) [44] is a ratio of the reduction of the total costs to detect and remove all faults using inspections in a project to the virtual testing cost if no inspections were performed. The testing cost is reduced by (ΔC_t-C_r) compared to the virtual testing cost (C_t+ΔC_t) if no inspection is executed. The model proposed by Kusumoto normalizes the savings by the potential fault cost. Hence, it can be compared across different inspections and projects, and is deemed most appropriate for our research purpose. Accordingly, the Kusumoto metric can be derived as:

$$M_k = (\Delta Ct - Cr) / (Ct + \Delta Ct)$$

M_k is intuitive as it can be interpreted as the percentage of fault rework savings due to inspections. Using M_k, cost-effectiveness can also be compared across inspections on different projects. This research appropriately uses the Kusumoto metric to evaluate the cost effectiveness of inspection teams formed using the LSs of the inspectors (i.e. dissimilar, similar and no preference).

2.2. Learning Styles

The LSs was introduced by Kolb [43]. Since then, cognitive psychologists developed multiple versions of LS models [17, 23, 33, 41, 43, 50, 54] and validated the use of LS in engineering education [29]. We have used Felder Silverman Learning Style Model which is the

most advanced and widely used to measure LS preferences among individuals through an instrument known as *Index of Learning Styles* (ILS) [29–31, 64]. The FSLSM model classified individuals across four LS dimensions listed below:

- *Sensing* (SEN) Learners (Oriented towards facts, follow concrete content, data, careful with details, follow existing ways) or *Intuitive* (INT) learners (abstract, conceptual, innovative, oriented towards theories and meaning, discovering possibilities);
- *Visual* (VIS) learners (prefer visual representations of presented material – pictures, diagrams, flow charts, time line, video, demonstration) or *Verbal* (VER) learners (prefer written and spoken explanations);
- *Active* (ACT) learners (learn by trying things out, working in groups, discussing, explaining, brainstorming) or *Reflective* (REF) learners (learn by thinking things through, working alone, writing summaries);
- *Sequential* (SEQ) learners (linear, orderly, learn in small logical steps) or *Global* (GLO) learners (holistic, context and relevance of the subject, learn in large jumps).

LS of an individual is measured over these four dimensions, through an instrument called *Index of Learning Styles* (ILS), shown in Figure. 2. ILS is an online questionnaire, empirically validated for its reliability and construct validity [31], with 44 questions. Each dimension has 11 questions with two options favoring each category in that dimension. For example, in *Sensing vs. Intuitive* dimension, out of 11 questions, if an individual answered 8 in favor of *Intuitive* and 3 in favor of *Sensing*, then the final score will be 8-3 = 5 towards *Intuitive* category with an 'X' on the top (as shown in Figure. 2). The number of questions answered in favor of a LS category (i.e. 8 and 3 in the example) is also known as actual score in this research. Hence, the LS of an

individual from Figure. 2 is: REF-INT-VIS-SEQ (formed from combining a category from each dimension). A balanced person towards both categories is represented by the score of 1-3 on the ILS. A Score of 5-7 and 9-11 states that the person has a moderate and strong LS preference towards a category in a LS dimension respectively. We have used ILS to measure LS of participants for our experiments.

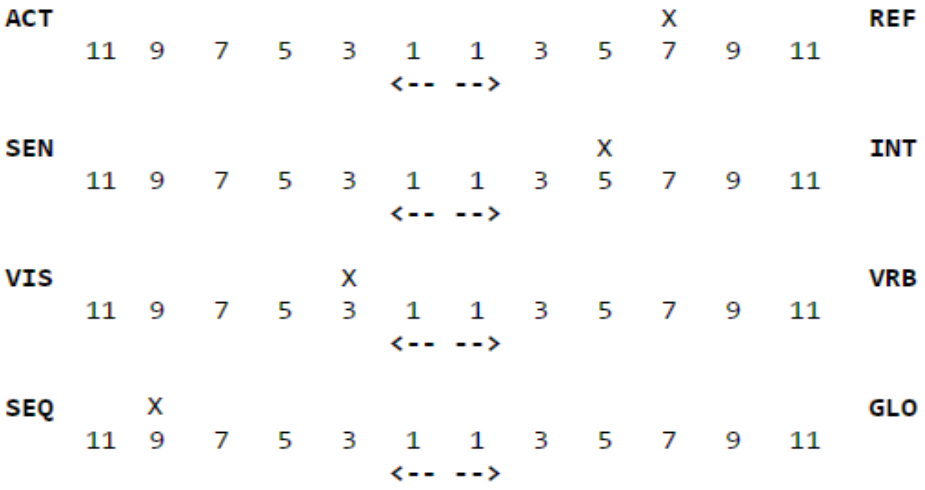


Figure 2. Example result of the questionnaire on the ILS

2.3. Eye Tracking

Eye movements and pattern of eye movements refers to the amount of cognitive processing by an individual [40]. Eye movement system is the result from Javal's gaze motion research in 1879. The system used set of mirrors to observe the eye movement of participants while reading [69]. The results showed that people tend to incorporate fixations and saccades instead of reading in a linear fashion. Modern eye tracking system works by reflecting infra-red light on an eye, and recording the reflection pattern. Early research [72] in eye tracking showed that, people tend to incorporate regressive *fixations* and *saccades* (instead of reading in a linear fashion) when faced with comprehension difficulty to review their understanding and retention.

Cognitive psychologists used eye tracking technology [18, 51] to understand Visual/Verbal and Sequential/Global LS preference of individuals by displaying information on a computer monitor. The results showed that visual learners tend to focus at the pictures whereas, sequential learners read sentences, took more time to read the information, and had less vertical eye movements. This study utilized eye movements to understand the reading patterns of students as they review requirements document. We have used eye-tracking to find reading patterns (i.e. their focus of attention on different parts and reading approach of requirement document) of inspectors and to correlate with their LS and inspection performance.

3. RESEARCH WORK: EMPIRICAL STUDIES AND MAJOR RESULTS

This chapter describes a series of studies that were conducted to validate the impact LSs had, on individual and team inspection performance. Section 3.1 describes feasibility studies conducted at NDSU that motivated the design of following experiments, Section 3.2 provides description of software tool that was developed to generate inspection teams based on LS input. Section 3.3 elucidates eye-tracking studies to find eye-movements of inspectors with different LSs, and Section 3.5 describes analysis and results for each study in detail in order to achieve our research objectives.

3.1. Feasibility Studies: Study 1 & Study 2

This study utilized participants from North Dakota State University to understand whether individual LS have an impact on inspection outcome. In both studies, LS of participants was gathered from online survey questionnaire. Participants then used the fault-checklist method to identify and record faults. The objective of these studies was to investigate whether LSs have an impact on inspections as an individual as well as in teams? Both the studies have the same design which is described in further subsections.

3.1.1. Artifact

Both the studies utilized the same requirements document (developed externally) that describe the requirements for Restaurant Interactive Menu (RIM) and contained naturally occurring faults (i.e. faults manifest during development of requirements document). RIM system is responsible for taking customer's orders in a restaurant with the help of an interactive online system. This gives customers a flexibility to make dining choices at their own pace and request assistance at their convenience. The system also gives restaurant owner better manageability over the menu, staff, inventory, and revenue/cost analysis.

3.1.2. Participants

Study 1 and 2 were conducted at North Dakota State University (NDSU). The eleven participants in study 1 and thirty-six participants in study 2 were undergraduate students enrolled in System Analysis and Design course. The participants were management students (i.e. not technical students). The course covers the understanding of the process of requirement and design development. Participants received training in-class via same instructor which involved reading of an SRS using standard fault-checklist method and recording faults found.

3.1.3. Experiment Procedure

Study 1 and study 2 follow the same experiment procedure where participants took LS survey, involved in inspection training based on fault-checklist based method, and performed individual inspections guided by fault-checklist method. Following are the detailed steps of the studies (Figure 3).

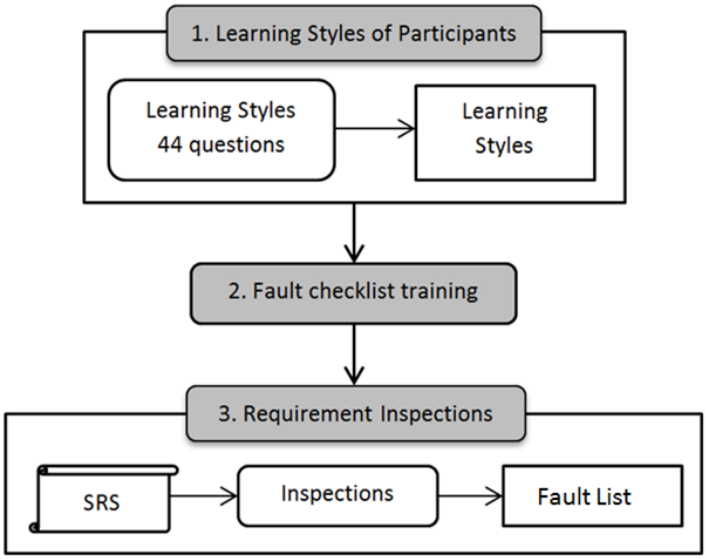


Figure 3. Feasibility study procedure

- *Step 1 - Learning Styles Questionnaire Survey:* at the beginning of the experiments, all participants were handed out Felder Silverman's LS questionnaire. Participants

answered all 44 multiple choice questions (https://www.engr.ncsu.edu/learningstyles/ilsweb.html) and, the LS results are generated for each participant on ILS scale. For each dimension on ILS (Active/Reflective, Sensing/Intuitive, Visual/Verbal, and Sequential/Global), the participant has score towards one category. Hence, only four LS categories (from each dimension) form LS of an individual with a score of either 1 or 3 or 5 or 7 or 9 or 11.

- *Step 2 - Training and Inspecting SRS for faults:* participants for both study 1 and study 2 were trained in-class by the instructor on how to use fault-checklist technique to detect faults in a given SRS. The inspection training lasts for one class session (i.e. 70 minutes) where they were given examples on how to detect and record faults in an SRS. Next, each participant was handed with RIM SRS for individual inspection and to log faults as their take home assignment.

3.1.4. Data Collection

The data from both studies includes the faults found by each participant using the fault checklist technique and the LS score of each participant. The participants used fault form to record the fault found during inspection. The fault form provides participants with the space to indicate inspection timing (i.e. start and end tie of inspection, time they found each fault, and break(s) if they took any). In addition, the fault reporting forms required participants to classify the fault identified during the inspection in one of the following fault types: General (G), Missing Functionality (MF), Missing Performance (MP), Missing Information (MI), Missing Environment (ME), Ambiguous Information (AI), Inconsistent Information (II), Extraneous (E), Wrong Section (WS), and Others (O). I evaluated the faults reported by each participant and

provided them feedback about true and false positives. I had the knowledge of the system for which requirements were developed and detection of true and false positives process includes reading through the faults reported by each participant to remove any false-positives before analyzing the data.

3.2. Research Approach

To achieve our research goals, a software tool [11, 36] was developed that generates virtual inspection teams that creates inspection teams from diverse to similar LSs of inspectors. This section describes various statistical techniques that were utilized to form inspection teams based on the LS of individual inspectors. Section 3.1 describes principal component analysis which was used to convert correlated variables (i.e. scores in LS categories across each LS dimension) to independent variables. Section 3.2 discusses cluster analysis which is used to group similar participants into different clusters based on their LSs. Section 3.3 explains discriminant analysis which is used to find out the probability of a participant to belong in a cluster.

3.2.1. Principal Component Analysis (PCA)

LS score of an individual is classified into two categories in each dimension (sensing/intuitive, visual/verbal, active/reflective and sequential/global). The relationship between two categories of each dimension is negatively correlated (i.e. if score on one category increases, the score on other decreases). PCA technique is utilized in this research to convert correlated LS scores (as shown in Figure. 2) across each dimension into uncorrelated variables that are also known as Principal Components (PCs) [7]. The possible number of PCs for each individual with a LS score is always less than or equals to 8 LS categories across four dimensions [70].

Each PC will account for a certain variance between the categories in each dimension and within the dimensions. The first PC will try to account for the maximum variance and the second PC will try to account for maximum variance that is not addressed by the first PC. This results in listing of PCs in order of decreasing variance where each PC represents a different property of the original data. Therefore, all possible number of PCs (i.e. less than or equal to the original number of variables) when combined, covers 100% variance of original data.

3.2.2. Cluster Analysis (CA)

The next step was to group similar participants into different clusters based on their LSs. CA is a multivariate technique, which form groups (also called clusters) with the objects that are relatively homogeneous within themselves and heterogeneous between each other [39]. The objective of CA in our research is to form clusters of similar individuals based on their LS data and to order teams that consists inspectors of dissimilar to similar LS preferences. The resulting clusters results in high similarity of LSs within each cluster and high dissimilarity of LSs between different clusters [65]. A team formed with inspectors of different clusters will lead to dissimilar team and a team formed with inspectors from the same cluster leads to a similar team.

We used k-means clustering technique [38], where the user inputs desired number of clusters (k or team size in our research), then the k-means assigns each participant to the nearest centroid out of k centroids. Next, participants are reassigned to the new closest centroid and the process is repeated until it results in no more changes [47]. The CA could be understood more from the Figure. 4, which shows two different clusters. The human clipart represents individual participants, which are then grouped into two different clusters (referred as active and reflective learners). Below each participant, values of active and reflective PCs are denoted. The

individuals belonging to each cluster are more similar in their characteristics (active cluster has participants with higher value of active PC and vice versa) as compared to other clusters.

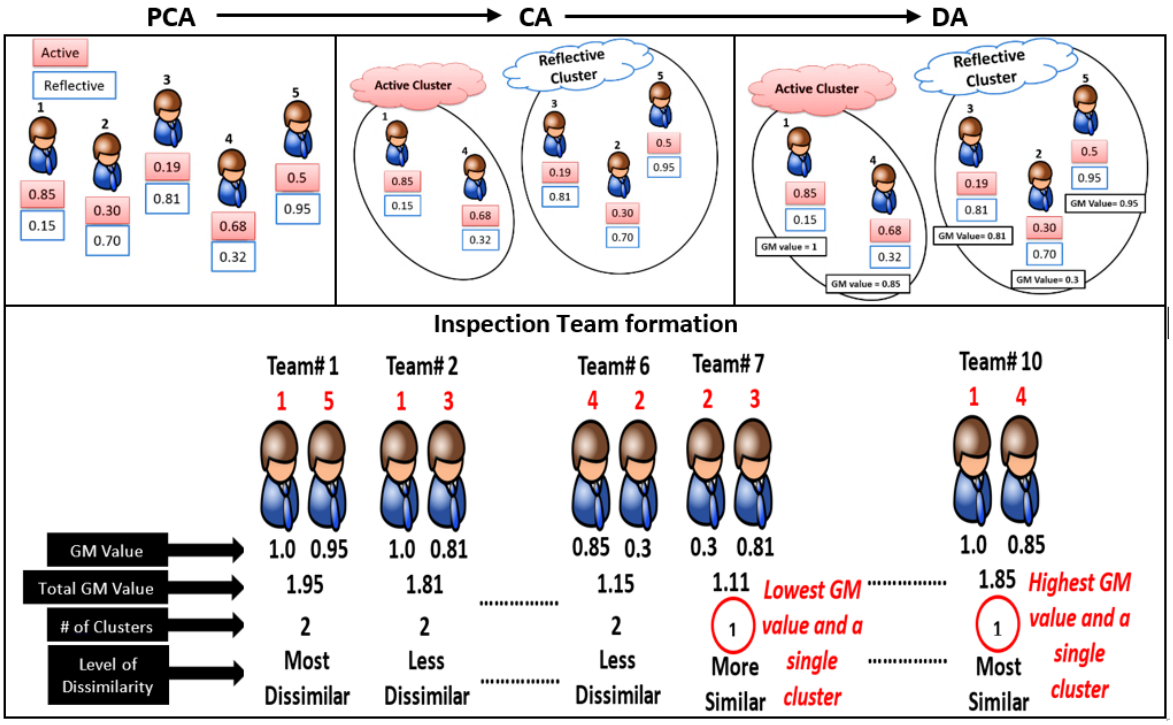


Figure 4. Example to show statistical techniques (PCA, CA, and DA) used for virtual inspection team creation

3.2.3. Discriminant Analysis (DA)

In the last step, DA was used to find out the probability of each participant belonging to a cluster. This result of the DA is used to maximize the LS variations across different clusters, and minimize the LS variations within each cluster [7, 42, 67]. DA provides *Group Membership* (GM) which is used to determine the dissimilarities between individual LSs within the same cluster in this research by using the difference between the GM values of individuals.

For example, in Figure. 4, GM values are indicated for each inspector in active and reflective PCs. In active cluster, inspector 1 has the highest GM value and in reflective cluster, inspector 5 has the highest GM value. When both inspectors (i.e. 1 and 5) are selected from each

cluster, they form the most dissimilar team. GM was used in our research to sort the teams ranging from most dissimilar LS to teams with most similar LS preferences and strengths.

3.2.4. Generation of Virtual Inspection Teams

The tool follows above techniques, shown in Figure 5, to generate inspection team of a particular size (r) based on LS of each inspector.

