USING LEARNING STYLES OF SOFTWARE PROFESSIONALS TO IMPROVE THEIR INSPECTION PERFORMANCE: AN EMPIRICAL STUDY

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

By

Abhinav Singh

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

April 2016

Fargo, North Dakota

North Dakota State University
Graduate School

Title

Using Learning Styles of Software Professionals to Improve their Inspection Performance – An Empirical Study

By

Abhinav Singh

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

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Gursimran Walia
Chair

Dr. Kendall Nygard

Dr. Limin Zhang

Approved:

04/14/2016	Dr. Brian Slator
Date	Department Chair

ABSTRACT

In the IT industry, good requirements specification plays a vital role in software projects success. Researches revealed that early detection of faults in a requirements document saves significant amount of rework. To achieve that, project managers employ inspectors to review and find faults in software requirement specification (SRS) document. Previous researches investigated that factors like educational background and work experience could impact the effectiveness of individual inspectors, with limited success. This research hypothesizes that Learning Style (LS) of individuals can be utilized to improve the performance of individual inspectors and inspection teams. The LS refers to the ways an individual processes or perceives any given information. LS and inspection data of participants were collected to investigate the effectiveness of inspection teams with different LS categories. Results show the teams of participants with most dissimilar LS categories found the largest number of faults compared to the teams with similar LS preferences.

Keywords— SRS; inspection; learning style; requirements

ACKNOWLEDGEMENTS

Sincere gratitude is hereby extended to Dr. Gursimran Walia who never ceased in helping until this report paper was structured and finalized. I appreciate the time, support, guidance and patience for the development and completion of this research paper. He has always motivated me and helped me at every step starting from writing the proposal to finalize the paper. He always answered all my questions with detailed and precise guidance and feedbacks I needed.

I am extremely grateful to Ph.D. student, Mr. Anurag Goswami, NDSU CS Department, a major contributor and author of the previous study that motivated this study. I am grateful for his time and effort he spared in assisting me with data understanding and analysis.

I thank, Dr. Kendall Nygard, Professor of Computer Science and Operations Research at North Dakota State University, for his time and to be a members of my supervisory committee. Also, for his continuous guidance and the compassion he had shown throughout my Master degree program that helped me immensely to achieve my goals.

I am grateful and appreciate Dr. Limin Zhang for her consideration and taking out time from her busy schedule to be a part of my the supervisory committee and showing interest in my research work.

A special thanks to the faculty of Computer Science department for all their help and support that was necessary at all the time throughout my program.

I also appreciate the individuals who generously participated and their time and experience for the purpose of this study.

I am grateful to my friends Tanveen, Saurabh and Jenifer for every step they have taken to make this all possible, to invest in me and motivate me with all the guidance I needed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

IT……..	Information Technology
LS…….....	Learning Styles
FSLSM…….....	Felder Silverman Learning Style Model
KLSI…….....	Kolb Learning Style Inventory
LSQ …….....	Learning Styles Questionnaire
ILS…….....	Index of Learning Styles
NL …….....	Natural Language
CA …….....	Cluster Analysis
DA……..……...	Discriminant Analysis
IV…….....	Independent Variable
DV…….....	Dependent Variable
PC…….....	Principal Component
PCA…….....	Principal Component Analysis
O…….....	Omission
A …….....	Ambiguous Information
II …….....	Inconsistent Information
IF …….....	Incorrect Information
E…….....	Extraneous
M…….....	Miscellaneous
GM…….....	Group Membership

1. INTRODUCTION

In the IT industry, failure rate of software development projects has been a major concern and multiple studies have been conducted to address the issue [1], [2], [3], [4]. Majority of these studies have focused on finding a comprehensive list, a scheme or a framework to determine the critical factors responsible for the success or failure of IT projects [2], [3], [4]. Software development involves a structured set of activities wherein requirements are gathered and analyzed followed by construction and testing of software product. Requirement elicitation is a critical phase that is performed at the beginning of the software development process. Empirical evidence has shown that finding and fixing faults in requirements document found at earlier stages saves significant amount of rework and resources [7]. According to Bob Lawhorn "60% - 80% of project failures can be attributed directly to poor requirements gathering, analysis and management [5]. Software requirements document, known as *SRS (Software Requirement Specification)*, is created by requirement engineers through discussion with broad spectrum of stakeholders and is documented using *Natural Language (NL)* to be able to communicate the requirements to the stakeholders. Use of NL makes the requirements document highly prone to faults due to imprecision, ambiguity and vagueness in NL language. A *fault* is an instance where a software requirement does not satisfy quality properties (e.g., ambiguity, incompleteness, inconsistency). Among different approaches used for detecting NL requirement faults (e.g. NL to State transitions [9], checklist based inspections [10], scenario based reading [11], ad hoc inspections [12]), *software inspections* are widely recognized as most effective. Software inspection is a technique first introduced by Michael Fagan [13]. Since then, it has been used widely to detect faults in software artifacts such as requirement specifications, development code, design documents and others software artifacts produced during the software development

process [14]. Fagan emphasizes (and other studies also show) that individual review of a phase is more effective than a team meeting review [10], [15]. Software Inspections' primary purpose is to find faults in the software so later activities are not contaminated [16]. Faults found at later stages make it difficult to deliver the product leading to project failures. To improve software quality, researchers and practitioners have collaborated to develop techniques that can find and fix faults committed during the requirements development [17].