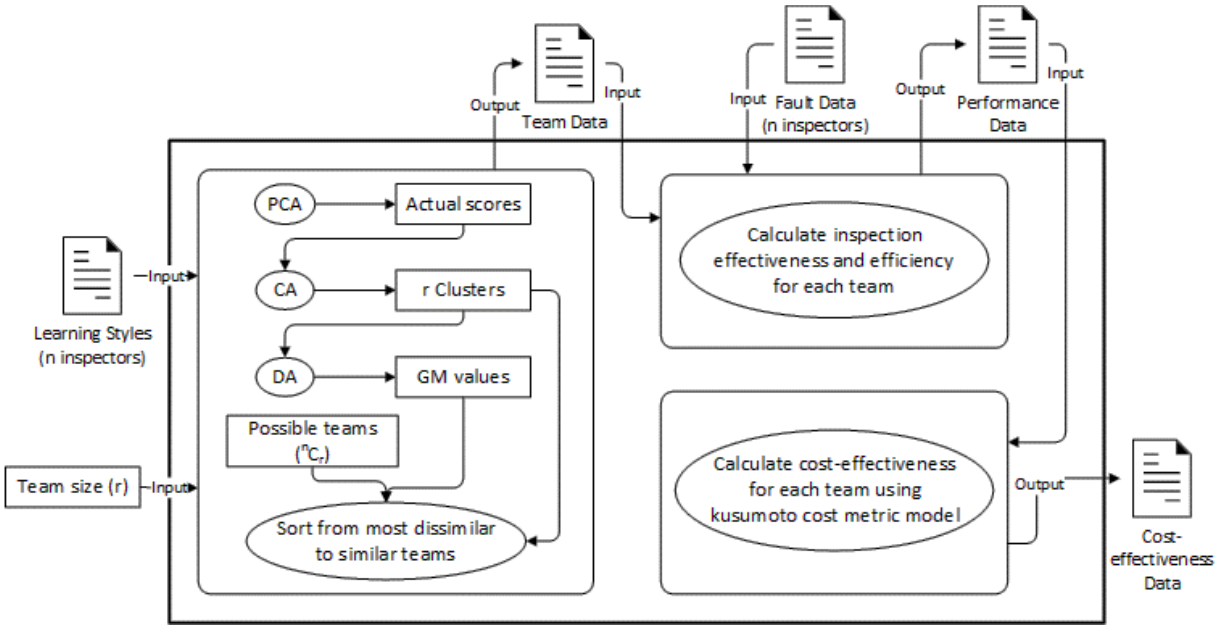


Figure 5. Tool approach

- *Step 1 – Create virtual inspection teams:* Virtual inspection teams (i.e. teams that did not actually meet) were created to determine the effect of LS on effectiveness and efficiency for various team sizes. For each experiment, we created inspection teams of size ranging from 2 to 10 with all the possible combinations of virtual teams. For example, to create inspection team size 4 out of 32 inspectors, we created 35960 inspection teams (i.e. $32C_4$) from the pool of 32 inspectors.
- *Step 2* – Grouping *of similar inspectors in clusters:* The LS scores of each participant is first converted into actual scores which has number of answers supported for each

category in a dimension. The correlated LSs of each inspector in each dimension were converted into uncorrelated variables by using PCA (Section 3.2.1). Next, based on PCs, different clusters (same in number as team size) were created and inspectors of similar LSs were grouped together using CA (Section 3.2.2). Table 1 shows the cluster output of team size of 4 inspectors (from data set 2) where first row represents cluster number and the second row shows the inspector ID who belongs to a cluster. In this example, for team size 4, tool creates 4 different clusters (i.e. equal to team size). Inspector 1, 5, 6, 7, 10, 15, 16, 20, 22, 24, 27, 28 and 32 grouped into cluster C1 and similarly, rest of the inspectors belong to cluster C2, C3 and C4.

Table 1. Grouping inspectors for team size 4 into clusters

Cluster No	**C1**	**C2**	**C3**	**C4**
Participant ID	1	12	3	2
	5	17	9	4
	6	23	22	8
	7	26	13	18
	10		21	19
	15		31	25
	16			29
	20			30
	22			
	24			
	27			
	28			
	32			

- *Step 3 – Sorting teams based on the LS of inspectors:* In this step, using DA, each inspector is assigned a GM value within the same cluster (shown in Table 2). First column within each cluster (i.e. C1, C2, C3 and C4) represents the inspector ID

number and second column shows the GM value of inspectors in their respective clusters. As shown in Table 2, for cluster C1, participant 5, 15 and 22 has GM value of 1 which is the highest probability to belong in cluster C1 as compared to rest of the participants in the same cluster.

Table 2. Group membership for 4 clusters

C1		C2		C3		C4	
ID	**GM**	**ID**	**GM**	**ID**	**GM**	**ID**	**GM**
1	0.98	12	1.00	3	1.00	2	0.76
5	1.00	17	1.00	9	0.81	4	1.00
6	0.79	23	0.89	11	0.99	8	1.00
7	0.99	26	0.68	13	1.00	18	0.98
10	0.97			14	1.00	19	0.99
15	1.00			21	0.90	25	0.99
16	0.93			31	0.95	29	1.00
20	0.99					30	0.90
22	1.00						
24	0.97						
27	0.97						
28	0.99						
32	0.97						

- Next, all inspection teams (i.e. $32C_4$, from step 1) were sorted in the order of decreasing level of LS dissimilarity (i.e. most dissimilar to similar) of the individual inspectors as shown in Figure. 6. Team with inspector number (ID) 5, 8, 12 and 13 is the most dissimilar team. The level of LS dissimilarity for most dissimilar teams (team with maximum number of clusters involved) in a team decreases (shown by decreasing value of total GM) as we move down in Figure. 6 till team number 2912. Similarly, as we move down the team list (Figure. 6) for most similar teams (i.e. lowest number of clusters or single cluster involved), similarity of LS preference

among inspectors decreases. Team number 35140 is the most similar team with highest GM value that represents the most similarity among the team members. The process was repeated for team size 2 to 10.

Team#	ID1	C#	GM	ID2	C#	GM	ID3	C#	GM	ID4	C#	GM	Total Clusters	Total GM
1	5	1	1.00	8	4	1.00	12	2	1.00	13	3	1.00	4	**4**
2	8	4	1.00	12	2	1.00	13	3	1.00	15	1	1.00	4	**4**
3	8	4	1.00	12	2	1.00	13	3	1.00	22	1	1.00	4	**4**
4	5	1	1.00	8	4	1.00	12	2	1.00	14	3	1.00	4	**4**
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2911	2	4	0.76	6	1	0.79	21	3	0.90	26	2	0.68	4	**3.13**
2912	2	4	0.76	6	1	0.79	9	3	0.81	26	2	0.68	4	**3.04**
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
35140	5	1	1.00	7	1	0.99	15	1	1.00	22	1	1.00	1	**3.99**
35141	5	1	1.00	15	1	1.00	22	1	1.00	28	1	0.99	1	**3.99**
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
35957	2	4	0.76	18	4	0.98	29	4	1.00	30	4	0.90	1	**3.64**
35958	2	4	0.76	18	4	0.98	19	4	0.99	30	4	0.90	1	**3.63**
35959	2	4	0.76	18	4	0.98	25	4	0.99	30	4	0.90	1	**3.63**
35960	12	2	1.00	17	2	1.00	23	2	0.89	26	2	0.68	1	**3.57**

Most Dissimilar Teams Set

Most Similar Teams Set

Figure 6. Virtual inspection teams created from 4 clusters

3.3. Replicated Studies: With Students: Study 3, With Professional in US: Study 4, With Professionals in India: Study 5

This section explains the replication studies that were conducted with different set of participants in the same academic environment, with industrial participants in United States, and with industrial participants in India. The previous results were validated in these studies with different number of participants as well as requirements documents. Later, all the results were combined using meta-analysis technique to generalize our results.

3.3.1. Artifact

Study 3 utilizes the same RIM requirements document (used in study 1 and study 2) with naturally occurring defects as described in Section 3.1.1. Study 4 and 5 utilized Loan Arranger System (LAS) document developed externally by Microsoft. LAS system provides functionality

that allows a financial organization to sell group of loans to other financial organizations. The organization purchasing the loan is provided with the capability to search for loans based on outstanding value, remaining term of the loan and, risk. The LAS document is 10 pages long and consists of 49 detailed requirements for the system. This document has been used in previous inspection studies and has been seeded with 30 defects [19, 21, 62].

3.3.2. Participants

Study 3 comprises of thirty-two participants enrolled in undergraduate course of System Analysis and Design at NDSU. Nineteen software professionals working in a software company participated for study 4. Some of them have worked on multiple projects in industry. The participants reported to have an average of three years of experience in interacting with user to writing and inspecting requirements and use cases. Forty professionals (with varying level of industrial experience) enrolled in Requirements Management workshop at Symbiosis International University (SIU) participated for study 5. The workshop covered the process of managing requirements as practices used in the real-world projects for delivering quality software product.

3.3.3. Experiment Design

Study 3 at NDSU follows the same experiment design as described in Section 3.1 for study 1 and study 2. This study varies on environment and time aspects as compared to study 1 and study 2. Inspections for this study held in-class and participants were given 70 minutes to complete their inspection task.

Study 5 was conducted as a workshop for participants to understand the concepts and details of requirements inspection. Participants in study 4 and study 5 went through a pre-study survey questionnaire. This helped us to gain some insights into their education background and

experience before experiment and their feedback after experiment. Following is the detailed experiment design for study 4 and study 5.

- *Step 1 - Pre-study questionnaire:* participants in both study 4 and study 5 went for pre-study survey questionnaire (http://rebrand.ly/PreStudySurveyIndustry) where they have to answer details about their educational background and experience of working in software industry. The survey also has a breakdown to get detailed information about their work experience of working with requirements phase during software development. This consists of their experience of interacting with user to write requirements, writing use cases, inspecting requirements, and changing requirements for maintenance. On average, participants in study 4 had about 3 years and participants in study 5 had about 2 years of experience.
- *Step 2 - Learning Styles Questionnaire Survey:* This step follows the same procedure for both study 4 and study 5 as described in Section 3.1.3 step 1.
- *Inspection Training:* For study 4, participants were provided with the training material (in the form of PowerPoint slides). For study 5, the participants were trained for thirty minutes. During training, examples were given to the participants on how to detect faults in a requirements document using fault checklist method.
- *Step 4 - Inspecting SRS for Faults:* Study 4 and 5 participants received LAS SRS document after their training where participants have to read through SRS individually, detect faults and report them in fault form as described in Section 3.2.3 Step 3. For study 4, participants received 60 minutes to complete their inspection task.

3.3.4. Data Collection

This part for all three studies (i.e. study 3, study 4, and study 5) follows the same steps as described in Section 4.2.4. For study 4 and study 5, master fault list (with 30 defect list) was used to detect false-positives from each individual's inspection data.

3.4. Tracking Eye Movements during Inspections: Study 6 and Study 7

As, described in Section 1, the use of eye-tracking method has been studied for different phases of SDLC expect requirement inspections. Since, individual has different viewing strategies and have a different learning preference; hence, we hypothesize that inspectors tends towards different reading approach during inspection which could affect inspection performance. We also want to gain insights into eye-movements of inspectors with different LSs. Therefore, we conducted studies with students of NDSU. This study validates the LS with the eye-movements of inspectors by inspecting an artifact at CVCN (Center for Visual and Cognitive Neuroscience) lab of NDSU. Research questions of this experiment are described in Section 1.1. The LS, fault, and eye-tracking data was collected for each participant. The results from this study would help us to find whether reading pattern of inspectors with certain LS aids in detection of more faults during inspection.

3.4.1. Artifact

For both the studies, participants inspected a common document (developed externally) that described the requirements for Parking Garage Control System (PGCS) developed externally at University of Maryland was used for inspection in eye tracking settings. The document describes requirements for a parking system that manages entry and exit of vehicles automatically. The system displays the status of parking garage to each driver while entering and even provides monthly access card for drivers in need of reserved parking space. PGCS

document was seeded with 34 realistic faults and have been used previously in inspection studies [19, 62]. PGCS is a generic domain and was selected to avoid any impact of domain knowledge bias. For both the studies, inspections took place by reading requirements document on a computer monitor with the eye-tracker tracking eye movement. Hence, document was modified in Microsoft Word 2013 to include line number before each line and was converted into high resolution image (1080 x 1920) for each page. This modification of SRS and its conversion was done with an objective of page by page navigation during inspection on computer monitor and collection of eye-tracking data for each page. Inspectors were able to point out faults by calling out line number before intended line (where fault was found) and describing faults with less disruption in eye-tracking data.

3.4.2. Eye Tracking Apparatus

For both the studies, EyeLink 1000 desktop mount, shown in Figure. 7, was used to track and record eye movements of participants during the inspection. The instrument consists of three main components: (a) high-speed camera, (b) infrared illuminator, and (c) a host PC connected via Ethernet dedicated to record eye movements. It is a non-invasive system that sits at the bottom of computer monitor/projection area (i.e. below the tracked area the participant is viewing) and allows participants to read from a monitor while eye tracker allows eye movement recordings with a sample frequency of 250-2000 Hz, a tracking range of 32^{o} x 25^{o} with an accuracy greater than 0.5 degree and a resolution less than 0.01 degree.

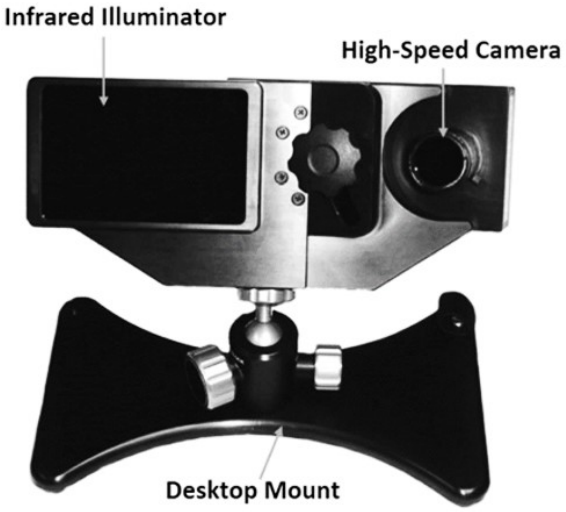


Figure 7. EyeLink 1000 desktop mount

3.4.3. Participants

Study 6 and Study 7 were conducted at North Dakota State University (NDSU). A total of thirty-nine graduate and undergraduate students participated in this study. Thirteen graduate students for (study 6) were enrolled in Requirements Engineering course and twenty-three undergraduate students for (study 7) were enrolled in System Analysis and Design course during at North Dakota State University (NDSU). System Analysis and Design course covers the basic understanding of requirements, design, and development of software system. Requirements Engineering course focuses on the requirements development technique and tasks which includes requirements inspection technique. Both courses were taught by the same instructor and students in both courses were required to learn software inspections and their impact on software quality improvement.

3.4.4. Experiment Design

To evaluate the relationship of reading patterns and LSs of requirements inspectors on their inspection performance, two studies were conducted in Computer Science course where LS of participants were gathered via online questionnaire. Participants were then trained on fault-

checklist based inspection process. Participants then individually performed requirements inspection in an eye-tracking laboratory settings. Their raw LSs, eye-tracking data along x & y axis, and timestamp and fault data (i.e. number of faults found and time taken) was collected to analyze the reading patterns of individual inspectors and virtual inspection teams of different LSs using EyeMMV toolbox (https://github.com/krasvas/EyeMMV). EyeMMV is a complete utility for post-experiment eye movement analysis for generating *fixation*, *scanpaths*, and *heatmaps*. This toolbox was used to visualize *fixations* (i.e. certain area where inspectors focused for a larger amount of time during inspection), *scanpaths* (i.e. what is the path of reading requirements document during inspection), and *heatmaps* (i.e. areas, represented by different colors, in the document which gained more attention of an inspector during the inspection process).

Following are the steps followed during the studies:

- *Step 1 - Learning Styles questionnaire survey:* at the beginning of the experiments, all participants were handed out Felder Silverman's LS questionnaire (https://www.engr.ncsu.edu/learningstyles/ilsweb.html). Participants answered all 44 multiple choice questions and, the LS results are generated for each participant on ILS scale. For each dimension on ILS (Active/Reflective, Sensing/Intuitive, Visual/Verbal, and Sequential/Global), the participant has score towards one category. Hence, only four LS categories (from each dimension) form LS of an individual with a score of either 1 or 3 or 5 or 7 or 9 or 11.
- *Step 2 - Training on Inspecting SRS for Faults:* Participants in both the studies were trained in-class by the instructor on how to use fault-checklist method to detect faults in SRS for 70 minutes. To ensure that students understood the fault inspection, as part

of their course objectives, students applied fault-checklist technique on LAS document (Step 2a) followed by a reflection of their inspection results (Step 2b).

- *Step 2a – Inspection of LAS document:* during this step, participants used their training to perform an individual inspection of LAS document and reported faults.
- *Step 2b – Reflection of LAS inspection results:* One of the researcher evaluated the faults reported by each participant and provided them feedback about true and false positives. Next, post inspection reflection was performed wherein, participants were provided a list of original 30 faults in LAS document (that they had inspected) and were asked to reflect upon the faults they saw (but did not reported) or missed during the inspection by comparing it against their reviewed fault list. Table 3 shows a sample of reflection document and each column is described as follows:
- Defect#: represents the defect ID in seeded fault list.
- *Req.#:* indicates the requirement ID(s) where fault is present.
- *Type:* denotes categorization of faults into different fault categories. For example, ambiguity (A) in the requirements.
- *Description:* brief description of the problem for an author to be able to understand and fix it.
- *Is it a defect:* Whether students to agree or disagree that the fault represents an actual requirement problem?
- *Did you see this:* Whether they were saw this fault (‘yes’ or ‘no’) during the inspection?
- *Did you report this:* Whether (‘yes’ or ‘no’) they reported this fault during inspection of LAS document?

- *Explain:* this field allowed a brief explanation if their response in the three earlier fields were inconsistent.

Table 3. Sample of reflection form for LAS document

Defect #	Req.#	Type	Description	Is it a defect?	Did you see this?	Did you report this?	Explain
1	1, 2	A	Are the reports in these requirements the same or separate?				
2	1, 2	O	When do the updates occur? Are they effective immediately?				

- *Post Reflection,* students discussed their doubts regarding inspections and reflection of their faults with researchers. A week after this exercise (to avoid *fatigue* effect), participants were provided with the quick recap of fault checklist based inspection technique.
- *Step 3 - Inspecting PGCS requirements via Eye tracker* - Next, each participant performed an individual inspection in eye-tracking laboratory as shown in Figure 8. Each participant individually read through the PGCS document on a computer monitor (rotated in portrait mode as requirements were documented in portrait mode). Throughout the process, EyeLink 1000 eye tracker sitting at the bottom tracked their eye movements during the inspection. One of the researchers was present in the eye-tracking laboratory that assisted the participants during the inspection process. At the beginning of the experiment, researcher gave the overview of the PGCS SRS document to be inspected, fixed eye distance with eye-tracking instrument, calibrated (i.e. focusing at known locations on the computer screen) and validated (determining

whether apparatus estimation of eye position is indeed close to the known position of the targets) eye movements, and drift correction (to correct small drifts in the calculation of gaze position). During the experiment, if participant needed break, researcher paused the inspection process and resumed it after performing drift correction.

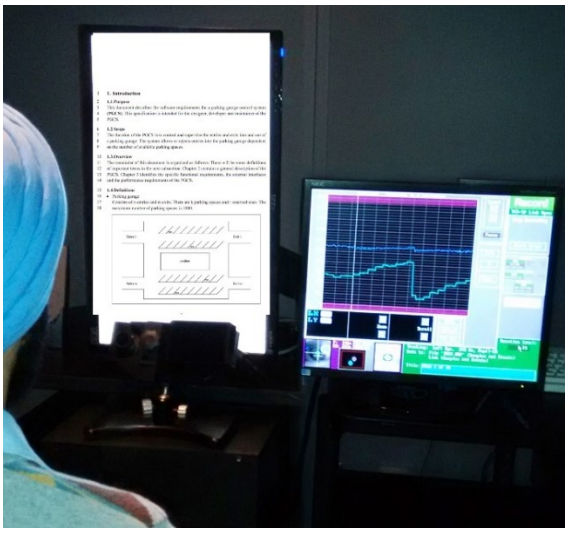


Figure 8. Inspection task in eye-tracking laboratory

- During the initial setup, participants were aligned at the center of a computer monitor (both horizontally and vertically). The distance between eye and camera was fixed between 55cm – 60cm. This distance was chosen through pilot testing with sample participants to evaluate if they were comfortable in reading document from the monitor. Also, this distance lies in the optimum distance to capture eye movements by the eye-tracker as suggested by SR research in EyeLink 1000 manual (i.e. between 40 cm – 70 cm) [74].
- Participant's eye movements were calibrated with the eye-tracker, using sampling rate of 250Hz, and then validated via EyeLink 1000 built in software. The validation

checks for the eye movement readings and compares it with the calibration reading to avoid any deviation in the data. Based on the comparison, results are generated (on the eye-tracking system screen) as poor, fair, good, and excellent. If results are poor, the validation process was repeated again. Before starting the inspection process, a final drift correction of participant's eye was performed (a corrective adjustment based on raw eye position of a participant).

- Each page of requirements document is displayed on the monitor at a resolution of 1080 × 1920 in portrait mode. Using right and left click on mouse, participants were able to switch pages of PGCS document forward and backward. Eye-tracker at the bottom of the monitor captures the eye movements along 'x' and 'y' axis of the monitor continuously during the entire inspection task. Inspection task also required the participants to talk-it-out-loud any faults that they discover during the inspection process so that they don't have to look away from the screen to disrupt eye-tracking. A voice recorder was used to assist the fault reporting. The participants speak out loud the place (line number) where they found fault and describe the fault to establish why it represents a problem.
- During the entire process, participants were allowed to take break(s). In such case, researcher with the control of eye-tracking system paused the eye-tracking and the voice recorder. During breaks, participants were encouraged to relax their eyes, look away from monitor but were not allowed to move their chair to avoid disruption of eye calibration. Whenever participant wanted to resume, drift correction was performed again and inspection was resumed (i.e. recording of eye movement was started again) from the same page where inspection was paused. After completing

inspection, recorded faults (in the form of audio) are transcribed as a fault list along with the timing data (i.e. start and finish times, time when each fault was found, breaks) for each participant.

- *Step 4 -Post-inspection*: Researchers again provided a complete list of defects in the PGCS document to the students (similar to LAS reflection) and asked them to reflect on their inspection experience. We also discussed the issues they may have faced in eye tracking environment to gain insights into their inspection results.

3.4.5. Data Collection

This section describes the raw eye tracking data collected during the inspection for study 6 and 7. The PGCS document was marked with twenty-seven ROI's where faults were present. The XY_{start} and XY_{end} coordinates of region where faults were present in the document were calculated using a software tool known as IrfanView (http://www.irfanview.com). We simply dragged the mouse pointer to select the ROI and released after selecting. IrfanView automatically records the coordinates of the region selected in a text file. ROI measure was used to analyze whether all participants in general and participants belonging to a certain LS were able to focus at the areas where faults were present. The eye tracking data (i.e. coordinates along 'x' and 'y' axis and timestamp) was written into *Export Data Format* (EDF) file by EyeLink 1000 software. To enable these files to be read into MATLAB system, they were first converted into ASCII format. While converting into MATLAB file, the unwanted data (e.g., during blink of an eye where no eye movement was recorded) was cleansed before analysis.

This converted eye movement data (in .mat extension format) for each participant was inputted to EyeMMV toolbox (https://github.com/krasvas/EyeMMV) for offline analysis. The tool runs under MATLAB environment and uses different functions to identify ***fixations***,

saccades, generate ***heatmaps***, and analyze ***ROI***. A sample eye movement visualization in the form of fixation, scanpath and heatmap for page 11 (marked with ROI's) is shown in Figure 9.

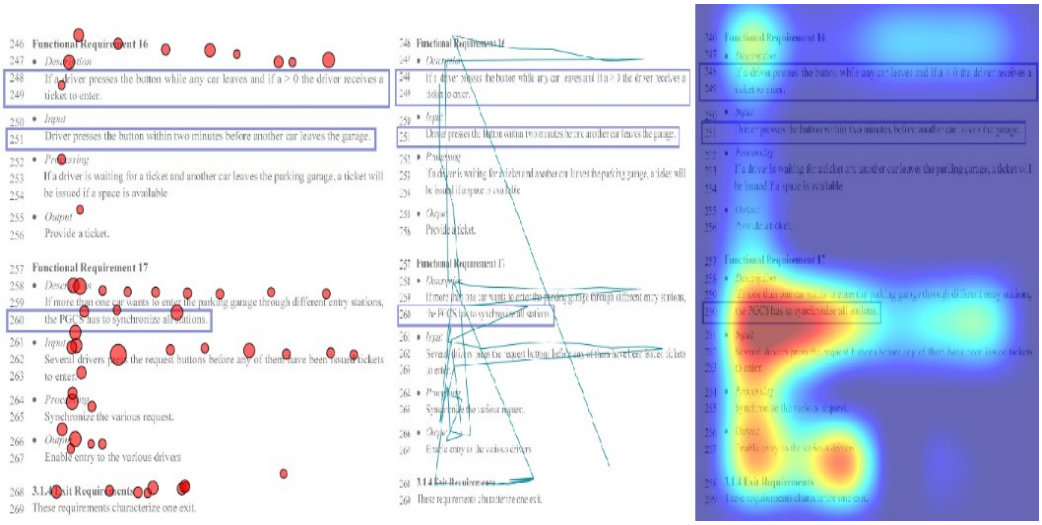


Figure 9. A sample fixation, scanpath, and heatmap

The tool extracted the gaze data of each participant and filtered the results (e.g., eye blinks where no data was recorded). The resulting data was then used for analysis in this study.

Apart from eye movement data along x & y axis and time for each participant, other variables were calculated and saved into separate files (.mat extension) using raw eye tracking data into EyeMMV toolbox. The variables are listed below:

- *Total time (T_{total})*: Each participant spent some time (in milliseconds) reviewing and reporting faults on each page. Hence, a total of fourteen durations were calculated (one for each page) and converted into seconds by EyeMMV toolbox.
- *Fixation time ($T_{fixation}$)*: During the review, participants tended to fixate on certain parts in the document (where eye movements are relatively stationary → coordinates roughly remains same with the time). These parts (fixations) involves cognitive processing and the radius of each fixation depicts the amount of time spent [40]. The total fixation time for each page was calculated. Fixation events were detected and

visualized using an algorithm based on spatial and temporal constraints [35]. The fixation identification depends of three basic parameters: two spatial parameters (tolerance 1: *t1* and tolerance 2: *t2*) measured in pixels and one minimum duration (minDur). The *minDur* value suggested by the algorithm was between 100ms to 900ms. "*t1"* depicts how tight a fixation cluster will be and "*t2"* depicts the discrimination between clusters. As inspection required reading in which participants fixated for a smaller duration, we used *t1=50, t2=25, minDur=150* to visualize fixations.

- *Scanpaths (S_{linear})*: To investigate the reading trend of inspectors, we utilized *scanpaths* during the first occurrence of reading the PGCS document. This was done because, while reading back, inspectors tend to search for information in the document (or report faults) that may not be indicative of their reading style. To differentiate between linear and random reading motion, we considered saccades that were greater than or equals to an angle of 30 degrees. We calculated percentage of linear motion on each page by taking ratio of linear saccades to all saccades on each page.

The EyeMMV tool also calculated following data at ROI's (Figure 9. area marked in SRS as rectangular boxes) during inspection task:

- F_{ROI}: total number of fixations by the participant at ROI in the PGCS document.
- T_{ROI}: total time taken (in seconds) by the participant to read through the ROI's during inspection.

For visualizing heatmaps (Figure 9), EyeMMV uses a parameter of gridSize (*defined in pixels*) which is used to generated heatmap from the point data. Grid size is inversely

proportional to the number of different regions generated on heatmaps and Bicubic interpolation was also used to smooth out the heatmap generated. For generating heatmaps in our experiment, we used *gridSize=135*. Heatmaps followed a color scale that helps investigator to find out the area which received attention at different levels of focus (based on different colors). The color scale in our experiment uses blue, green, yellow, orange, and red to represent minimum (blue) to maximum (red) region of attention.

The PGCS requirements document was also divided into three major sections: (a) Introduction, (b) General Description, and (c) Functional Requirements. During the inspection, it was observed that participants went back and forth between different pages or different sections to search for some information to gain context of the system and to report faults. To achieve that, if a participant is at page 7 and wants to go back at page 4, they had to switch between pages 6, 5, and then 4 using left mouse click. This pattern was evaluated by collecting data regarding the number of times a previous section was revisited (or referenced) and the total time spent on referencing previously read sections. To collect this data, eye tracking data (the sequence by which pages were read, fixation & saccade information, time spent) was exported into Microsoft Excel format and a script was written in *Visual Basic for Applications* (VBA). The script removed eye movement data on intermediate pages that were not targeted during search with a criteria of fixation number less than or equals to 10. Accordingly, following variables were collected in context of referring different sections backwards:

- I_{count}: number of times the introduction section was referenced;
- GD_{count}: number of times the general description section was referenced;
- FR_{count}: number of times the requirements section referenced;

- T_{count}: Total number of times a previously read section was referenced (I_{count} + GD_{count} + FR_{count});
- I_{time}: time spent in referring back to the introduction section
- GD_{time}: time spent in referring back to the general description section
- FR_{time}: time spent in referring back to the requirements section
- T_{time}: time spent in referring back to previously read sections (I_{time} + GD_{time} + FR_{time})

Therefore, a total of 13 eye movement variables were collected and analyzed to find the impact of reading pattern on inspection performance.

3.5. Results

This section reports results from studies conducted over a period of time to answer research questions in order to achieve our research goals.

3.5.1. Inspectors with certain LS have positive impact on inspections?

Results from feasibility study (i.e. study 2) and replicated study (study 3) are described in this section. The motive was to find, do LS have an impact on inspections or not? If yes, which are the LS that favors inspection outcome? While looking at the raw LS data, it was found that, out of 36 inspectors in study 2, only one had a preference towards the verbal LS category (with the score of 1 on ILS) and out of 32 inspectors in study 3; only 4 had a preference towards the verbal category with a score in the balanced range (i.e. 1 or 3 on ILS). So, we removed Visual/Verbal dimension for analysis from both studies. Therefore, only remaining 6 LSs (*Active* – ACT, *Reflective* – REF, *Sensing* – SEN, *Intuitive* – INT, *Sequential* –SEQ, and *Global* – GLO) were analyzed. We haven't covered data from study 1 as number of participants were very less (i.e. 11).

Using 6 LS categories across 3 LS dimensions, we created 8 clusters of possible LS combinations namely: 1) ACT-SEN-SEQ, 2) ACT-INT-SEQ, 3) REF-SEN-SEQ, 4) REF-INT-SEQ, 5) ACT-SEN-GLO, 6) ACT-INT-GLO, 7) REF-SEN-GLO, and 8) REF-INT-GLO. Using the LSs results, participants were then grouped into these clusters. As an example, in Figure 2, ILS score sheet represents that a person has a preference towards REF, SEN, and SEQ LSs and can be placed in the REF-SEN-SEQ cluster.

All 36 participants in Study 2 were grouped into 8 clusters based using their ILS score sheet. The number of participants belonging to each cluster after this step is: ACT-SEN-SEQ (***seven***), ACT-INT-SEQ (***three***), REF-SEN-SEQ (***nine***), REF-INT-SEQ (***zero***), ACT-SEN-GLO (***five***), ACT-INT-GLO (***three***), REF-SEN-GLO (***six***), and REF-INT-GLO (***three***). The REF-INT-SEQ cluster (that contained zero participants) was removed from further analysis. Next, the effectiveness of each cluster was calculated by averaging the number of unique faults found by inspectors belonging to that cluster. This process is repeated for every cluster and for the efficiency results (i.e., average faults rate). Figure. 10 shows the average effectiveness and efficiency of participants belonging to each cluster (the results are ordered by the most *effective* to the least *effective* cluster).

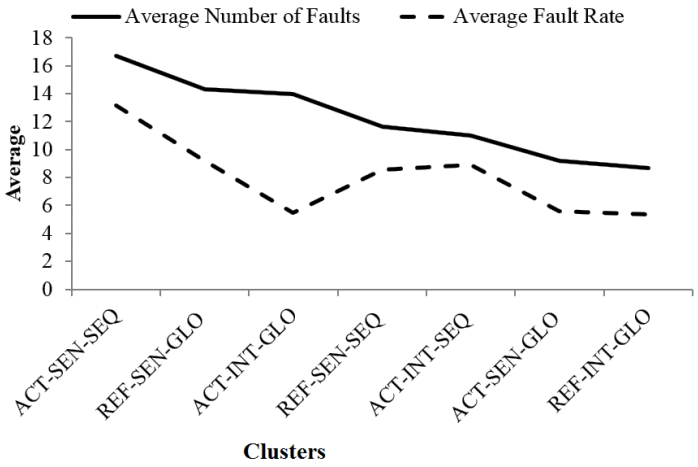


Figure 10. Number of faults found by inspectors of different LSs for study 2

The major result from Figure 10 is that, the inspectors with ACT-SEN-SEQ LSs had the maximum average effectiveness and the maximum average efficiency. Conversely, the inspectors with REF-INT-GLO (i.e., exact opposite of the ACT-SEN-SEQ) LSs uncovered the least number of faults and at a slowest pace during the requirements inspection. Based on the results presented in this section, inspectors with ACT-SEN-SEQ are best suited for an effective and efficient inspection of requirements document. This means that, an effective and efficient inspector is more oriented towards facts, concrete data and is careful with the requirement details, and these inspectors prefer step by step learning and follow a logical process to read through a requirements document and record faults present in the document.

For study 3, only effectiveness of participants is analyzed due to fixed inspection timings in this study. 32 participants were grouped into 8 clusters based on their ILS score sheet. The number of participants belonging to each cluster after this step is: ACT-SEN-SEQ (***five***), ACT-INT-SEQ (***two***), REF-SEN-SEQ (***nine***), REF-INT-SEQ (***four***), ACT-SEN-GLO (***five***), ACT-INT-GLO (***two***), REF-SEN-GLO (***two***), and REF-INT-GLO (***three***). Effectiveness for each cluster is calculated.

Figure 11 shows the average effectiveness of participants belonging to each cluster (results ordered from the most effective to less effective cluster). The major result from Figure 12 is that, the inspectors with REF-INT-SEQ LSs (categories with positive correlation) had the maximum average effectiveness. Conversely, the inspectors with ACT-SEN-GLO (i.e. exact opposite of REF-SEN-SEQ) LSs uncovered the least number of faults.

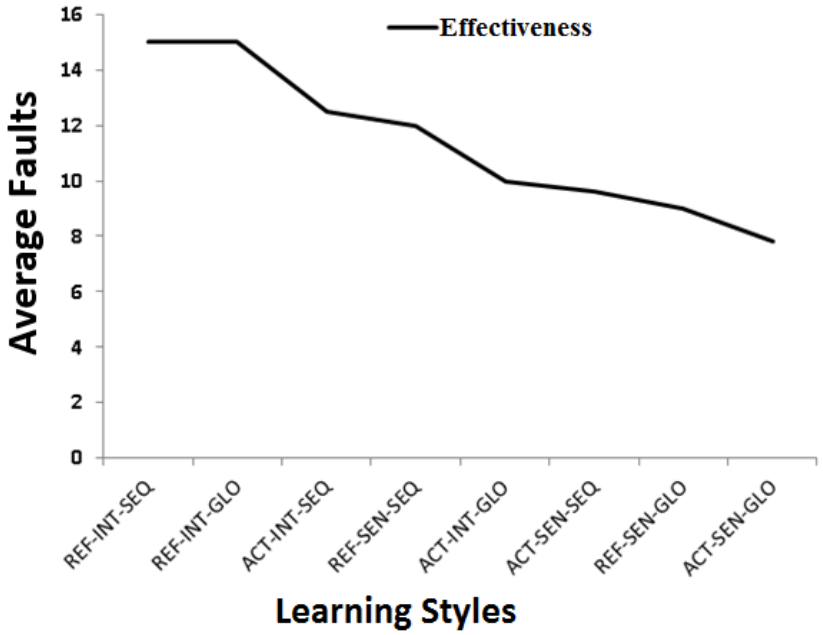


Figure 11. Number of faults found by inspectors of different LSs for study 3

Based on the results presented in this section, inspectors with REF-INT-SEQ are best suited for an effective and efficient inspection of requirements document. This means, an effective and efficient inspector thinks about the information first, tends to understand from theories and its meaning written in requirements document, and these inspectors prefer step by step learning and follow a logical process to read through requirements document and record faults present in the document.

We were also interested in finding out the impact of LS on the type of faults found during inspections. To perform this analysis, we compared the classification of faults (explained in the data collection Section 4.1.4) found by the participants belonging to each cluster (listed above and shown in Figure. 11). Figure 12 (for study 2) and Figure 13 (for study 3) compares the average number of unique faults detected by each cluster (e.g., ACT-SEN-GLO) for each fault type (e.g., MF, AI).

Results from study 2 showed that, the inspectors belonging to the ACT-SEN-SEQ cluster found the maximum number of General (G) faults and Ambiguous Information (AI) faults. Another observation is that, the REF-SEN-SEQ cluster detected the maximum number of Inconsistent Information (II) faults. Also, REF-SEN-GLO inspectors detected maximum number of Missing Information (MI) faults. The inspectors in the ACT-INT-GLO found the most number of Missing Functionality (MF) faults. This shows that, a combination of different LSs enabled inspectors to find higher number of a particular fault type. This gave an indication that difference in the LS of inspectors will enable a higher coverage of faults present in an artifact. The result also revealed that inspectors belonging to the four LS clusters (i.e., *ACT-SEN-SEQ*, *ACT-INT-GLO*, *REF-SEN-SEQ*, *REF-SEN-GLO*) out of 8 clusters were able to uncover all the types of faults present in a requirements document.

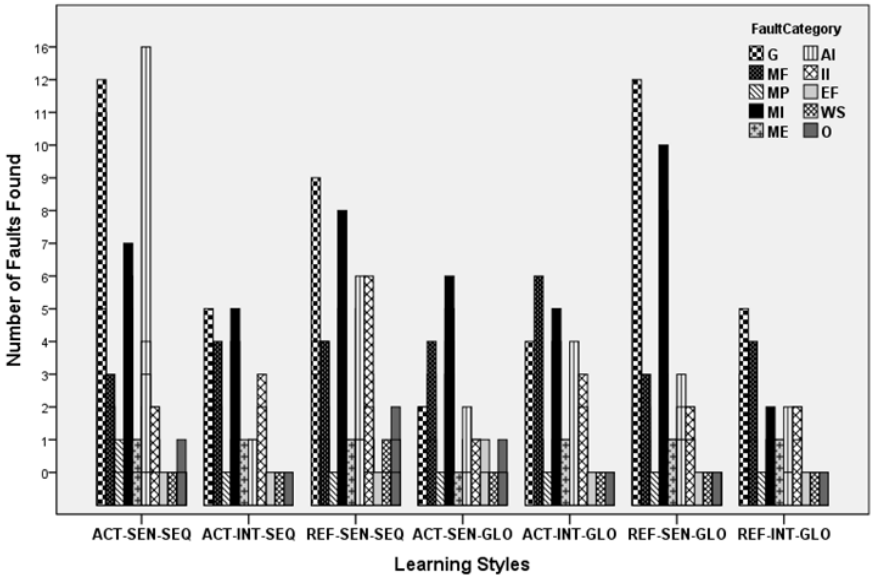


Figure 12. Faults type found by participants in different LS clusters for study 2

Results from study 3 (Figure 13) showed that, the inspectors to the REF-INT-SEQ cluster found the maximum number of Missing Information (MI), Missing Environment (ME) and General (G) fault type. Another observation is that REF-INT-GLO cluster has found the

maximum number of Ambiguous Information (AI) fault type. Also, ACT-INT-SEQ cluster found the maximum number of Inconsistent Information (II) and Extraneous (EF) faults and ACT-INT-GLO cluster found maximum of Missing Functionality (MF) faults. The result also revealed that inspectors belonging to the five LS clusters (i.e., *REF-INT-SEQ*, *REF-INT-GLO*, *ACT-INT-SEQ, ACT-INT-GLO, ACT-SEN-GLO*) out of 8 clusters were able to uncover all the types of faults present in a requirements document.