Many researches [18], [19], [20] have been conducted to optimize effectiveness of inspection process to find defects accurately in the software artifacts (SRS or design document). Some research work have suggested to improve defect detection technique whereas other studies suggested an approach to improve inspection performance by adding diversity within the group of inspectors by using cognitive-based team selection [18], [19], [20]. To introduce cognitive-based heterogeneity in a team, our study focuses on using LS concept and investigates the team performance based on heterogeneity.

Felder and Silverman utilized *Learning Styles* (LS) [21], [22], a cognitive psychology approach. An individual's LS can be defined by the way that *individual acquires, retains and retrieves information*. This concept has been validated in educational research, which showed that students are better able to understand information in a manner conducive to their learning style. This research tries to apply the LS concept to improve the quality of software artifacts produced during the software development. Goswami and Walia had previously shown that LS concept could be used to improve inspection team performance through a controlled empirical study conducted with students at North Dakota State University (NDSU) [15]. During the study, LS of students enrolled in undergraduate and graduate software engineering classes were gathered and manipulated to evaluate their team performance while inspecting a SRS document.

The preliminary results from the study showed that, LS's can used to create high performing inspection teams (i.e., teams that find a larger number of faults) by enabling a larger disparity in the LS's of individual inspectors included an inspection team.

To be able to generalize these results, it's important that the research should be conducted in real world software project settings. Therefore, this study replicated the previously conducted study at NDSU, in industrial setting with software development professionals. This paper presents the empirical study wherein we investigated the effect of LS's of inspectors on the effectiveness of the inspection team inspecting a software requirement document. The study was conducted with software professionals working in different divisions of a medium size software organization. Nineteen software professionals working in different work environments participated in the study and had an average of 3 years industrial experience. LS's of participants were gathered, and each participant individually inspected the same requirement document using the fault-checklist technique to find faults [23], which were recorded in a standardized fault report form. To analyze the effect of LS's on the participants' inspection performance, virtual teams of size 2 to 10 were created by varying the disparity in the LS's of individual inspectors. The results show that teams consisting of inspectors with dissimilar LS's performed significantly better and were more effective than those with similar LS's. The results also provided additional insights into the LS's with respect to the individual inspection performance and are discussed in more details in Section 4.

The remainder of the paper is outlined as follows. Section2 explains the previous work done in the field of LS and also provides insights into LS concept and multivariate statistical techniques like principal component analysis, cluster analysis and discriminant analysis that were used to analyze the collected data. Section 3 describes the experiment design to evaluate the

effect of LS of individuals on team performance in finding faults from the requirements document. Section 4 describes the data analysis and results. Section 5 lists the threats to validity of the results. Section 6 summarizes the paper and evaluation results. Section 7 concludes the study and the scope of future work to extend this study.

2. BACKGROUND

This section provides information on previous LS models. Various LS models were developed to determine learning style characteristics of an individual. These characteristics are then used to develop teaching methods that improve the effectiveness of instructions given to students. We can infer from previous researches that this concept could be utilized in software engineering context. More research is required to be performed in this area. Study at NDSU among the students was to investigate how LS preferences of inspectors could impact the effectiveness of team performance in finding faults during requirement document inspection [15]. To investigate and verify the results of the study at NDSU, we performed a similar study in industrial settings where the set of participants was replaced with the IT professionals working on real-time projects.

2.1. Learning Styles Concept and FSLSM Model

This section explains the concept of LS dimensions and its categories. Individuals respond best to the mode of instructions that suits their LS. To figure out which mode of instruction is optimal, it's important to assess their learning style [23]. Researchers have worked to develop a technique to measure LS of individuals [24]. Kolb [25], [26] was the first to be credited for the measurement of LS in 1984. Below is the list of several models developed to measure the LS of individuals based on different criteria:

2.1.1. Kolb Learning Style Inventory (KLSI)

ELT – Experiential Learning Theory described as "the process where knowledge is created through the combination of grasping and transformation of experience" [25]. ELT model is based on two dialectically related modes of grasping experience (Concrete Experience (CE) and Abstract Conceptualization (AC)) and transforming experience (Reflective Observation

(RO) and Active Experience (AE)). As per ELT, this portrayal touches all the bases for learning. To investigate ELT, Kolb came up with an instrument called Learning Style Inventory (LSI). Below is a brief details from the observation based on KLSI:

Table 1: KLSI observation based on ELT portray

ELT Portrayal of learning abilities	Personality Style (used in KLSI)	Description
CE-RO	Diverging Style	work in groups, listening to different views with open mind, receiving personalized feedback
AC-RO	Assimilating Style	readings, lectures, exploring analytical models, having time to think things through
AC-AE	Converging Style	experimenting new ideas, simulations, lab assignments, practical applications
CE-AE	Accommodating Style	work with others, set goals, do field work, test different things to complete a project

2.1.2. Peter Honey and Alan Mumford's Model (Learning Styles Questionnaire (LSQ)) [27] [28]

LSQ was developed around Kolb's (1984) model and four types of learners were identified. It is expected that individuals naturally rely on one of these styles to learn or do an activity. These four styles are as follows:

- Activists: try anything first, tends to revel in short-term crisis, enjoy new experiences, make decisions intuitively, dislike structured procedures
- Theorists: focus on ideas, logic, systematic planning but do not believe on intuition or emotional involvement
- Pragmatists: practical, group-work, debate, risk takers but avoid deep level of understandings

- Reflectors: observe and describe processes, try to predict outcomes, focus on reflecting and trying to understand meaning

2.1.3. Dunn and Dunn Model

As per this model, student learning could be improved based on the individual's learning style and instructional environment. This model was developed to improve the effectiveness of instruction particularly for middle school students who are having academic difficulties [29]. There are 21 unique elements divided into 5 stimuli categories [29]. Below is the table giving brief insight into those elements

Table 2: Dunn and Dunn learning style model

Stimuli Type	Description/Preferences
Environmental	Where do learners prefer to learn -light, temperature, noise, seating design are explored
Emotional	Preferences for motivation, persistence, and responsibility and structure are examined
Sociological	Preferences for learning alone, in pairs, in a team, with or without an authority figure are examined
Physiological	Preference in perceptual strengths (visual, auditory, or kinesthetic), energy levels, preferences for intake and mobility are examined
Psychological	Preference in processing and responding to information, or impulsive or reflective, or global or analytic are surveyed

2.1.4. Felder Silverman Learning Style Model (FSLSM)

The concept of LS has been used in researches in education field to improve learning techniques and Felder validated the use of LS's in engineering education [23], [30], [31].

Felder and Silverman's Learning Style Model (FSLSM) introduced an instrument called Index of Learning Styles (ILS), which is widely used by researchers to measure the LS's [30], [31]. FSLSM is different from LSI and LSQ approaches. In FSLSM, a set of four dimensions is

used to classify the individuals based on their strengths, and the way they "perceive" and "process" information. Perceiving information is categorized under two dimensions – Sensing/Intuitive and Visual/Verbal. Processing information is categorized under other two dimensions – Active/Reflective and Sequential/Global. Below (Table 3) is a brief description about these dimensions:

Table 3: Index of Learning Styles

Felder-Soloman Index of Learning Styles	
Sensing	Intuitive
oriented towards facts, concrete data, careful with details, follow existing ways	abstract, conceptual, innovative, oriented towards theories and meaning, discovering possibilities
Visual	Verbal
prefer visual representations of material – pictures, diagrams, flow charts, video, demonstration	prefer written/spoken explanations
Active	Reflective
learn by trying things out, working in groups, discussing, explaining, brainstorming	learn by thinking things through, working alone, writing summaries
Sequential	Global
linear, orderly, learn in small logical steps	holistic, context and relevance of the subject, learn in large jumps

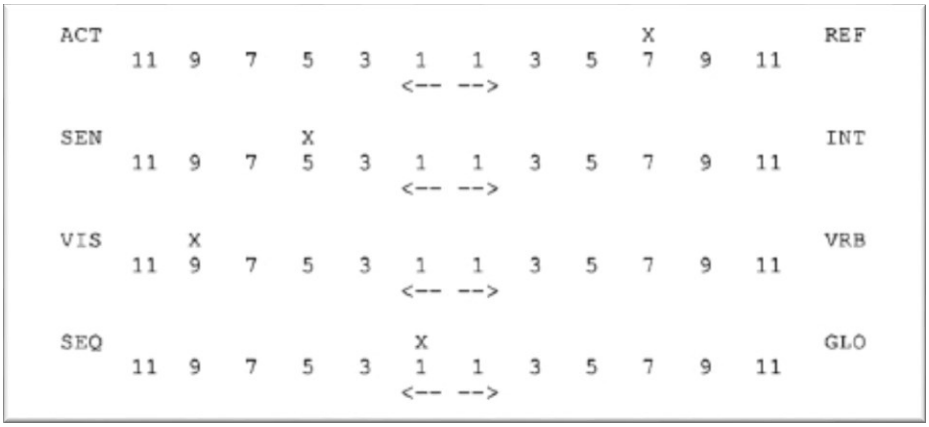

Figure 1: ILS result sample

The LS of an individual was measured using ILS (Table 3) over these four dimensions (see Fig. 1). ILS has been approved empirically for its reliability and validity [30], [31]. Using ILS, the LS of an individual was calculated based on the score received in a questionnaire having 44 questions. In ILS survey, questions are divided into four parts, 11 questions per LS dimension. LS score was calculated from the questionnaire results from the actual scores obtained in each category of the dimension. If a person answered 11 questions for a dimension out of which 5 answers were favoring Active category and 6 towards Reflective category then the resultant score would be 1 (i.e. 6-5) towards reflective on the scale. The actual score for each category in the above example is 5 for Active and 6 for Reflective category. The 'x' in Fig 1 illustrates the resultant score showing the characteristic strength towards a category in LS dimension. Individuals having a score from 1 to 3 on ILS are considered balanced, with preference towards both categories. However, resultant score from 7 – 9 is considered that a person has moderate to high inclination towards one category in a LS dimension respectively.

After careful analysis of different LS models, we concluded that ILS model of FSLSM is most appropriate to collect the LS scores of individuals.