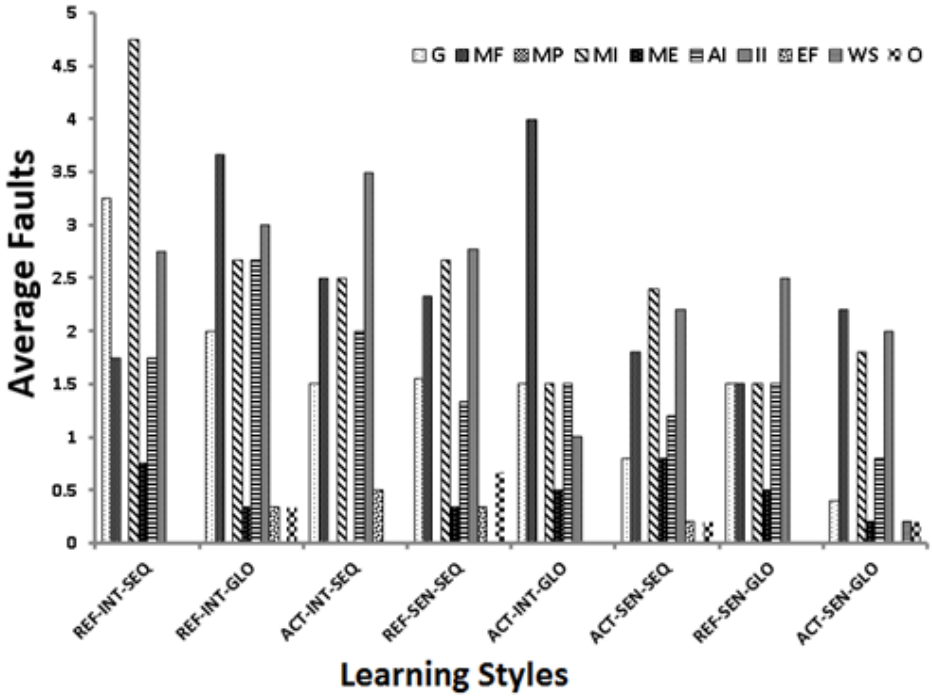


Figure 13. Faults type found by participants in different LS clusters for study 3

Results presented above clearly showed that difference in the LS of inspectors will enable a higher coverage of faults present in an artifact. These results also gave an indication that individual LSs have an impact on the inspection effectiveness of inspectors. Inspectors with certain LSs do find more faults as compared to others during the inspection.

3.5.2. In each LS dimension, which LS category favors inspections?

To investigate the impact of LS categories in each dimension (except VIS/VER), we performed multiple regression test for study 2 and study 3.

The results in Table 4 (for study 2) show that ACT, SEN, and SEQ categories had a positive relationship (represented by shaded cells) with the number of faults found by individual inspectors. The SEN and SEQ LS categories were also positively correlated to the fault rate of individual inspectors. Conversely, the REF, INT, GLO learning styles are negatively correlated to the inspection results. This means that the inspectors with a higher preference towards ACT, SEN, and SEQ learning styles are more likely to perform an effective and efficient inspection. While this result is positive, it was not found to be significantly correlated. An analysis of the LS data showed that there was not enough variation in the participants LSs within each category that is required to obtain a significant correlation to the inspection results. This means, each category out of two in a dimension does affects inspection performance positively.

Table 4. Six LS categories vs. Effectiveness and efficiency

LS category	ACT	REF	SEN	INT	SEQ	GLO
Effectiveness: Correlation	.036	-.079	.005	-.106	.082	-.196
p-value	.418	.327	.489	.273	.321	.129
Efficiency: Correlation	-1.33	-.117	.166	-2.17	.239	-.334
p-value	.223	.252	.171	.105	.083	.025

The results in Table 5 (for study 3) shows that REF and SEQ categories have a significant positive correlation (represented by shaded cells) with the number of faults found as well as fault detection rate by individual inspector. Conversely, ACT and GLO LS categories have significant negative correlation with the inspection performance. Also, INT LS category has a positive

(whereas, SEN is negatively correlated) but does not have significant correlation with effectiveness and efficiency of inspection. This means that inspectors with a higher preference towards REF, INT and SEQ LSs perform an effective and efficient inspection.

Table 5. Multiple regression results for effect of each LS category

LS Category	ACT	REF	SEN	INT	SEQ	GLO
Effectiveness: Pearson Corr.	-0.169	0.504	-0.007	0.084	0.379	-0.305
p-value	0.178	0.002	0.485	0.323	0.016	0.045

All these results are in accordance with the results presented in Section 4.5.1 where high performing LS (e.g. REF-INT-SEQ) had the same set of LS categories from each dimension which favors inspection positively and vice versa is also true. Results presented in Section 4.5.1 and 4.5.2 shed a light that LS do have an impact on inspection outcome and constructing teams with inspectors of different LS might provide better fault coverage. Therefore, improved inspection performance.

3.5.3. How inspection performance is affected as dissimilarity in teams increased?

LSs of the 11 (from study 1), 32 participants (from study 3), and 19 participants (from study 4) as well as their fault data to generate all possible virtual inspection teams for inspection team size of 2 up to 10 inspectors as described in Section 3.2. For each inspection team size (2 to 10), inspection teams were sorted in decreasing order of the LS dissimilarity amongst the inspectors that make up the team. The unique fault count (i.e. effectiveness) and the fault rate (i.e. efficiency) of each inspection team.

Inspection data of participants from three studies was individual data; so we had to combine individual results to form a team score for each virtual inspection team. Fault detection *effectiveness* for teams was found by combining the count of unique faults each participant had

found in RIM requirements document. The reason for choosing unique fault count is because we were interested in overall fault coverage by a team as opposed to high overlap in the fault list. *Efficiency* is the number of unique faults found by each team in one hour (calculated by dividing total unique fault count by the total time spent during the inspection). The time taken by each participant in a team is combined to calculate the overall time took by team to complete inspection process. This analysis was performed all possible virtual inspection teams for all sizes. We haven't analyzed efficiency for study 3 and study 4 as inspection timings were fixed for both studies. Following are the results described for each study.

To provide an overview of the results, Figure 14 shows the *average number of faults* (shown by solid line) and the *average number of faults per hour* (shown by dotted line) found by teams of 2 to 10 inspectors for study 1. To understand the impact of the LS, for each team size, the *effectiveness* and the *efficiency* results are organized by the increasing number of clusters involved in the team formation (the higher the cluster number, the higher is the dissimilarity in LSs of team members). Note that the number of clusters that could participate in the team formation is always less than or equal to the team size (i.e., 1 or 2 clusters for team size 2; 1 or 3 clusters for team size 3 and so on). However, for some larger team sizes (e.g., team size 8 in Figure 14), virtual inspection teams could not be formed using only 1, or 2, or 3, or 4 clusters. This is because for team size 8, the tool creates 8 different clusters via k-means algorithm. All 11 participants were distributed across these 8 clusters. There was no single cluster that contained all the participants. Therefore, no virtual teams were created (for team size 8) from only one cluster. Simply put, as the team size increases, the number of participants that belong to same cluster decreases which reduces the probability that a team will be formed from less number of clusters.

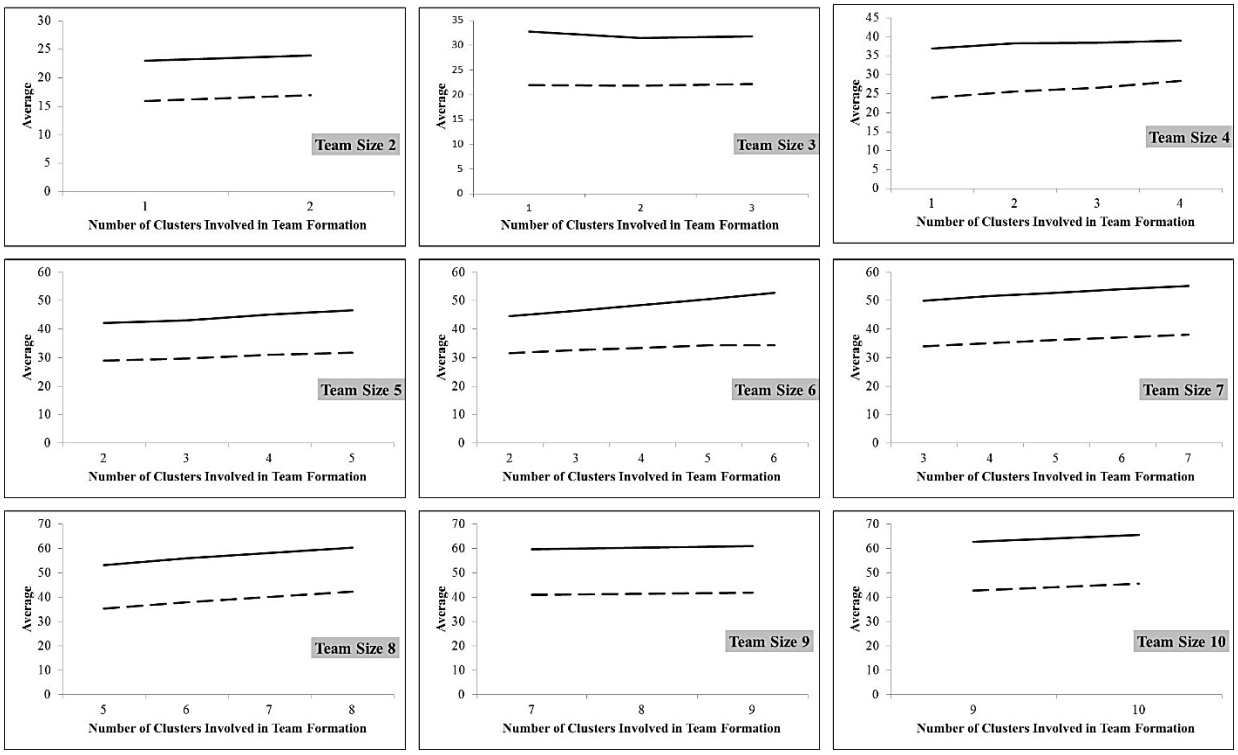


Figure 14. Inspection effectiveness and efficiency of team size two to ten inspectors for study 1

The *effectiveness* results (solid lines in Figure 14) showed that, for team size of 4 up to 10 inspectors, teams that include inspectors belonging to more number of clusters (i.e., inspectors with dissimilar LSs) are able to detect larger number of faults as compared to teams with inspectors belonging to fewer number of clusters (i.e., inspectors with more similar LSs). This result is not evident for team size of 2 and 3 inspectors where the inspectors belonging to the same cluster were almost as effective as compared to the inspectors pooled from the different clusters. Analysis of the raw data revealed that, in case of 3 or less number of inspectors, the inspectors were distributed over 2 or 3 clusters.

As a result, a clear separation of LSs of inspectors across different clusters was not achieved with a small number of clusters. After team size 3, inspection effectiveness of teams shows a consistent increase with an increase in the number of clusters used to formulate inspection teams. That is, teams with diverse LSs of inspectors are more effective. We also

compared the effectiveness and efficiency result within the same cluster for each team size (e.g. all inspection teams of size 6 generated from 2 clusters) based on the decreasing value of GM (which denote the decrease in their dissimilarity of the LSs within that cluster size). We did not found any consistent decrease in the inspection abilities of teams within a particular cluster size as the GM value decreases. This might be due to the fact that inspectors within the same cluster are very similar in their LS preferences even if they have different GM values within the same cluster. The *efficiency* results (dotted lines in Figure 14) also follow the same trend as effectiveness. That is, for team size 3 to 10, teams with inspectors belonging to different clusters (i.e., are dissimilar in their LSs) are able to detect more faults per hour as compared to teams with similar inspectors.

We performed a linear regression test to see whether the dissimilarity in the LSs of inspectors is positively correlated with the number of unique faults found by inspection teams of different sizes. The results show that, dissimilarity in the LS of inspectors had a strong and significant positive correlation (shown in Table 6) with the team effectiveness for team size 5, 6, 7, and 8. There is no significant correlation for team size of 2, 3 and 4 inspectors, which was due to the fact that the team size ranging from 5 to 8 inspectors covers most of the LSs (across 8 LS categories). Diverse set of inspectors in these teams were able to find different types of faults (also discussed later) resulting in an increase in their team effectiveness. Also, for team size 9 and 10, though the team effectiveness increased with an increase in the cluster size, this increase was not significant because there are only 8 LS categories and adding more number of inspectors in a team increased the overlap of the faults.

Table 6. Correlation between LS and inspection effectiveness for study 1

Team size	Correlation with the faults (within same team size)
5	$p<0.001$; $r^2=0.037$
6	$p<0.001$; $r^2=0.079$
7	$p=0.002$; $r^2=0.030$
8	$p<0.001$; $r^2=0.118$

Similarly, a linear regression test evaluated the correlation between the dissimilarity in the LSs of inspectors and the fault rate of inspection teams. The results show that, the LS had a strong and significant positive correlation (Table 7) with team efficiency for team sizes 4, 5, 6, 7, and 8. Therefore, based on these results, creating a team of 5 up to 8 number of inspectors based on the dissimilarity in their LS strengths (guided by the number of clusters involved in their formation) does appear to significantly increase the fault detection effectiveness and fault detection efficiency during an inspection of requirements document.

Table 7. Correlation between LS and inspection efficiency for study 1

Team size	Correlation with the efficiency (within same team size)
4	$p=0.006$; $r^2=0.023$
5	$p<0.001$; $r^2=0.028$
6	$p=0.005$; $r^2=0.017$
7	$p=0.001$; $r^2=0.032$
8	$p<0.001$; $r^2=0.157$

For study 3 results, the *effectiveness* results (solid lines in Figure 15) showed that, for team size of 4 up to 10 inspectors, teams that include inspectors belonging to more number of clusters (i.e., inspectors with dissimilar LSs) are able to detect large number of faults as compared to teams with inspectors belonging to fewer number of clusters (i.e., inspectors with more similar LSs). This result is not evident for team size of 2 and 3 inspectors where the

inspectors belonging to the same cluster were almost as effective as compared to the inspectors pooled from the different clusters. Analysis of the raw data revealed that, in case of 3 or less number of inspectors, the inspectors were distributed over 2 or 3 clusters. As a result, a clear separation of different LSs of inspectors across various clusters was not achieved with a small number of clusters. After team size 3, inspection effectiveness of teams shows a consistent increase with an increase in the number of clusters used to formulate inspection teams. That is, teams with diverse LSs of inspectors are more effective.

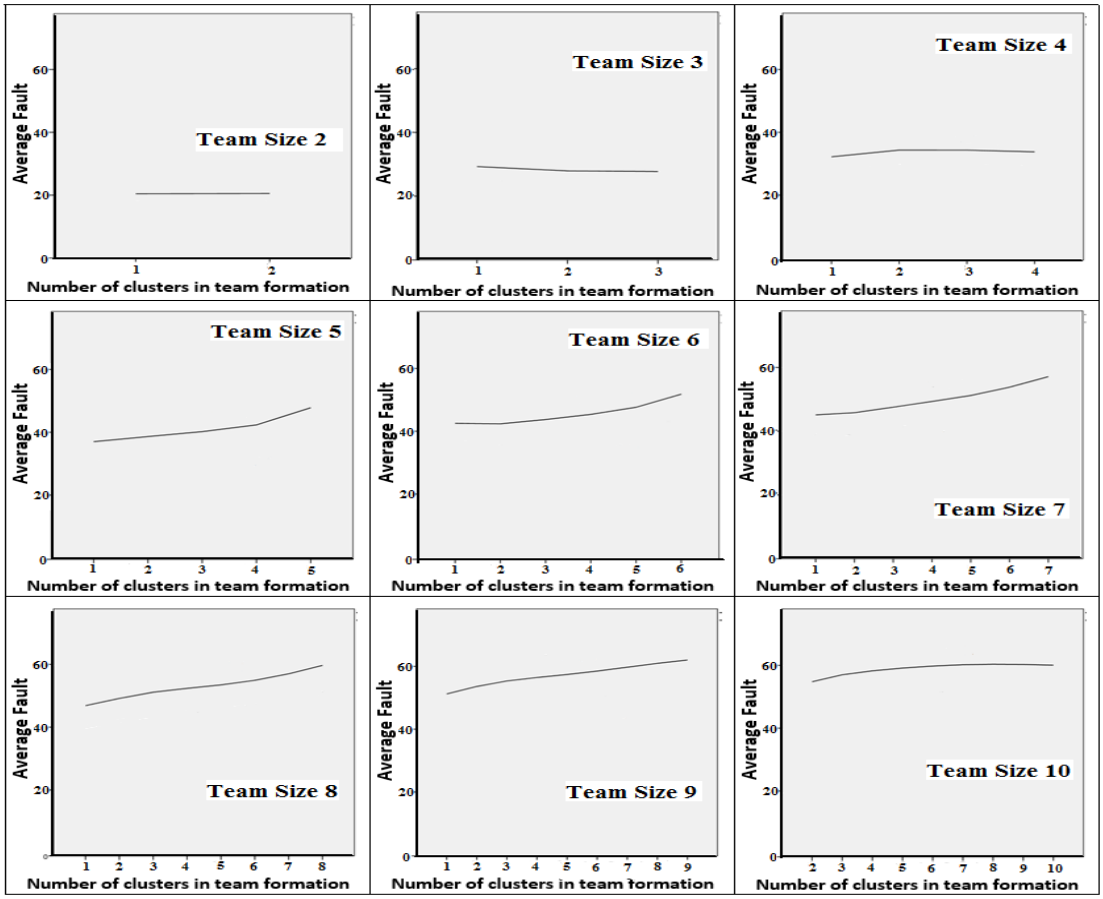


Figure 15. Inspection effectiveness of team size two to ten inspectors for study 3

Liner regression results (Table 8) show that, dissimilarity in the LS of inspectors had a strong and significant positive correlation with the team effectiveness for team size 4 to 10.