2.2. Using LS Concept in Software Engineering

In previous researches, FSLSM has been widely used in the field of education to determine LS of individuals [21], [22], [24]. However, it lacks work in in improving software quality and reliability at an industrial level. It has been proven that reviewing and finding non-overlapping faults in any artifact with same point of view is less effective than with different [11]. Reviewing software artifacts presents the same problem and needs to be addressed as it can improve effectiveness of inspection process. Using the concept of LS in software industry is a novel idea. As mentioned before, this study is the replication of the study that was conducted at NDSU to examine the effect of LS on finding faults in SRS at academic level. In the IT industry, each software engineer/analyst has different ways to 'perceive' and 'process' information from the documented artifacts. This characteristic can be utilized to review the documents by different inspectors. This issue was addressed by using LS concept while inspecting the software artifact (SRS document) with a team of individuals consisting of dissimilar LS preferences, which resulted in finding more non-overlapping faults.

2.3. Multivariate Statistical Techniques

Multivariate analysis is used to find relationships among multiple independent (predictor) and dependent variables (outcome or phenomenon of interest) in a study [32]. The commonly used techniques for this are multivariate analysis of variance, Factor Analysis (or Principal Component Analysis), and Prediction of Group Membership (like discriminant analysis).

Factor analysis is used to determine underlying structures of a data set to develop or test a theory [33]. When independent variables (IVs) are highly correlated Principal Components (PCs) are used to reduce the number of independent variables (IVs) by grouping together the related

variables (that share common variance based on basic underlying factors of IVs) [33]. We used PC analysis (PCA) technique to analyze LS's of individuals. Section 2.3.1 describes PCA in detail. Cluster analysis (CA) is used for grouping a set of objects so that there are more similar (in some context) objects in the same group called a *cluster* than the other. In section 2.3.2, we discuss how we used cluster analysis for determining clusters of participants with similar LS's. Discriminant analysis (DA) is another technique that utilizes dependent variable (DV) along with IVs. It identifies a combination of quantitative IVs that predicts the group membership defined by a single DV with two or more categories e.g. it identifies the type of LSs that has most positive effect on the team performance [DV]. Section 2.3.3 provides further insight in DA technique to calculate the probability of a participant belonging in a cluster.

2.3.1. Principal Component Analysis

PCA is used for data analysis, to convert scores of all LS categories (correlated variables) into uncorrelated variables. It identifies the dominant pattern in the data matrix and prepares a set of score and loads the plot based on it [34]. For this study, data matrix was prepared by following FSLSM (explained in section 2.1.4) classification of LS dimensions in terms of two categories (sensing/intuitive, visual/verbal, active/reflective and sequential/global). These two categories from each dimension of LS are negatively correlated. PCA was used to understand the correlation between two categories for all four dimensions and between all four dimensions for each individual. It also helped us to plot the graph of relationship between the categories, dimensions and individuals by transforming original correlated data (sample data of one individual in Fig. 1) into linearly uncorrelated variables called Principal Components (PCs) [35], [36]. Number of PCs is always less than or equal to the number of original variables [35]. Each PC on the data matrix has certain variance among categories in each dimension and among all

four dimensions. PCA identifies the dominant pattern of that variance for each PC starting with maximum possible variance (PC1) going down to least possible variance (PC2) which could not be explained by PC1 and so on. Hence, it requires all PCs (which could be less than original number of variables (as mentioned before)) to cover total variance [36].

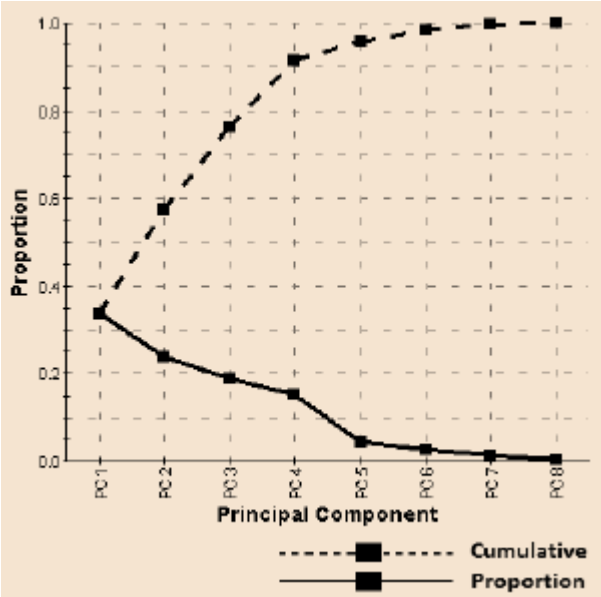

Figure 2: Plot of Principal components

The scree plot (Fig. 2), is a simple line segment plot showing total variance in the original data which is covered by respective PCs. Variance covered by each PC is shown by solid line and dotted line illustrates the cumulative variance (covered by respective PC). PC1 shows the maximum variance covered and PC8 shows the lowest (variance not covered in original data is close to zero). PC1 (~35%) and PC2 (~25) itself covers 60 % of the cumulative variance.

2.3.2. Cluster Analysis

This section explains what cluster analysis (CA) is and how we utilized it to group participants having similar LS's into different clusters. CA divides data into groups (clusters) for the purposes of summarization or improved understanding [37]. The goal is to have similar

objects into one group and different objects in other groups [38]. As mentioned before, in our research we used CA to make clusters of individual based on their LS resulting in clusters with high similarity within each cluster and high dissimilarity among different clusters [37].

There are many clustering techniques developed over the period of time (hierarchical (nested), partitional (un-nested) clustering etc.). For this research, we have used K-means clustering method, a prototype based, partitional clustering technique that attempts to find a user-specified number of clusters (K), which are represented by their centroids [38] [39]. Basic K-means algorithm can be explained as follows [38]:

1. Select K points as initial centroids
2. Repeat
3. Form K clusters by assigning each point to its closest centroid.
4. Re-compute the centroid of each cluster
5. Until Centroids do not change

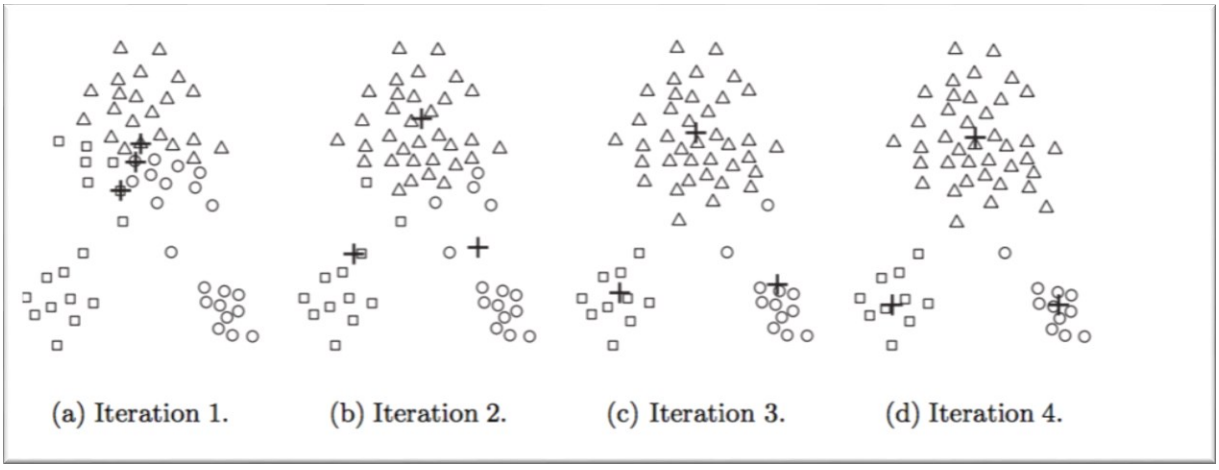

Figure 3: Clustering of different entities using K-Means algorithm

To explain clustering process better, the triangle, circle and square shaped blocks (fig. 3) represent subjects grouped in three different clusters (notice '+'). Each cluster exhibits

individuals with more similar characteristics as compared to other clusters. K initial centroids (user specified parameter) gets assigned to each collection of data points (named clusters), then the centroid of each cluster is then updated repeatedly based on the data points assigned to each cluster. This happens until the data points in clusters do not move from its position and the centroid remains the same [38] i.e. minimize the mean squared distance from each data point to the nearest centroid (fig. 3). To get to the mathematical calculations of how points are assigned to the closest centroid, we consider the data in Euclidean distance.

In this research, after collecting the data points for the LS's for each individual, we assigned centroids to make clusters and the process was repeated until we get the clusters of individuals with most similar LS's. These clusters helped us in finding the relationship between LS of team members with dissimilar to similar LS preferences. The teams formed from different cluster members helped us to create dissimilar LS groups and the teams formed from same cluster helped us to get groups with similar LS preferences.

2.3.3. Discriminant Analysis

This section describes discriminant analysis (DA) and how it's used to create participants clusters with dissimilar LS and order them in each cluster. DA is a multivariate statistical method to predict group membership based on the set of IVs (independent variables) [35], [39]. It tries to interpret pattern of differences among the IVs and thereby produce several sets of IVs. This is done by creating a linear combination of IVs to discriminate among groups. The IV sets are then used to predict group membership [33]. This can be seen as the opposite of MANOVA (*Multivariate ANalysis Of Variance* – identifies group differences on a combination of quantitative DVs). DA derives the combination of IVs to best predict the group membership as defined by one dependent variable (DV) with two or more categories. Categories are defined

based on factors such as - which LS preferences of individuals (similar or dissimilar – IVs) distinguish between improved performances (DV) [33]. DA is the best concept to be applied for this research after performing cluster analysis since the groups were formed based on similar or dissimilar LS preferences. In this study, LS variations were partitioned into a "between group" (dissimilar LS) and a "within-group" (similar LS) component. DA provided Group Membership (GM) to find the dissimilarities between the LS of each individual in the same cluster with respect to LS of participants from other clusters.