There is no significant correlation for team size of 2 and 3 inspectors, which was due to the fact that the team size ranging from 4 to 8 inspectors covers most of the LSs (across 8 LS categories). Diverse set of inspectors in these teams were able to find different types of faults (also discussed later) resulting in an increase in their team effectiveness.

Table 8. Correlation between LS and inspection effectiveness for study 3

Team size	Correlation with the efficiency (within same team size)
4	$p=0.048$; $r^2=0.000$
5	$p=0.001$; $r^2=0.043$
6	$p=0.001$; $r^2=0.046$
7	$p=0.001$; $r^2=0.059$
8	$p=0.001$; $r^2=0.034$
9	$p=0.001$; $r^2=0.028$
10	$p=0.001$; $r^2=0.005$

Similar analysis was done with study 4 data. Figure 16 compares the average number of faults found by virtual inspection teams. Results are organized by the increasing number of clusters (or increasing dissimilarity) involved in the team formation (i.e., the higher the cluster number, the more dissimilarity the team members). To reiterate, the *number of clusters* could participate in the team formation is always less than or equal to the team size (e.g., 1 or 2 or 3 clusters for team size 3). Based on the results in Figure 16, for each inspection team size (2-10), the inspection teams formed with more number of clusters (i.e., dissimilar LSs) are more effective at finding defects as compared to the teams with inspectors belonging to a fewer number of clusters.

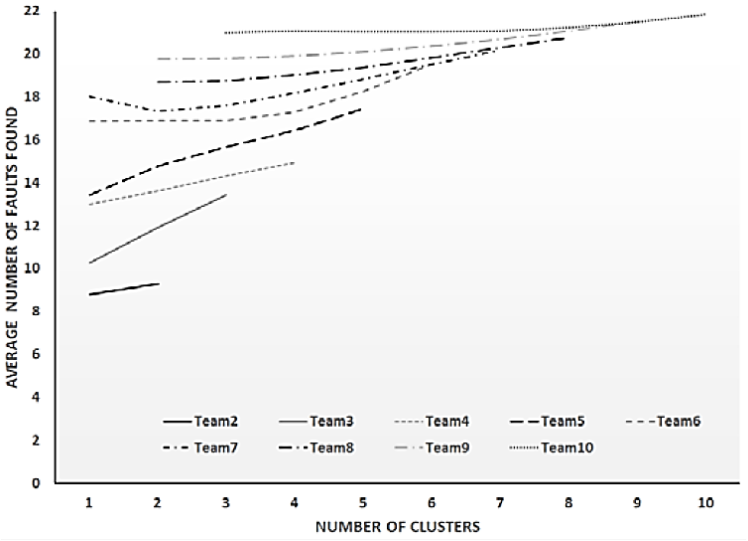


Figure 16. Inspection effectiveness of team size two to ten inspectors for study 4

The results show a consistent increase in the inspection effectiveness with an increase in the number of clusters used to form teams. We believe that the teams with more dissimilar LSs of inspectors have less fault overlap and consequently, their inspection effectiveness increases. To evaluate this, we performed a linear regression test to see whether the dissimilarity in the LSs of inspectors is positively correlated with the number of unique faults found by inspection teams of different sizes. The results (Table 9) show that, dissimilarity in the LS of inspectors had a strong and significant positive correlation with the team effectiveness for team size 3 to 10.

Table 9. Correlation between LS and inspection effectiveness for study 4

Team size	Correlation with the faults (within same team size)
3	$p=0.001$; $r^2=0.015$
4	$p=0.001$; $r^2=0.034$
5	$p=0.001$; $r^2=0.072$
6	$p=0.001$; $r^2=0.052$
7	$p=0.001$; $r^2=0.085$
8	$p=0.001$; $r^2=0.044$
9	$p=0.001$; $r^2=0.028$
10	$p=0.001$; $r^2=0.004$

For study 5, we followed meta-analysis approach. To provide an overview of results, Figure. 17 compares the inspection team *effectiveness* of inspection teams formed with similar LSs (left side of the scale of treatment effect) vs. dissimilar LSs (right side of the scale of treatment effect) of individual inspectors for inspection team size of 2 to 10 inspectors. For varying team sizes, forest plots for effectiveness describes the following parameters:

- *Hedges' g* denotes *effect size* (the difference between means divided by the standard deviation). For comparing results for each team, *effect size* measures the magnitude of the impact of LSs variation on team effectiveness.
- *95% Confidence Intervals (CI)* – describes the uncertainty associated with a *sampling method.* 95% CI of overall effect is measured across the width of diamond in plot.
- *p-value* denotes level of statistical significance for each team size and overall significance due to all team sizes.
- *Scale of treatment effect* (and line of no effect) - is denoted by center line. If width of diamond is towards right (doesn't overlap with center line), then results would be statistically significant for *dissimilar* teams.

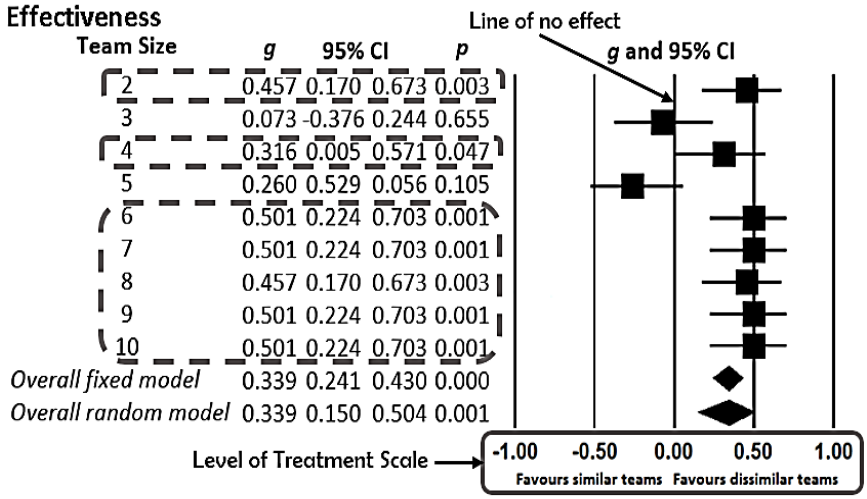


Figure 17. LSs dissimilarity vs. Inspection effectiveness for varying team sizes for study 5

Regarding *effectiveness*; the results in Figure. 17 show that, for team size 2, 4, and 6-10 inspectors; *dissimilar* teams found significantly more number of faults than *similar* teams. Since this analysis is based on the participants from a single study, the only variance is due to the participant variability. Therefore, *fixed effect* model (instead of random effect model) should be used to evaluate the overall effect of LS dissimilarity on the inspection team effectiveness. The overall effects (diamonds on the right side of the line of no effect) for effectiveness are also highlighted in Figure. 17. The 95% CI of the overall effect estimate (i.e., width of diamond) for larger teams does not overlap 0 (i.e., line of no effect). So, there is statistical significance (p=0.000 for effectiveness) towards the right side of treatment scale (i.e., favors dissimilarity of LSs).

For smaller teams, (e.g., 2 to 5), the effect of dissimilarity vs. inspection effectiveness is uneven. That is, team size of 3 and 5 favors LS *similarity* (left side of 0) and team size of 2 and 4 favor LS *dissimilarity* (right side of 0). Digging deeper, this was because inspectors are distributed into small number of clusters (e.g., three clusters for team size three) which doesn't always enable LSs disparity among inspectors resulting in random results. Therefore, based on these results, an inspection team size of 6-10 inspectors showed a significant improvement in fault detection *effectiveness* with an increase in LS disparity amongst individual inspectors.

For study 6 and 7, we combined their inspection results. Figure 18 compares the average number of unique faults detected (*effectiveness* – left side) and fault detection rate (*efficiency* – right side) for virtual inspection teams of size 2 to 10. Each line in the graph represents inspection effectiveness and efficiency for a team size. The number of clusters involved in team formation is always equal to team size. The more number of clusters involved, the more dissimilar a team for a particular team size would be (as it increases the LS disparity among

inspectors). For example, in team size 4, inspection teams created from a total of four clusters would be the most dissimilar team whereas, teams created from one cluster will be the team with inspectors of most similar LSs. An evident observation from the Figure 18, is that, effectiveness and efficiency of inspection teams increases with an increase in the LS disparity.

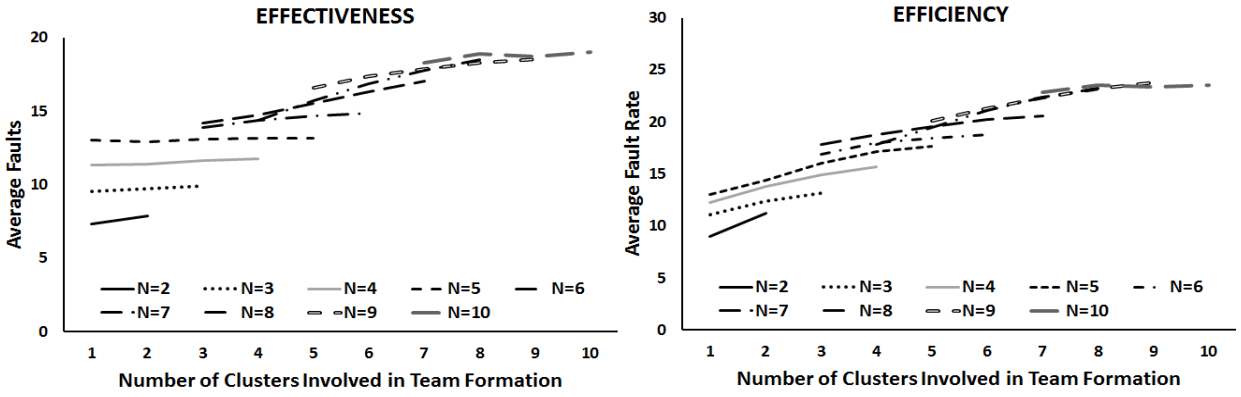


Figure 18. LSs dissimilarity vs. Inspection effectiveness and efficiency for team size 2 to 10 for study 6 & 7

To evaluate this effect, we performed regression analysis to see *whether dissimilarity in LSs yields higher fault coverage and higher fault rate for inspection teams?* The results in Table 10 shows that inspection performance of larger teams (i.e., N = 6-9 for *effectiveness* and N = 4-9 for *efficiency*) significantly favors LS dissimilarity amongst individual inspectors. For smaller team sizes (e.g., team size 3), inspection performance is not significant because smaller teams are created from less number of clusters (e.g., team size 3 has maximum 3 clusters) which does not lead to LS disparity among team members in dissimilar teams. Also, for larger team sizes (e.g., team size 10), the more number of cluster creation leads to the repetition of team members (due to 8 possible LS categories) with same LS preference which does not results in unique fault reporting.

Table 10. LS vs. Inspection performance of virtual team

Team Size	Effectiveness	Efficiency
2	p=0.294	p=0.006
3	p=0.072	p=0.100
4	p=0.017	p=0.011
5	p=0.086	p=0.002
6	p=0.025	p=0.042
7	p<0.001	p=0.002
8	p=0.001	p=0.001
9	p=0.003	p=0.001
10	p=0.215	p=0.259

Research [48] suggests that increasing the team size beyond a certain number of inspectors would not significantly diversify the LSs of inspectors in a team (due to 8 possible LS categories). Generally, companies do not employ a large inspection teams (which is the reason we had analyzed up to team of 10 inspectors). Based on this result, creating a team of 2 up to 10 number of inspectors based on the dissimilarity in their LS strengths (guided by the number of clusters involved in their formation) appears to increase the fault detection effectiveness during an inspection of requirements document. Based on these results, we can generalize that increasing inspectors (up to an extent) with diverse LSs in an inspection team do increase inspection effectiveness.

3.5.4. Are dissimilar teams more cost-effective as compared to random or similar teams?

This section utilizes results from study 1 for measuring cost effectiveness of inspection teams of varying LSs and of various sizes. For each team size (ranging from 2 to 9 inspectors), all possible virtual inspection teams were generated. The result was then sorted by teams with most dissimilar LSs (involved largest number of clusters) to the teams with most similar LSs (least number of clusters involved) of participating inspectors. Inspection data was calculated for

each of these virtual teams. The data consists of the unique number of faults uncovered and total time taken by the team members during the inspection of RIM document. Next, for each team size, from a pool of all possible virtual inspection teams (e.g., $11C_2$ for team size 2), 10 teams were selected from each set of LSs (i.e. for each team size, 10 teams of inspectors with dissimilar LSs, 10 teams of inspectors with similar LSs; 10 teams of inspectors randomly selected). For each team chosen, cost effectiveness (M_k) was calculated.

Step 1) Average cost to detect a fault in inspection (cr): adding all the faults found by inspectors, average number of faults found by an inspector is calculated. From the available values of time and average fault detected by an inspector, cr is calculated from the following equation:

cr = Cr / Dr (cost of inspection / total faults found during inspection)

- **"Cr - Cost of inspection"**, is calculated by total time spent by all inspectors during inspection.
- **"Dr"**, is the total number of unique faults found by all the inspectors during inspection.

Step 2) Virtual testing cost (Cvt): is the product of average cost to detect a fault in testing (i.e., ct) and the total number of faults present in the product (i.e., Dtotal).

- **"ct – Average cost to detect a fault in testing"**, is calculated as 6 times the average cost to detect a fault during the inspection (cr). The average cost to detect a fault in testing varies with the team size as it depends on the time taken and faults found by the inspectors. However, testing is independent of inspection and inspection team size, and considering all faults of the same severity, the average cost to detect a fault

in testing (ct) is kept constant for the evaluation regardless of the inspection team size.

- ***"Dtotal – Total fault count"***, is the total number of faults present in the document. The artifact used has 98 natural faults in it.

Step 3) Cost saved from inspection (ΔCt): testing cost saved by inspections is the product of unique faults found during inspection (Dr) and average cost to detect a fault in testing (ct). That is, **ΔCt = Dr * ct**.

The difference between the testing cost saved by inspection and cost spent on inspection provides the total reduction in costs. ***Kusumoto metric*** (M_k) is then obtained as follows:

M_k = Reduction is cost to detect all faults (i.e., ΔCt - Cr) /

Virtual testing cost (i.e., Cvt)----- Eq 3.5.4

The Mk value can range from -1 to +1. The M_k value of 1 means the most cost-effective inspection. A positive M_k value indicates that cost saved from inspection outweighs the costs spent on inspection. A negative M_k value indicates a cost ineffective process, and M_k value of 0 is when the inspection cost equals the testing savings.

Using Eq 3.5.4, we computed the M_k values for 10 virtual inspections for all team sizes (ranging 2-9 inspectors) and using all three means of team formation (dissimilar vs similar vs no LS preferences). As an example, 10 M_k values (representing 10 virtual inspections) for a team size of 6 inspectors (using the dissimilar LSs) as shown in Figure 19. Similar process was used to derive the M_k values for all the 10 virtual inspection teams by varying inspection team size (2-9) and for all three types of LS variations.

Virtual Inspection #	Total Defect Count (D_{total})	Defects Found (D_r)	Cost of Inspection (C_r)	Avg. Cost to detect a defect in Testing (C_t)	Testing Cost Saved by Reviews ($\Delta C_t = D_r * c_t$)	Virtual Testing Cost (C_{vt})	Kusumoto Metric $M_k = (\Delta C_t - C_r)/ C_{vt}$
1	98	51	550.20	64.73	3301.20	6343.48	0.433673
2	98	60	586.02	58.60	3516.12	5743.00	0.510204
3	98	54	466.02	51.78	2796.12	5074.44	0.459184
4	98	63	502.02	47.81	3012.12	4865.52	0.535714
5	98	48	502.02	62.75	3012.12	6149.75	0.408163
6	98	58	538.02	55.66	3228.12	5454.41	0.493197
7	98	49	481.02	58.90	2886.12	5772.24	0.416667
8	98	58	535.02	55.35	3210.12	5424.00	0.493197
9	98	47	501.00	63.96	3006.00	6267.83	0.399660
10	98	56	537.00	57.54	3222.00	5638.50	0.476190

Figure 19. Calculation of Kusumoto metric for team size 6

To provide an overview of our results, Figure 20 shows the average cost effectiveness (shown by lines) and average unique faults detected (shown by bars) for each inspection team size (ranging from 2–9 inspectors) for each team formation set (i.e. dissimilar, random and similar). Results are organized in the order of increasing team size. Left y-axis shows the average number of unique faults detected and secondary y-axis on the right shows the cost-effectiveness (M_k) of each team.

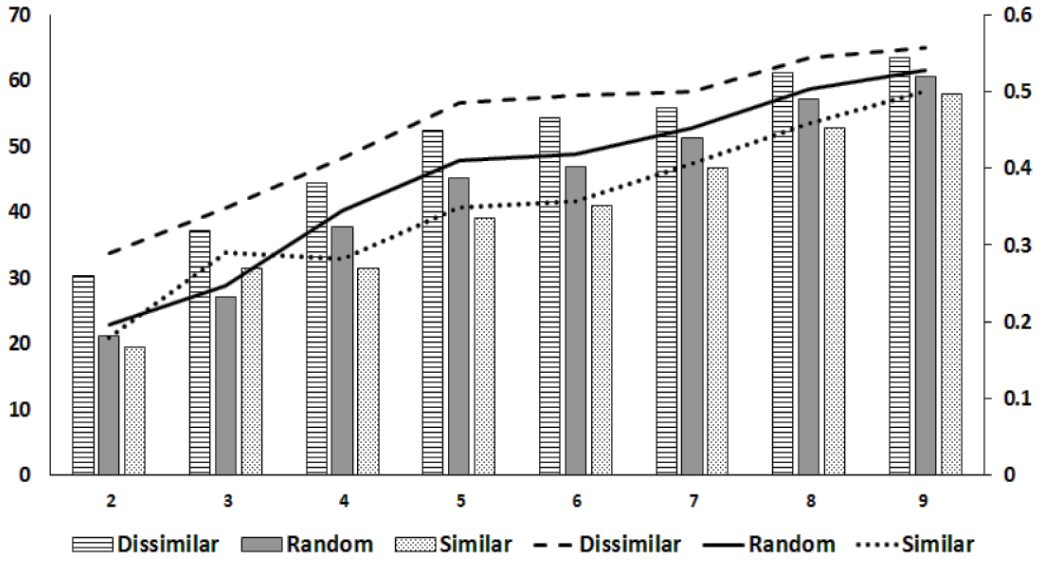


Figure 20. Comparison of LS on cost-effectiveness for team size 2 – 9

Cost effectiveness results show that inspection teams with dissimilar inspectors always found more number of faults and were more cost-effective as compared to teams of inspectors with similar or randomly selected. The results also show that teams whose members were randomly selected are more cost effective (except for team size 3) when compared to the team of inspectors with similar LSs. This is because, when randomly selected, there is still a higher chance of inspectors being dissimilar as opposed to intentionally selecting the more similar inspectors based on their LSs. Overall, dissimilar inspection teams were always cost effective across all team sizes as compared to random or similar teams. That is, more diverse inspectors uncovered the requirements from different perspectives and find a larger number of faults faster.

We performed a paired t-test to see whether inspection teams with dissimilar inspectors are significantly more cost-effective as compared to inspection teams of random or similar LSs. Results in Table 11 shows that dissimilar inspectors in an inspection team had a strong and significant impact as compared to inspection teams with inspectors of random (pair 1) and similar LSs (pair 2). Among random and similar (pair 3), teams with random LSs inspectors had strong and significant impact as compared to similar inspectors in a team which was due to the chance that some variation in the LS of the inspectors is achieved during the random selection of inspectors. Therefore, based on the results, forming inspection team with dissimilar inspectors significantly increased the cost-effectiveness.

Table 11. Paired sample test result

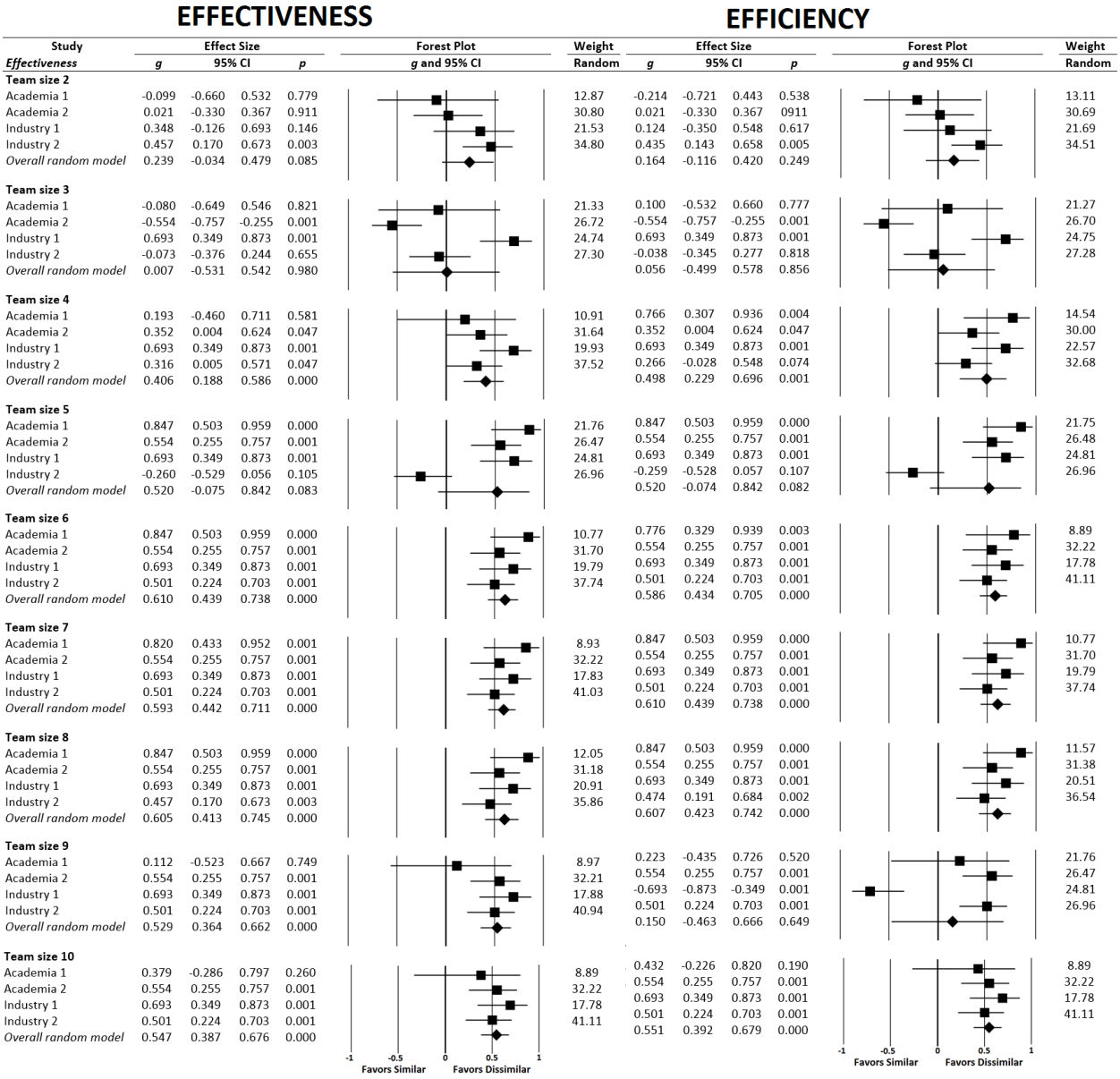
		Mean	Sig. (2-tailed)
Pair 1	Cost Dissimilar	**0.45414665**	<0.001
	Cost Random	0.38718237	
Pair 2	Cost Dissimilar	**0.45414665**	<0.001
	Cost Similar	0.35261604	
Pair 3	Cost Random	**0.38718237**	0.029
	Cost Similar	0.35261604	

Based on this result, creating a team of 2 up to 10 number of inspectors based on the dissimilarity in their LS strengths (guided by the number of clusters involved in their formation) appears to increase the fault detection effectiveness during an inspection of requirements document. Based on these results, we can generalize that increasing inspectors (up to an extent) with diverse LSs in an inspection team do increase inspection effectiveness.

3.5.5. Are results from previous studies validated when combined?

The LS scores of students (in studies 1 and 3) and industrial professionals (studies 4 and 5) were utilized to evaluate the impact of inspectors' LSs on their inspection team effectiveness and efficiency for teams ranging from size 2 up to 10 inspectors. To provide an overview of the results, the *effectiveness* results from study 5 are shown in Figure. 17 using a forest plot. To address heterogeneity and to generalize the results across four experiments, meta-analysis was performed that combined the results from all four studies and is shown in Figure. 21 (discussed later in this section). Figure. 21 shows the forest plots generated from four experiments for standardized effect of effectiveness and efficiency for inspection team size of 2 – 10 inspectors. The overall effect is also shown for each team size. Random effect model is more appropriate for deriving conclusions due to heterogeneity of population across four experiments. The thickness

of each block for each team size in the meta-analysis (Figure. 21) shows the relative random weight assigned to the result of each experiment by random effects model.



Study	Effect Size				Forest Plot	Weight	Effect Size				Forest Plot	Weight
Effectiveness	*g*	95% CI		*p*	*g* and 95% CI	Random	*g*	95% CI		*p*	*g* and 95% CI	Random
Team size 2												
Academia 1	-0.099	-0.660	0.532	0.779		12.87	-0.214	-0.721	0.443	0.538		13.11
Academia 2	0.021	-0.330	0.367	0.911		30.80	0.021	-0.330	0.367	0911		30.69
Industry 1	0.348	-0.126	0.693	0.146		21.53	0.124	-0.350	0.548	0.617		21.69
Industry 2	0.457	0.170	0.673	0.003		34.80	0.435	0.143	0.658	0.005		34.51
Overall random model	0.239	-0.034	0.479	0.085			0.164	-0.116	0.420	0.249		
Team size 3												
Academia 1	-0.080	-0.649	0.546	0.821		21.33	0.100	-0.532	0.660	0.777		21.27
Academia 2	-0.554	-0.757	-0.255	0.001		26.72	-0.554	-0.757	-0.255	0.001		26.70
Industry 1	0.693	0.349	0.873	0.001		24.74	0.693	0.349	0.873	0.001		24.75
Industry 2	-0.073	-0.376	0.244	0.655		27.30	-0.038	-0.345	0.277	0.818		27.28
Overall random model	0.007	-0.531	0.542	0.980			0.056	-0.499	0.578	0.856		
Team size 4												
Academia 1	0.193	-0.460	0.711	0.581		10.91	0.766	0.307	0.936	0.004		14.54
Academia 2	0.352	0.004	0.624	0.047		31.64	0.352	0.004	0.624	0.047		30.00
Industry 1	0.693	0.349	0.873	0.001		19.93	0.693	0.349	0.873	0.001		22.57
Industry 2	0.316	0.005	0.571	0.047		37.52	0.266	-0.028	0.548	0.074		32.68
Overall random model	0.406	0.188	0.586	0.000			0.498	0.229	0.696	0.001		
Team size 5												
Academia 1	0.847	0.503	0.959	0.000		21.76	0.847	0.503	0.959	0.000		21.75
Academia 2	0.554	0.255	0.757	0.001		26.47	0.554	0.255	0.757	0.001		26.48
Industry 1	0.693	0.349	0.873	0.001		24.81	0.693	0.349	0.873	0.001		24.81
Industry 2	-0.260	-0.529	0.056	0.105		26.96	-0.259	-0.528	0.057	0.107		26.96
Overall random model	0.520	-0.075	0.842	0.083			0.520	-0.074	0.842	0.082		
Team size 6												
Academia 1	0.847	0.503	0.959	0.000		10.77	0.776	0.329	0.939	0.003		8.89
Academia 2	0.554	0.255	0.757	0.001		31.70	0.554	0.255	0.757	0.001		32.22
Industry 1	0.693	0.349	0.873	0.001		19.79	0.693	0.349	0.873	0.001		17.78
Industry 2	0.501	0.224	0.703	0.001		37.74	0.501	0.224	0.703	0.001		41.11
Overall random model	0.610	0.439	0.738	0.000			0.586	0.434	0.705	0.000		
Team size 7												
Academia 1	0.820	0.433	0.952	0.001		8.93	0.847	0.503	0.959	0.000		10.77
Academia 2	0.554	0.255	0.757	0.001		32.22	0.554	0.255	0.757	0.001		31.70
Industry 1	0.693	0.349	0.873	0.001		17.83	0.693	0.349	0.873	0.001		19.79
Industry 2	0.501	0.224	0.703	0.001		41.03	0.501	0.224	0.703	0.001		37.74
Overall random model	0.593	0.442	0.711	0.000			0.610	0.439	0.738	0.000		
Team size 8												
Academia 1	0.847	0.503	0.959	0.000		12.05	0.847	0.503	0.959	0.000		11.57
Academia 2	0.554	0.255	0.757	0.001		31.18	0.554	0.255	0.757	0.001		31.38
Industry 1	0.693	0.349	0.873	0.001		20.91	0.693	0.349	0.873	0.001		20.51
Industry 2	0.457	0.170	0.673	0.003		35.86	0.474	0.191	0.684	0.002		36.54
Overall random model	0.605	0.413	0.745	0.000			0.607	0.423	0.742	0.000		
Team size 9												
Academia 1	0.112	-0.523	0.667	0.749		8.97	0.223	-0.435	0.726	0.520		21.76
Academia 2	0.554	0.255	0.757	0.001		32.21	0.554	0.255	0.757	0.001		26.47
Industry 1	0.693	0.349	0.873	0.001		17.88	-0.693	-0.873	-0.349	0.001		24.81
Industry 2	0.501	0.224	0.703	0.001		40.94	0.501	0.224	0.703	0.001		26.96
Overall random model	0.529	0.364	0.662	0.000			0.150	-0.463	0.666	0.649		
Team size 10												
Academia 1	0.379	-0.286	0.797	0.260		8.89	0.432	-0.226	0.820	0.190		8.89
Academia 2	0.554	0.255	0.757	0.001		32.22	0.554	0.255	0.757	0.001		32.22
Industry 1	0.693	0.349	0.873	0.001		17.78	0.693	0.349	0.873	0.001		17.78
Industry 2	0.501	0.224	0.703	0.001		41.11	0.501	0.224	0.703	0.001		41.11
Overall random model	0.547	0.387	0.676	0.000			0.551	0.392	0.679	0.000		

Figure 21. Meta-Analysis: comparing and combining results from four experiments

Based on the *overall effect* of results across four experiments as shown in Figure 21, major observations are listed as follows:

- Team sizes 4, 6-10 (regarding *effectiveness*) and team sizes 4, 6 to 8, 10 (regarding *efficiency*) showed significantly positive overall effect of LS dissimilarity on the inspection performance.
- The *effectiveness* results are consistent in experiment 2, 3, and 4 as compared to the results from experiment 1 (that contained 11 students). The relative weights assigned to each experiment (lower weight for experiment 1) also highlight the lack of consistency and preciseness in the results.
- *Overall* diamond's line (that also covers left side of line of no effect) for teams that are not significant (e.g. team size 2, 3, 5) shows that, though positively correlated, the particular team size does not always favor dissimilarity for effectiveness and efficiency of inspection.

Therefore, based on the results from all three experiments, it can be concluded that, "staffing inspection teams of 6-10 inspectors formed with varying LSs would results in detection of larger number of faults but would not always result in detection of faults faster".

3.5.6. Is overall inspection performance affected by the way inspectors read requirements document?

As mentioned in Section 3.4.1, participants inspected PGCS requirements and verbalized faults found while reading the document on a computer monitor. During this process, we also recorded their eye movements (to understand their reading patterns). Using the *fault and timing data* (to calculate fault count and fault rate) and *eye tracking data* (5 variables namely – $T_{fixation}$, T_{total}, S_{linear}, F_{ROI}, and T_{ROI} described in Section 3.4.5), we performed multiple regression with model selection (forward selection) to gain insights into the reading patterns of more *effective* (found more faults) and *efficient* (found faults faster) inspectors. Table 12 reports the results in

the form of correlation and p-values between 5 eye movement variables and 2 inspection performance variables.

Table 12. Eye tracking vs. Inspection performance

		T_{total}	$T_{fixation}$	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	0.18	0.083	-0.162	**0.294**	0.193
	p-value	0.137	0.308	0.163	**0.035**	0.120
Efficiency	Correlation	***-0.314***	***-0.361***	-0.104	-0.104	-0.209
	p-value	***0.026***	***0.012***	0.264	0.263	0.101

Major observations from Table 12 are listed below:

- In terms of inspection effectiveness, apart from linear reading motion (S_{linear}) all eye movement variables (T_{total}, $T_{fixation}$, F_{ROI}, and T_{ROI}) were positively correlated but ***F_{ROI}*** had a significant positive correlation. That is, participants who reported more faults, were able to quickly find inconsistencies at the area where faults were present (i.e. ROI) and then spent more time fixating on ROI's to detect and log faults present.
- Contrary to the effectiveness results, all eye movements were negatively correlated and two variables (T_{total} and $T_{fixation}$) had a significant negative correlation. That is, fixation on ROI's is more important than the total fixation (which in fact may be due to lack of information processing).
- Also, participants reading in a linear fashion, were negatively correlated with efficiency. Based on the participant behavior (observed by one of the researcher during experiment), some *sequential* readers tend to read and comprehend the entire information to understand the system first and then searched for faults which results in more inspection time.

Overall, effective inspectors tend to fixate more at the regions where faults exist to comprehend information instead of spending more time to fixate non ROI's on each page. This might be due to the fact that, during the inspection, effective inspectors find odds in reading NL requirements mostly at ROI's. Therefore, they fixate to find and report faults documented in the requirements at ROI's which led to significantly positive correlation between number of fixations at ROI (as opposed to the total fixation) and inspection effectiveness. Also, inspectors who spent more time fixating on each page were significantly less efficient because of the difficulty in processing requirements information. Following up on this result, in later sections, we present analysis of LSs of inspectors that exhibited such behavior (i.e., tendency to fixate on ROI's) to be make more informed decisions for selecting inspection team.

3.5.7. Does inspection teams, based on LSs have a particular reading pattern that impacts their inspection performance?

To find the impact of inspection teams based on their LS diversity and eye movements, the overall eye movement data for all virtual inspection teams in each team size was calculated. The eye movement data for each individual was averaged to calculate reading pattern of an inspection team of a particular size. We performed linear regression analysis to evaluate eye tracking factors that affects the high inspection performance of dissimilar teams of varying inspection team sizes. The results (Table 13) shows the linear regression output of eye tracking vs. LS dissimilarity data of team size 2 to 10. The shaded portion in Table 13 shows significant positive correlation wherein, bold with italics denote negative correlation. Major observations from Table 13 are listed below:

Table 13. Eye movement vs. Virtual inspection teams of size 2-10

Team Size	Fixation Time Per Page $T_{fixation}$	Time Per Page T_{total}	Total Fixations at ROI F_{ROI}	Total Duration at ROI T_{ROI}
2	***p=0.003***	***p=0.006***	*p=0.109*	***p=0.010***
3	*p=0.103*	*p=0.100*	*p=0.556*	***p=0.026***
4	***p=0.011***	***p=0.029***	*p=0.058*	***p=0.004***
5	***p=0.003***	***p=0.003***	***p=0.003***	***p=0.003***
6	***p=0.038***	***p=0.038***	p=0.038	p=0.038
7	p=0.129	p=0.074	p<0.001	p<0.001
8	***p=0.001***	***p=0.001***	p<0.001	p<0.001
9	***p<0.001***	***p<0.001***	p<0.001	p=0.001
10	p=0.198	p=0.198	p=0.198	p=0.198

- High-performing teams (i.e. team size 6 to 9) spent less time fixating per page ($T_{fixation}$) and overall spent less time on each page (T_{total}). This was especially true for larger teams;
- Conversely, fixating on ROI's (F_{ROI}) and total time spent at fault locations (T_{ROI}) resulted in increased inspection performance for larger team sizes (N=6 to 9).

Overall, if individual inspectors in a team of diverse LSs of inspectors naturally tend to spend more time processing information and extracting faults by fixating at the area where faults are present, it resulted in improved performance (effectiveness and efficiency) as well.

3.5.8. Does inspectors belonging to a particular LS category have a reading pattern which supports inspection outcome positively?

We also wanted to investigate the impact of eye movement of individual LSs category (e.g., ACT vs REF) on inspection outcome to gain more insights into how eye movement factors relates to different LS categories. Therefore, individual LS categories were separated from LS of all participants and results in individual LS categories consists of following number of

participants: ACT (4) vs REF (9), SEN (8) vs INT (5), VIS (10) vs VER (3), and SEQ (7) vs. GLO (6). All individual LS category data (inspection and eye tracking) was collected. Total fixations at ROI were converted as one tenth of the actual value ($F_{ROI}/10$) to accommodate all results in the graph. While comparing the eye movement performance of each LS category, it was found that, participants with ***ACT*** - Active LS (Figure 22) had the highest number of fixation time per page and time per page (which were shown to be negatively correlated to inspection performance in Section 4.5.6). Also, shown in Figure 22, participants with ***SEQ***- Sequential LS preference had maximum number of total fixation at ROI and total duration at ROI (that was positively correlated with inspection performance).