DA was typically used to minimize LS variation within each cluster and maximize the LS variation between the clusters [35], [40]. CA gave us data about similarities of LS preferences of each individual. However, it was not effective for the data with dissimilarities in LS preferences in same clusters. An individual was classified into a cluster that has the maximum GM. Higher GM indicated similar LS when compared to a specific cluster. Hence, DA provided GM values for each individual with respect to each cluster. DA also helped us to examine and justify the CA results. Studies show that it is rare having a CA classification a little different than DA classification [39]. However, if different, there are evidences from literature that DA classification can improve the misclassification of CA results [40]. Hence, all known clusters were assigned with the individuals based on DA classification. In this way, we have used the GM values to sort the inspection teams ranging from most dissimilar to similar LS preferences. The process of creating and sorting teams based on their LS preferences was automated by using a tool (see 3.4).

3. STUDY DESIGN

This study was conducted to investigate whether a team of inspectors having dissimilar learning style (LS) would be able to detect more number of faults in comparison to inspectors having similar learning styles. The LS scores were collected via online survey. Fault-checklist method was used by participants to find and report faults in the software requirement document. All participants took one pre-study survey (work-experience related) and one post-study survey (research participation satisfaction). After collecting the LS's of 19 inspectors, all possible virtual teams were created by random selection process from the pool of 19 inspectors. Team sizes were maintained from 2 inspectors to 10 inspectors sorted from most dissimilar LS's of inspectors to most similar LS's of inspectors for each team size. For each virtual inspection team, we analyzed the data to find effectiveness. The details of the studies are provided below:

3.1. Research Goal

The goal is to research the use of LS concept regarding real world issues to improve software quality.

1. Most dissimilar LS's of individuals improving effectiveness of team performance:

- Analyze: if LS's of individuals have positive correlation with the team performance
- Purpose: to address the impact of most dissimilar LS preferences of individuals on team performance
- For project managers, business analysts, HR department (software managers): chose the right individuals for inspection with most dissimilar LS's in a team
- Context: finding and addressing most number of faults in an SRS document with seeded faults

2. Also, with data points (faults found and LS's of individuals), we want to focus on evaluating which type of LS preference(s) can find the most relevant and most non-over-lapping faults.

- Analyze: LS's of individuals
- Purpose: Evaluate SRS inspection data and LS preferences to find which LS preference is contributing most for the effective inspection team performance
- For software managers: to compose inspection teams with individuals having certain LS preferences
- Context: finding most number of non-overlapping faults in the SRS with seeded faults

3. Analyze the team size, i.e. the numbers of inspectors in a team ranging from 2-10 per team effect the team performance

- Analyze: team size for most effective team performance (most non-overlapping faults found)
- Purpose: Evaluate which team size most effectively able to find faults in the SRS
- For software managers: to have right number of inspectors in a team
- Context: finding most non-overlapping faults in a requirement document

3.2. Study Variables

3.2.1. Independent Variable

Learning style – to determine how an individual prefer to perceive information and interprets it.

3.2.2. Dependent Variable

Effectiveness (Fault data) – total number of unique faults found by an inspector in a requirement document.

3.3. Experiment Procedure

Artifact: Artifact used for the study included a common and generic requirements document (developed externally by Microsoft) describing requirements for Loan Arranger Financial System (LAFS) for an artificial financial firm and containing seeded faults was provided. The requirements document for LAFS was 10 pages long with 49 detailed requirements with 30 seeded faults [41], [42]. This requirements document has been used in multiple inspection related studies. LAFS allows the functionality for a financial institution to sell group of loans (bundles based on user-specified characteristics) to other financial institutions. Loan analysts at the financial institutions have capability to access the repository of mortgages (loans) purchased by the financial firm, searching for loans based on outstanding value, risk, and remaining term of loan. The loans are managed by loan arranger for investment and resell by creating portfolios. All loan related information, lenders details, borrowers information is managed using the application.

Pre-Study Questionnaire: A survey containing four experience related questions was used to determine the experience level of each individual. The survey elicited information about their experience in working with end-users, use-cases, gathering and inspecting requirements and changing requirements for maintenance.

LS questionnaire survey: The FSLSM (Felder's-Silverman Learning Style Model) questionnaire with a total of 44 multiple-choice questions (available at http://www.engr.ncsu.edu/learningstyles/ilsweb.html), was provided to all 19 participants in order to determine the LS of each individual. From the questionnaire results given above (see Fig. 1), the LS Score was measured for all 8 categories. To calculate the actual LS scores for each individual in both categories across all four dimensions, we analyzed the scorecards. There

were 11 questions per LS dimension. To analyze the scorecards and get actual scores, 1 point is awarded for each correct answer for each category. Table 4 illustrates an example of actual score table. For example, for ID 1 in Table 4, the individual LS score for Active/Reflective is 3 (7 minus 4) towards reflective.

Table 4: Example of actual scores of participants

ID	Act	Ref	Sen	Int	Vis	Ver	Seq	Glo
1	4	7	8	3	5	6	3	8
2	7	4	5	6	2	9	8	3
3	6	5	7	4	3	8	10	1

Inspectors: The study was conducted on 19 individuals working in the IT industry and at universities with different levels of experience. On an average, the participants had 4 to 5 years of work experience. Participants' daily duties and responsibilities included conducting systems analysis, developing project plans, software development and re-factoring code, database design and development, and data collection, analysis and reporting. They worked with clients for requirement gathering, code specifications, testing, verification, implementation, and delivery of upgrades, hotfixes and releases. Also, system security and integrity, meeting hardware requirements, providing client support and system maintenance was part of their duties and responsibility.

Inspection process: A brief introduction to the inspection process and fault checklist method was given to the participants. Relevant training material along with a sample fault checklist and a template spreadsheet for recording faults was also provided. All the participants inspected a SRS document for LAFS using inspection process guidelines. During inspection, they recorded faults in the fault checklist template in a 60 minutes time frame excluding breaks.

Data points: LS Score from ILS survey and Fault Data from requirements document inspection of each participant were collected. The faults reported by each participant were verified by one of the researchers for true-positives. The researcher compared the faults reported with the seeded faults to remove any false-positives before analyzing the data. All faults recorded by each individual were then validated to avoid any false positives before analyzing the data. All faults were classified under following fault types: Omission (O), Ambiguous Information (A), Inconsistent Information (II), Incorrect Fact (IF), Extraneous (E) and Miscellaneous (M).

3.4. Evaluation Criteria

We gathered the LS scores of participants to evaluate: (a) if dissimilar LS preferences in a team is effective (Goal 1), (b) effect of each type of LS favoring (or not favoring) the inspection results (Goal 2), and (c) the effect of number of people involved in the inspection team on team performance (Goal 3). Generally, an inspection team is composed of 4 to 6 inspectors in a software project. Hence, a decision was made to create virtual teams of 2 to 10 inspectors. We developed a tool to generate all possible virtual inspection teams based on data generated from LS of participants and the faults found by them. The tool was designed to detect fault coverage (unique faults found) and fault overlap. It also sorted the teams in descending order from most dissimilar to least. The steps below explain further how data is evaluated for the research goals.

a) **Creating Virtual teams** – Virtual teams were generated by the tool we developed. These teams consisted of members who never met each other. Fault data gathered from participants with different LS was randomly combined to create teams. Data was divided into sets, e.g. Data set 1 would have 2 inspectors in the teams with replacement from the pool of 19

inspectors; Data set 2 would have a team of 3 inspectors with replacement and so on. Inspectors were shared among teams. For example, to create a team of 4 (from a pool of 19 inspectors), 3876 virtual inspection teams (i.e. ${}_{19}C_4$) were created.

b) **Grouping similar inspectors in clusters –** Correlated categories in each LS dimension were converted into uncorrelated variables (PCs) based on the PCA results (mentioned in section 2.3.1). Grouping was done using CA (as explained in section 2.3.2) and inspectors with similar LS's were grouped together in the same cluster (number of cluster was same as the team size being analyzed). Table 5 shows the cluster output for a team of size 2 inspectors formed by the tool, first column represents inspector ID number and the second column shows the cluster number for that respective inspector. In this example, it forms 2 clusters with teams of size 2, inspectors 2, 3, 6, 8, 11, 12, 13, 16, 17 and 18 grouped into cluster# 1 and the rest in cluster# 2.

Table 5: Grouping (clustering) of inspector team size 2

Inspector ID	Cluster #		Inspector ID	Cluster #
1	2		11	1
2	1		12	1
3	1		13	1
4	2		14	2
5	2		15	2
6	1		16	1
7	2		17	1
8	1		18	1
9	2		19	2
10	2			

Now, using DA (explained in 2.3.3), the tool calculated GM (group membership) for each inspector in a cluster (shown in table 5). In table 6, Column 1 shows inspectors ID and Column 2 shows GM for 2 clusters C1 and C2. Inspector id #11 has GM value of 1 for C1, this implies that inspector #11 has the highest possibility of belonging to the C1 cluster as compared to the other inspectors.

The tool then sorted the inspection teams based on the level of dissimilarity of LS preferences from most dissimilar to least. In fig. 4, the output of the sorted teams (for team size 2) is shown, where the inspectors # 7, #11 shows the team with most dissimilar LS preference, and where each inspector belongs to a different cluster. Moving down the list, it can be seen that Total GM is decreasing, thereby showing the decrement in dissimilarity of LS. Team #30 has the least dissimilarity in LS, this team contains individual inspectors belonging to 2 different clusters. Teams #31 - #55 represent the dissimilar to similar LS of teams having individual inspectors from same clusters (either C1 or C2), which implies that #55 is the team with most similar LS's (of all 55 teams) – having the least 'Total GM' values from all the teams having individuals from same cluster. This procedure was repeated for all the teams across the size from 2 to 10 inspectors.

Table 6: Group membership for 3 clusters

C1	**InspectorID**	**2**	**3**	**6**	**8**	**11**	**12**	**13**	**16**	**17**	**18**
	GM	0.88	0.97	1	0.91	1	0.93	0.97	0.88	0.91	0.93
C2	**InspectorID**	1	4	5	7	9	10	14	15	19	-
	GM	0.65	0.7	0.61	1	0.78	0.95	0.98	1	1	-

Team#	Inspector ID	Cluster No.	GM	Inspector ID	Cluster No.	GM	No. of Clusters Involved	Total GM
1	7	2	1	11	1	1	2	2
2	6	2	1	11	1	1	2	2
3	3	1	0.97	7	2	1	2	1.97
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
30	1	2	0.65	8	1	0.94	2	1.59
31	6	2	1	7	2	1	1	2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
53	2	2	0.88	4	2	0.7	1	1.58
54	1	2	0.65	2	2	0.88	1	1.53
55	1	2	0.65	4	2	0.70	1	1.35

Figure 4: Teams formed from 2 clusters

c) **Evaluating inspection performance of teams** – Performance of each team was evaluated by combining the individual fault data and calculating the unique number of faults and the redundant number of faults found by each virtual team of all sizes.