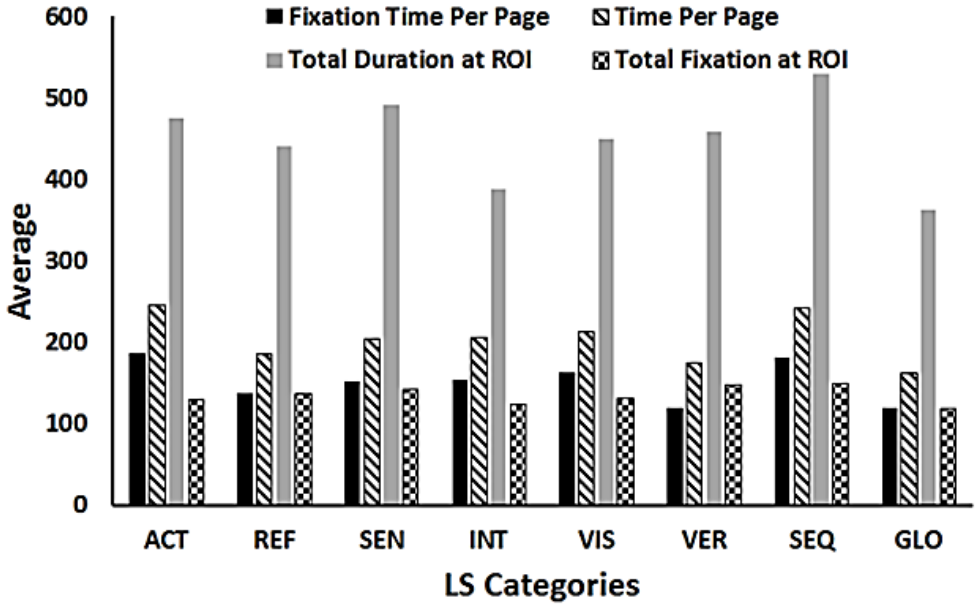


Figure 22. Comparison of eye movements of LS categories

To enable such comparison for each LS category, we performed multiple regression to find the eye movement factors in each LS category that impacts inspection performance (i.e. effectiveness and efficiency). The results are shown in Figure 23. The results (Figure 23) shows the following observations:

$T_{fixation}$ - *Fixation Time Per Page*, T_{total} - *Time Per Page*, S_{linear} - *Linear Saccade Per Page (in %)*, F_{ROI} - *Total Fixations at ROI*, T_{ROI} - *Total Duration at ROI*

	ACT	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	-0.624	-0.359	-0.822	-0.118	-0.102
	P-value	0.188	0.321	0.089	0.441	0.449
Efficiency	Correlation	*-0.997*	*-0.923*	-0.134	-0.199	-0.452
	p-value	*0.002*	*0.039*	0.433	0.401	0.274
	REF	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	0.305	0.371	-0.267	0.584	0.524
	P-value	0.213	0.163	0.244	0.049	0.074
Efficiency	Correlation	-0.368	-0.341	-0.012	-0.006	-0.074
	p-value	0.165	0.185	0.488	0.494	0.425

	SEN	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	0.660	0.533	0.053	0.845	0.831
	P-value	0.037	0.087	0.450	0.004	0.005
Efficiency	Correlation	-0.227	-0.380	0.780	0.261	0.171
	p-value	0.294	0.176	0.011	0.266	0.343
	INT	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	-0.674	-0.475	*-0.843*	-0.799	*-0.875*
	P-value	0.106	0.209	*0.036*	0.052	*0.026*
Efficiency	Correlation	*-0.881*	-0.748	*-0.815*	*-0.886*	*-0.967*
	p-value	*0.024*	0.073	*0.046*	*0.023*	*0.004*

	VIS	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	0.328	0.255	-0.028	0.720	0.689
	P-value	0.178	0.238	0.469	0.009	0.014
Efficiency	Correlation	*-0.485*	-0.573	0.351	0.098	0.003
	p-value	*0.078*	0.042	0.160	0.393	0.497
	VER	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	-0.170	0.396	*-1.00*	-0.179	-0.254
	P-value	0.446	0.370	*0.005*	0.443	0.418
Efficiency	Correlation	-0.665	-0.418	-0.857	-0.671	-0.727
	p-value	0.269	0.453	0.172	0.266	0.241

	SEQ	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	0.515	0.473	-0.105	0.848	0.847
	P-value	0.118	0.142	0.412	0.008	0.008
Efficiency	Correlation	-0.003	-0.054	-0.052	0.620	0.593
	p-value	0.497	0.454	0.456	0.069	0.080
	GLO	$T_{fixation}$	T_{total}	S_{linear}	F_{ROI}	T_{ROI}
Effectiveness	Correlation	-0.542	-0.239	*-0.754*	*-0.747*	*-0.856*
	P-value	0.133	0.324	*0.042*	*0.044*	*0.015*
Efficiency	Correlation	*-0.859*	-0.676	-0.373	*-0.926*	*-0.890*
	p-value	*0.014*	0.070	0.233	*0.004*	*0.009*

Figure 23. Multiple regression results comparing inspection performance of LS categories based on eye movement factors

- ***SEN*** LS category has the maximum number of significantly positive supporting factors of eye movement ($T_{fixation}$, F_{ROI}, and T_{ROI}) for high inspection effectiveness.
- ***SEQ*** category had the second highest number of eye movement factors (F_{ROI} and T_{ROI}) that significantly supports inspection effectiveness positively and are important in terms of high performance.
- For inspection efficiency, there was no significant positive factor in eye movement of different LS categories but participants who tend towards ***INT*** and ***GLO*** LS category had eye movement factors that significantly affects inspection efficiency in a negative manner.
- Interestingly, for ***VER***- verbal learners (conducive for reading a NL document like SRS); reading in a linear fashion is significantly negatively correlated to the overall inspection effectiveness (p=0.005).

Based on the results, it could be said that participants that tend towards ***Sequential*** learning (and ***Sensing*** and ***Reflective*** to some extent) are preferred for requirements inspections

by fixating on the ROI's. Therefore, a certain combination of LSs should be selected that would enable inspectors to focus on areas where faults may exist and enable enough diversity to be able to reduce overlap of faults and increase inspection performance.

3.5.9. What insights can be gained from the eye tracking and inspection results to help improve the readability of requirements development?

We also wanted to evaluate the general effect of looking back to previously read information (to search, comprehend or recall information) on inspector's ability to detect and report faults. To do so, we performed linear regression (Figure 24) between eye tracking variables related to page referencing (I_{count}, GD_{count}, FR_{count}, I_{time}, GD_{time}, and FR_{time}) and inspection outcomes (FC and FR).

	I_{count}		GD_{count}		FR_{count}		T_{count}		I_{time}		GD_{time}		FR_{time}		T_{time}	
	R	p-val	R	p-val	R	p-val	R	p-val	R	p-val	R	p-val	R	p-val	R	p-val
FC	0.062	0.707	0.190	0.247	0.196	0.231	0.210	0.200	0.045	0.786	0.192	0.242	0.097	0.555	0.173	0.292
FR	***-0.343***	***0.033***	*-0.246*	*0.131*	*-0.153*	*0.352*	*-0.303*	*0.061*	***-0.359***	***0.025***	*-0.263*	*0.097*	*-0.174*	*0.291*	***-0.324***	***0.044***

Figure 24. Page transition vs. Inspection performance

Major obervations from Figure 24 (highlighted cell denote significant results) are:

- Time spent referring back to previous sections of document (***I_{time}***, ***GD_{time}***, ***FR_{time}***, ***T_{time}***) had a positive correlation (though non-significant) with inspection effectiveness (i.e., FC). Considering the length and nature of requirements document, inspectors tend to refer back to either check their undestanding, or recall information.
- While all transition variables were negatively correlated with FR, the number of times (**I_{count}**) and time spent (***I_{time}***) going back to the *Introduction* section had significant (but weak) negative corerlation on inspection efficiency (FR). Also, participants that spent more time (***T_{total}***) going back to previous sections of document, their inspection efficiency was significantly hampered.

Further, we wanted to evaluate which section (and accompanying information) received maximum amount of recall. To find most frequently (number of times) and heavily (time spent) referenced sections, the results from paired t-test showed that subjects went back to the ***general description*** section at a significantly higher rate (both in terms of count: p=0.005 and time spent p=0.004) than the ***introduction*** and functional/non-functional ***requirements*** section. To help readers understand this result, the general description section was written in paragraphs, introduction section was mix of paragraphs and system diagram, and functional requirements were written in bullet points. When generated heat maps when going back (using EyeMMV toolbox and color band of blue, green, orange, yellow, red) for a sample of pages in each section (Figure 25), it was clear that description section received maximum fixations and the fixation duration (shown in red) from participants specifically due to the dense information. This result is consistent with previous eye tracking research [40] that individuals reading passages find it difficult to comprehend and retaining information and their frequency and duration of fixations increase (especially at the end of sentences as seen with general description section in Figure 25) with the increasing difficulty of passages written in NL.

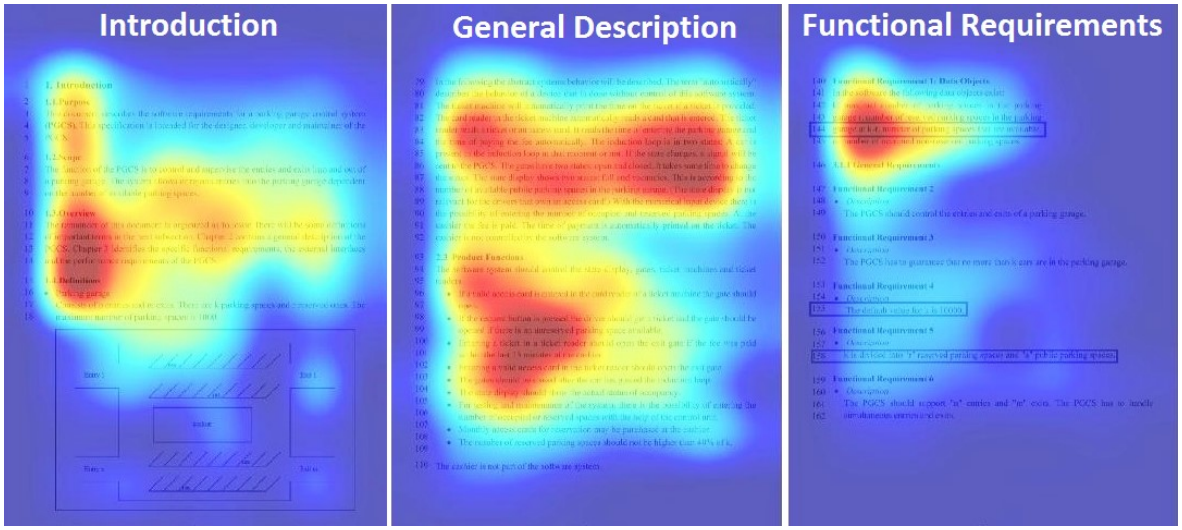


Figure 25. Heatmaps for introduction, general description and functional requirements page

Therefore, based on these results; to overcome the reference overhead, requirements should be written in a manner that participants don't have to spend more time in going back and searching for the relevant information to verify the correctness of functional/non-functional requirements. Adding pointers for the reference information may help reduce the reference overhead and improve readability of document under inspection.

3.5.10. How does the reading patterns of high-performing inspectors varies across different LS categories and dimensions?

We also investigated the relationships between the eye movements for inspectors belonging to different LS categories (e.g., SEN vs INT) and their inspection performance. Out of 39 participants, only 3 had any preference towards verbal LS. Therefore, we removed Visual/Verbal LS category from analysis and grouped participants into remainder of six LS clusters: ACT (17), REF (22), SEN (29), INT (10), SEQ (18), GLO (21). Next, we performed linear regression to evaluate relationships between eye movements and inspection performance for each LS category. The results are shown in Figure 26 & 27 and are discussed as follows:

- ***Sequential vs. Global***: Of all LS categories, ***SEQ*** has the maximum number of factors (***F_{ROI}, T_{ROI}, GD_{count}, FR_{count}, T_{count}, GD_{time}, FR_{time},*** and ***T_{time}***) that had a significant positive correlation with inspection effectiveness (FC). That is, inspectors with SEQ LS spent more time at fault areas (i.e. ROI's), spent more time referencing previously reviewed sections and subsequently found significantly more number of faults and found them faster.
- ***Sensing vs. Intuitive***: SEN LS category had the second highest number of factors (***GD_{count}, T_{count}, GD_{time},*** and ***T_{time}***) that supports inspection effectiveness positively.

But these learners spent more time fixating at regions other than ROI's (***T_{total}*** and ***$T_{fixation}$***) which negatively impacted their inspection efficiency.

- ***Active vs. Reflective***: REF learners saw a positive correlation between eye tracking variables and inspection effectiveness, however none of the relationship was significantly correlated.

		T_{total}	$T_{fixation}$	F_{ROI}	T_{ROI}	I_{count}	GD_{count}	FR_{count}	T_{count}	I_{time}	GD_{time}	FR_{time}	T_{time}
ACT	**R**	*-0.061*	*-0.175*	0.175	0.043	*-0.243*	*-0.111*	0.052	*-0.144*	*-0.279*	0.013	0.236	0.007
	p-val	*0.816*	*0.503*	0.503	0.869	*0.348*	*0.672*	0.843	*0.581*	*0.278*	0.960	0.361	0.979
REF	**R**	0.244	0.187	0.212	0.267	0.380	0.318	0.188	0.342	0.233	0.220	0.035	0.178
	p-val	0.273	0.404	0.344	0.230	0.081	0.150	0.401	0.119	0.296	0.325	0.876	0.427
SEN	**R**	0.305	0.357	0.003	0.038	0.257	**0.512**	0.278	**0.465**	0.234	**0.558**	0.134	**0.407**
	p-val	0.108	0.057	0.989	0.847	0.178	**0.005**	0.145	**0.011**	0.222	**0.002**	0.488	**0.028**
INT	**R**	*-0.203*	*-0.492*	*-0.280*	*-0.348*	*-0.336*	*-0.414*	*-0.034*	*-0.367*	*-0.337*	*0.397*	0.059	*-0.324*
	p-val	*0.574*	*0.149*	*0.434*	*0.324*	*0.342*	*0.234*	*0.925*	*0.297*	*0.341*	*0.256*	0.871	*0.361*
SEQ	**R**	0.456	0.436	**0.584**	**0.531**	0.219	**0.528**	**0.637**	**0.509**	0.275	**0.617**	**0.735**	**0.667**
	p-val	0.057	0.071	**0.011**	**0.024**	0.382	**0.024**	**0.004**	**0.021**	0.270	**0.006**	**0.001**	**0.002**
GLO	**R**	*-0.251*	*-0.409*	0.110	*-0.220*	*-0.175*	*-0.190*	*-0.108*	*-0.219*	*-0.233*	*-0.305*	*-0.147*	*-0.304*
	p-val	*0.272*	*0.066*	0.635	*0.339*	*0.449*	*0.410*	*0.642*	*0.339*	*0.308*	*0.178*	*0.526*	*0.180*

Figure 26. Comparison of eye tracking data and inspection fault count between ACT/REF, SEN/INT, and SEQ/GLO LS category

		T_{total}	$T_{fixation}$	F_{ROI}	T_{ROI}	I_{count}	GD_{count}	FR_{count}	T_{count}	I_{time}	GD_{time}	FR_{time}	T_{time}
ACT	**R**	***-0.668***	***-0.700***	0.277	0.133	***-0.586***	*-0.379*	*-0.120*	*-0.475*	***-0.590***	*-0.348*	0.047	*-0.402*
	p-val	***0.003***	***0.002***	0.281	0.611	***0.013***	*0.134*	*0.648*	*0.054*	***0.013***	*0.171*	0.858	*0.109*
REF	**R**	*-0.340*	*-0.370*	0.037	0.036	*-0.030*	*-0.159*	*-0.180*	*-0.189*	*-0.149*	*-0.255*	*-0.255*	*-0.327*
	p-val	*0.121*	*0.090*	0.870	0.873	*0.792*	*0.481*	*0.423*	*0.399*	*0.508*	*0.253*	*0.252*	*0.137*
SEN	**R**	***-0.428***	***-0.379***	0.251	0.182	*-0.222*	*-0.012*	*-0.123*	*-0.137*	*-0.235*	*-0.078*	*-0.149*	*-0.188*
	p-val	***0.021***	***0.042***	0.190	0.345	*0.247*	*0.951*	*0.526*	*0.479*	*0.220*	*0.688*	*0.441*	*0.328*
INT	**R**	***-0.640***	***-0.783***	*-0.383*	*-0.428*	*-0.586*	***-0.659***	*-0.256*	***-0.661***	*-0.324*	*-0.577*	*-0.278*	*-0.598*
	p-val	***0.046***	***0.007***	*0.275*	*0.217*	*0.075*	***0.038***	*0.476*	***0.037***	*0.054*	*0.081*	*0.437*	*0.068*
SEQ	**R**	*-0.327*	*-0.317*	0.084	*-0.133*	*-0.383*	*-0.162*	*-0.093*	*-0.156*	*-0.345*	*-0.065*	0.178	*-0.055*
	p-val	*0.186*	*0.199*	0.740	*0.598*	*0.116*	*0.521*	*0.715*	*0.509*	*0.161*	*0.797*	0.481	*0.827*
GLO	**R**	***-0.603***	***-0.664***	*-0.304*	*-0.409*	*-0.240*	*-0.301*	*-0.306*	*-0.426*	*-0.352*	***-0.437***	*-0.338*	***-0.539***
	p-val	***0.004***	***0.001***	*0.180*	*0.066*	*0.295*	*0.184*	*0.177*	*0.054*	*0.118*	***0.048***	*0.134*	***0.012***

Figure 27. Comparison of eye tracking data and inspection fault rate between ACT/REF, SEN/INT, and SEQ/GLO LS category

- ACT, INT, and GLO learners had a significant negative correlation with inspection efficiency and eye tracking variables. Inspectors belonging to these three clusters spent more time fixating throughout document (***T_{total}*** and ***$T_{fixation}$***) instead of ROI's which negatively impacted inspection efficiency. Additionally, consistent to the LS

theory, GLO learners took large jumps and INT leaners spent more time transitioning back and forth which significantly affected their inspection performance.

Based on these results, inspection performance was impacted by the ways participants read information contained across different sections of SRS document along with the effort spent to search or recall information during the inspection. Effective inspectors fixate at areas they find logical inconsistencies (at ROI) whereas underperforming inspectors make more regressive fixations due to comprehension problems (i.e., $\boldsymbol{T_{fixation}}$). This is especially true when trying to comprehend information written in a paragraph form (as seen in the ***General Description*** section of SRS document). Additionally, reading a document in small logical steps helps them detect more faults even if they switch between pages to comprehend information. This trend helps them to understand requirements better during requirements review and find odds at fault areas during inspections. Results also indicate that inspection efficiency could increase if requirements document could be structured in a manner that reduces the amount of transition overhead when referencing back to previous pages to search for certain information or to check their understanding.

4. RELEVANCE TO SOFTWARE ENGINEERING

Human learning preferences have been captured widely using FSLSM, but mostly in academia. Software engineering research validates that the overall inspection effectiveness decreases due to the overlap in the faults detected when inspectors are reviewing from same point of view. Results indicated that stakeholders involved during the requirement elicitation have different LSs and that the LSs of individual inspectors can have an impact on their inspection results. We believe that extending the concept of LS into software engineering context can be very effective in improving the quality of software artifacts. Software engineers vary in the way they "perceive" and "process" the information that is documented in software artifacts. This is especially true in context of NL requirements which involves collaboration between technical and non-technical stakeholders.

The concept of LS is applicable in software inspections domain and can help to manage the quality of software by creating high performance inspection team(s). Since each participant spent the same amount of time performing an inspection, we anticipate that the efficiency (faults found per time) results will follow the trend. The concept of using LS to form more effective software inspection teams is not just limited to the requirements phase, but rather applicable to the inspection of software deliverables produced at the other stages of the software development lifecycle (e.g., code, test plans etc.).

5. DISSEMINATION

The following are the list of publications that disseminated the work done during the dissertation. There are several papers under review and preparation that would broaden the dissemination of research done at NDSU.

Goswami, A., Walia, G., McCourt, M., and Padmanabhan, G. “Improving the Requirements Inspection Abilities of Computer Science Students through Analysis of their Reading and Learning Styles”, 124th Annual ASEE Conference, June 25 - 28, 2017, Columbus, Ohio

Goswami, A., Walia, G., and Rathod, U. "Using Learning Styles to Staff and Improve Software Inspection Team Performance", 27th IEEE International Symposium on Software Reliability Engineering, Ottawa, Canada (ISSRE 2016).

Goswami, A., Walia, G., McCourt, M., and Padmanabhan, G. "Using Eye Tracking to Investigate Reading Patterns and Learning Styles of Software Requirement Inspectors to Enhance Inspection Team Outcomes", In Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '16). ACM, New York, NY, USA, Article 34, 10 pages.

Goswami, A., Walia, G., and Singh, A. "*Using Learning Styles of Software Professionals to Improve their Inspection Team Performance*", Proceedings of the 27th International Conference on Software Engineering and Knowledge Engineering. July 6- 8, SEKE 2015 Pittsburgh, USA.

Goswami, A., and Walia, G. “*Using Learning Styles to Improve Cost Effectiveness of Software Inspection Teams*", Proceedings of the 26th International Conference on Software Engineering and Knowledge Engineering. July 1- 3, SEKE 2014 Vancouver, Canada.

Goswami, A., and Walia, G. "*An Empirical Study of the Effect of Learning Styles on the Faults found During the Software Inspection*", Proceedings of the 24th IEEE International Symposium on Software Reliability Engineering - ISSRE'2013, Research Track. Pasadena, CA, USA.

6. CONCLUSION AND FUTURE WORK

Based on our results, the concept of LS is applicable in software inspections domain and can help to manage the quality of software by creating high performance inspection team(s). The technique can be used by researchers and project managers to staff inspections depending on their need of overall fault coverage or to focus specifically on a particular type. Based on the results, we recommend the use of LSs concept for managing software inspections. We recommend having large number of inspectors in pool of inspectors from which inspection team will be selected that helps to select diverse inspectors for a team. For larger team sizes, higher the LSs disparity among individual inspectors, more effective the virtual inspection teams tend to be. Therefore, to deliver quality software product, manager should select inspection team size between 6 to 10 inspectors from the large pool of inspectors. We do not recommend inspectors with *Global* learning preference in academic inspections as they have a strong negative impact on fault detection effectiveness. But Global inspectors with a combination of *Reflective* and *Sensing* preference could be considered for inspections in industrial environment. Our results also showed that the dissimilarity in the LS of the participating inspectors had a direct and positive relationship with the cost effectiveness of inspection teams. This means, for an inspection team size, higher the dissimilarity in LS of inspectors more number of unique faults are found during the inspection of requirements. When the result was tested statistically, there was significant correlation between the LS dissimilarity and inspection cost effectiveness.

Results from our eye-tracking studies showed that effective inspectors tend to find more inconsistencies at the region where fault exists which was evident from the fixations at ROI's to comprehend information that leads to significantly positive correlation between number of fixations at ROI (as opposed to the total fixation) and inspection effectiveness. Also, inspectors

who face difficulty in understanding and processing information spend more time fixating on each page and have a significant negative impact on inspection efficiency. The results also showed that high performance inspection teams (i.e. with diverse members) actually fixate significantly at the fault area that leads to improved inspection performance. From different LS categories (ACT vs REF, SEN vs INT, VIS vs VER, and SEQ vs GLO), participants that had ***Sequential*** learning (and ***Sensing*** and ***Reflective*** to some extent) preference fixate more on the ROI's and hence, certain combination of LSs with LS diversity in teams would enable inspectors to focus on the fault areas where faults may exist, reduce overlap of faults, and increase inspection performance. The results from referring different sections (i.e. introduction, general description, and requirements) back during inspections showed that participants went back to the ***general description*** section at fixated more due to the paragraph nature of the section as compared to mix of paragraph and images, bullet points in other two sections (***introduction*** and functional/non-functional ***requirements***). Hence, it was concluded that requirements should be written in a manner (small logical steps) that reduces search overhead during inspections by adding pointers for the reference information that may improve readability of document under inspection.

Our future work involves replicating the study with software engineers in the industry which involves non-technical inspectors. Another future work is to replicate the study for inspection of other software artifacts (e.g., software design, test plan) and to select inspection teams based on LS of author of software artifact. The other future work is to evaluate the impact of LSs on an agile software development environment.

REFERENCES

[1] Aceituna, D. et al. 2011. Evaluating the use of model-based requirements verification method: A feasibility study. *Empirical Requirements Engineering (EmpiRE), 2011 First International Workshop on* (2011), 13–20.

[2] Aceituna, D. et al. 2014. Model-based requirements verification method: Conclusions from two controlled experiments. *Information and Software Technology*. 56, 3 (2014), 321–334.

[3] Ackerman, A.F. et al. 1989. Software Inspections: An Effective Verification Process. *IEEE Software*. 6, 3 (1989), 31–36.

[4] Agogino, a. M. and Hsi, S. 1995. Learning style based innovations to improve retention of female engineering students in the Synthesis Coalition. *Proceedings Frontiers in Education 1995 25th Annual Conference. Engineering Education for the 21st Century*. 2, (1995), 4a2.1-4a2.4.

[5] Albayrak, Ö. and Carver, J.C. 2014. Investigation of individual factors impacting the effectiveness of requirements inspections: a replicated experiment. *Empirical Software Engineering*. 19, 1 (2014), 241–266.

[6] Allert, J. 2004. Learning style and factors contributing to success in an introductory computer science course. *Proceedings - IEEE International Conference on Advanced Learning Technologies, ICALT 2004* (2004), 385–389.

[7] Anderson, T.W. 2009. *An Introduction To Multivariate Statistical Analysis, 3rd Ed*. Wiley India Pvt. Limited.

[8] Aranda, G.N. et al. 2005. A cognitive-based approach to improve distributed requirements elicitation processes. *Fourth IEEE Conference on Cognitive Informatics 2005, ICCI 2005* (2005), 322–330.

[9] Aurum, A. et al. 2002. State-of-the-art: Software inspections after 25 years. *Software Testing Verification and Reliability*. 12, 3 (2002), 133–154.

[10] Basili, V.R. et al. 1996. The Empirical Investigation of Perspective-based Reading. *Empirical Software Engineering*. 1, (1996), 133–164.

[11] Bavanari, H. 2012. *Software Inspection Team Formation Based on the learning Style of Individual Inspectors*.

[12] Bednarik, R. and Tukiainen, M. 2006. An eye-tracking methodology for characterizing program comprehension processes. *Proceedings of the 2006 symposium on Eye tracking research & applications - ETRA '06*. (2006), 125.

[13] Berry, D.M. 2008. Ambiguity in Natural Language Requirements Documents, Innovations for Requirement Analysis. From Stakeholders' Needs to Formal Designs: 14th Monterey Workshop 2007, Monterey, CA, USA, September 10-13, 2007. Revised Selected Papers. Springer-Verlag, Berlin, Heidelberg.

[14] Berry, D.M. and Kamsties, E. 2004. Ambiguity in requirements specification. *Perspectives on software requirements*. Springer. 7–44.

[15] Boehm, B. and Basili, V.R. 2001. Software Defect Reduction Top 10 List. *Computer*. 34, (2001), 135–137.

[16] Briand, L.C. et al. 2000. Assessing the cost-effectiveness of inspections by combining project data and expert opinion. *Software Reliability Engineering, 2000. ISSRE 2000. Proceedings. 11th International Symposium on* (2000), 124–135.

[17] Busato, V. V et al. 1998. The relation between learning styles, the Big Five personality traits and achievement motivation in higher education. *Personality and Individual Differences*. 26, 1 (1998), 129–140.

[18] Cao, J. and Nishihara, A. 2012. Understand learning style by eye tracking in slide video learning. *Journal of Educational Multimedia and Hypermedia*. 21, 4 (2012), 335–358.

[19] Carver, J. et al. 2003. Observational studies to accelerate process experience in classroom studies: An evaluation. *Proceedings - 2003 International Symposium on Empirical Software Engineering, ISESE 2003* (2003), 72–79.

[20] Carver, J. 2004. The impact of background and experience on software inspections. *Empirical Software Engineering*. 9, 3 (2004), 259–262.

[21] Carver, J. et al. 2008. The impact of educational background on the effectiveness of requirements inspections: An empirical study. *IEEE Transactions on Software Engineering*. 34, 6 (2008), 800–812.

[22] Chamillard, A.T. and Karolick, D. 1999. Using learning style data in an introductory computer science course. *ACM SIGCSE Bulletin* (1999), 291–295.

[23] Charkins, R.J. et al. 1985. Linking teacher and student learning styles with student achievement and attitudes. *The Journal of Economic Education*. 16, 2 (1985), 111–120.

[24] Chillarege, R. et al. 1992. Orthogonal defect classification-a concept for in-process measurements. *IEEE Transactions on software Engineering*. 18, 11 (1992), 943–956.

[25] Doolan, E.P. 1992. Experience with Fagan's inspection method. *Software: Practice and Experience*. 22, 2 (1992), 173–182.

[26] Fabbrini, F. et al. 1998. Achieving quality in natural language requirements. *Proceedings of the 11th International Software Quality Week* (1998), 4–5.

[27] Fagan, M.E. 1986. Advances in Software Inspections. *IEEE Transactions on Software Engineering*. SE-12, 7 (1986), 744–751.

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

[29] Felder, R. and Silverman, L. 1988. Learning and teaching styles in engineering education. *Engineering education*. 78, June (1988), 674–681.

[30] Felder, R.M. 2010. Are Learning Styles Invalid? (Hint: No!). *On-Course Newsletter, September 27, 2010*. (2010).

[31] Felder, R.M. and Spurlin, J. 2005. Applications, Reliability and Validity of the Index of Learning Styles. *International Journal of Engineering Education*. 21, 1 (2005), 103–112.

[32] Freimut, B. et al. 2005. Determining inspection cost-effectiveness by combining project data and expert opinion. *IEEE Transactions on Software Engineering*. 31, 12 (2005), 1074–1092.

[33] Friedman, P. and Alley, R. 1984. Learning/teaching styles: Applying the principles. *Theory into Practice*. 23, 1 (1984), 77–81.

[34] Fusaro, P. et al. 1997. A replicated experiment to assess requirements inspection techniques. *Empirical Software Engineering*. 2, 1 (1997), 39–57.

[35] Goldberg, J.H. and Kotval, X.P. 1999. Computer interface evaluation using eye movements: Methods and constructs. *International Journal of Industrial Ergonomics*. 24, 6 (1999), 631–645.

[36] Goswami, A. 2015. *Using Learning Styles to Create Virtual Inspection Teams: Technical Report*.

[37] Granka, L. a. et al. 2004. Eye-Tracking Analysis of User Behavior in WWW Search. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. (2004), 478–479.

[38] Hartigan, J.A. and Wong, M.A. 1979. A K-Means Clustering Algorithm. *Applied Statistics*. 28, 1 (1979), 100–108.

[39] Jolliffe, I. 2002. *Principal component analysis*. Wiley Online Library.

[40] Just, M. and Carpenter, P. 1980. A theory of reading: from eye fixations to comprehension. *Psychological review*. 87, 4 (1980), 329–354.

[41] Kane, M. 1984. Cognitive styles of thinking and learning part one. *Academic Therapy*. 19, 5 (1984), 527–536.

[42] Klecka, W.R. 1980. *Discriminant analysis*. Sage.

[43] Kolb, D. 1984. Experiential learning: Experience as the source of learning and development. *Journal of Organizational Behavior*. 8, 4 (1984), 359–360.

[44] Kusumoto, S. et al. 1992. A new metric for cost effectiveness of software reviews. *IEICE Transactions on Information and Systems*. 75, 5 (1992), 674–680.

[45] Laitenberger, O. 2002. A Survey on Software Inspection Technologies. *Handbook on Software Engineering and Knowledge Engineering*. 517–555.

[46] Leszak, M. et al. 2000. A case study in root cause defect analysis. *Proceedings of the 22nd international conference on Software engineering - ICSE '00*. (2000), 428–437.

[47] MacKay, D.J.C. 2005. *Information Theory, Inference, and Learning Algorithms David J.C. MacKay*.

[48] Mandala, N.R. et al. 2012. Application of kusumoto cost-metric to evaluate the cost effectiveness of software inspections. *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*. (2012), 221–230.

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

[50] McCarthy, B. 1987. *The 4MAT system: Teaching to learning styles with right/left mode techniques*. Excel, Incorporated.

[51] Mehigan, T.J. and Pitt, I. 2012. Detecting Learning Style through Biometric Technology for Mobile GBL. *International Journal of Game-Based Learning*. 2, 2 (2012), 55–74.

[52] Miller, J. and Yin, Z. 2004. A cognitive-based mechanism for constructing software inspection teams. *IEEE Transactions on Software Engineering*. 30, 11 (2004), 811–825.

[53] Montgomery, S.M. 1995. Addressing diverse learning styles through the use of multimedia. *Proceedings Frontiers in Education 1995 25th Annual Conference. Engineering Education for the 21st Century*. 1, (1995), 13–21.

[54] Myers, I.B. et al. 1985. *Manual: A guide to the development and use of the Myers-Briggs Type Indicator*. Consulting Psychologists Press Palo Alto, CA.

[55] Nalbant, S. 2005. *An Evaluation of the Reinspection Decision Policies for Software Code Inspections*. Middle East Technical University.

[56] Parnas, D.L. and Lawford, M. 2003. The role of inspection in software quality assurance. *IEEE Transactions on Software Engineering* (2003), 674–676.

[57] Parnas, D.L. and Weiss, D.M. 1985. Active Design Reviews: Principles and Practices. *Proceedings of the 8th International conference on Software engineering*. (1985), 132–136.

[58] Porter, A. et al. 1998. Understanding the sources of variation in software inspections. *ACM Transactions on Software Engineering and Methodology*. 7, 1 (1998), 41–79.

[59] Porter, A.A. et al. 1995. Comparing detection methods for software requirements inspections: A replicated experiment. *IEEE Transactions on software Engineering*. 21, 6 (1995), 563–575.

[60] Pressman, R.S. 2005. *Software engineering: a practitioner's approach*. Palgrave Macmillan.

[61] Russell, G.W. 1991. Experience with inspection in ultralarge-scale development. *IEEE software*. 8, 1 (1991), 25–31.

[62] Shull, F. et al. 2001. An empirical methodology for introducing software processes. *ACM SIGSOFT Software Engineering Notes*. 26, 5 (2001), 288.

[63] Shull, F. et al. 2000. How perspective-based reading can improve requirements inspections. *Computer*. 33, 7 (2000), 73–79.

[64] Solomon, B.A. and Felder, R.M. 1999. Index of learning styles. *Raleigh, NC: North Carolina State University. Available online*. (1999).

[65] Steinbach, M. et al. 2004. The challenges of clustering high dimensional data. *New Directions in Statistical Physics*. Springer. 273–309.

[66] Subramanian, G. et al. 2007. Software quality and IS project performance improvements from software development process maturity and IS implementation strategies. *Journal of Systems and Software*. 80, 4 (2007), 616–627.

[67] Tatsuoka, M.M. and Tiedeman, D. V 1954. Chapter IV: Discriminant Analysis. *Review of Educational Research*. 24, 5 (1954), 402–420.

[68] Tian, J. 2005. Software quality engineering: testing, quality assurance, and quantifiable improvement. *Quality Engineering*. 2, (2005), 1–52.

[69] Tinker, M.A. 1928. A photographic study of eye movements in reading formulae. *Genetic Psychology Monographs*. (1928).

[70] Torbick, N. and Becker, B. 2009. Evaluating principal components analysis for identifying Optimal bands using wetland hyperspectral measurements from the Great Lakes, USA. *Remote Sensing*. 1, 3 (2009), 408–417.

[71] Uwano, H. et al. 2006. Analyzing Individual Performance of Source Code Review Using Reviewers' Eye Movement. *Eye tracking research & applications (ETRA)*. (2006), 133–140.

[72] Yarbus, A.L. 1967. Eye movements and vision. *Neuropsychologia*. 6, 4 (1967), 222.

[73] Yusuf, S. et al. 2007. Assessing the comprehension of UML class diagrams via eye tracking. *IEEE International Conference on Program Comprehension* (2007), 113–122.

[74] 2008. EyeLink® User Manual. SR Research Ltd.

APPENDIX A. PRE-STUDY SURVEY

This is a pre-study questionnaire. Please fill information in the questions below.

*Required

1. Please enter your First name and Last name: *

   _____

2. Please rate your English Level Reading/Writing (*Mark only one oval).* *

	1	2	3	4	5	
Min	◯	◯	◯	◯	◯	Max

3. What is your previous experience with software development in practice? (Check the bottom-most item that applies.) *

   - ◯ I never developed software.
   - ◯ I have developed a software on my own.
   - ◯ I have developed software as a part of a team, as part of a course.
   - ◯ I have developed software as a part of a team, in industry one time.
   - ◯ I have worked on multiple projects in industry.

4. Duration of your work experience (as mentioned above. E.g., 4 years 9 months)? *

   _____ Years ____ Months

5. **Software Development Experience** *

   Please rate your experience with respect to the following 5 point-scale:

   1 = No experience; **2** = learned in class or from book; **3** = used on a class project;

   4 = used on one project in industry; **5** = used on multiple projects in industry.

- Experience writing requirements 1 2 3 4 5
- Experience interacting with user to write requirements 1 2 3 4 5
- Experience writing use cases 1 2 3 4 5
- Experience reviewing requirements 1 2 3 4 5
- Experience reviewing use cases 1 2 3 4 5
- Experience changing requirements for maintenance 1 2 3 4 5
- Experience with software inspection 1 2 3 4 5

6.Comments:

_____

_____

_____

_____

_____

APPENDIX B. FAULT REPORTING FORM

Full Name: _____ **Checklist Start Time and Date: _____**

Fault #	Page #	Requirement #	Fault Class	Description	Time Found