4. ANALYSIS AND RESULTS

This section provides the analysis of inspection results of different team structures. These teams were of different sizes based on the LS and fault data to test the effect of variation in LS's on the inspection team performance. The teams were also classified based on distribution of fault types (mentioned in section 3.4) across different LS dimensions and categories.

In this section, we utilized the LS scores of 19 subjects to investigate the effect of different LS's of different individual inspectors. The result was calculated based on the effectiveness of the inspection teams. As discussed earlier (in Section 3.4), teams of each size were sorted with most dissimilar LS's (highest number of clusters involved) to the teams with most similar LS's (have least number of clusters involved).

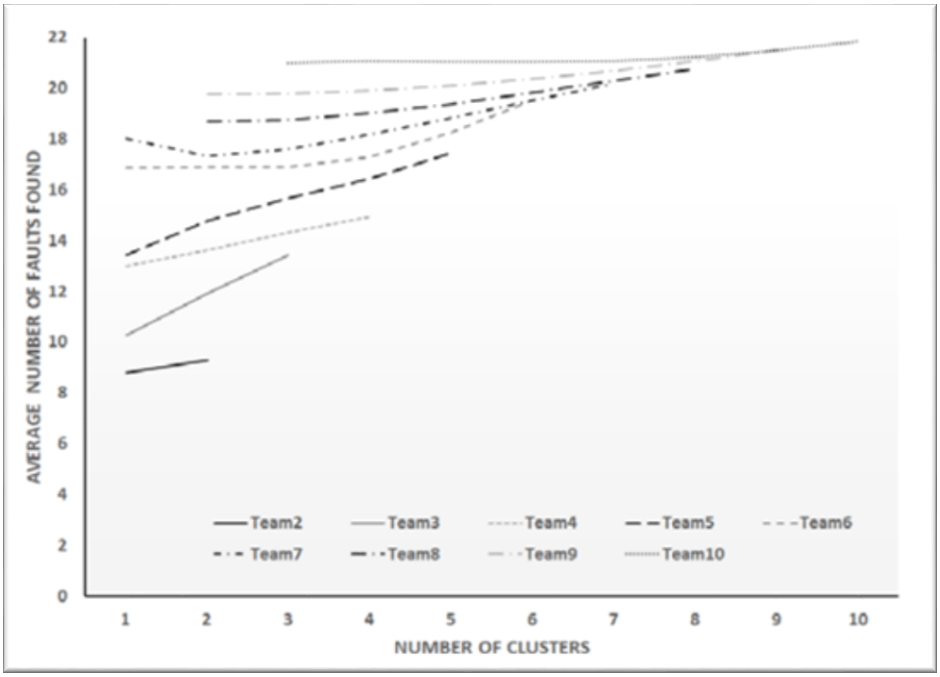

Figure 5: Comparison of the LS on the effectiveness of inspection team of each size (2 to 10 members)

For each team size, all possible virtual inspection teams were formed. These teams were then sorted from most dissimilar LS to most similar LS of participating inspectors.

We combined the individual result data (inspection data from section 3.4) to form a team score for each virtual inspection team. To investigate the fault detection effectiveness for each team, only the unique faults found in the "Loan Arranger" requirement documents were considered. All possible virtual inspection teams for all sizes were analyzed under the same condition.

Fig. 5 shows the overview of the results i.e. the average number of faults found by teams having 2 to 10 inspectors. The effectiveness of the results was organized by the increasing number of clusters involved in the team formation (where clusters involved are directly proportional to dissimilarity – higher the cluster number, more dissimilar LS of team members).

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 2 or 3 clusters for team size 3 and so on. As the team size increased, the number of participants that belong to the same cluster decreased. This reduced the probability that a team would be formed from lesser number of clusters. The teams could not be formed for larger team sizes e.g. a team of size 10 (follow Fig. 4) because the tool created 10 different clusters via k-means algorithm. 10 participants were distributed in these 10 clusters and there was no single cluster that contained all the participants. Therefore, no virtual teams could be created in a single cluster.

The effectiveness results (Fig. 5) shows that, for a team size of 2 to 10 inspectors, teams formed with the inspectors selected from higher number of clusters (i.e., inspectors with dissimilar LS) were able to detect more faults as compared to teams with inspectors belonging to fewer number of clusters (i.e., inspectors with similar LS). However, effectiveness for team size

2 was not that evident. In this case, the inspectors were grouped only between two clusters that do not provide a clear separation between their LS's. However, teams with sizes greater than 2 showed a proportional increase in fault detection with an increase in the number of clusters used to compose inspection teams. For certain team sizes (example size 6 and 7), the count of unique faults dropped a little for teams formed with higher number of clusters but increased again with increased number of clusters. This implies that teams with diverse LS of inspectors are more effective. In cases of teams of size 9 and 10, there's no significant increase in the number of unique faults detected even with the increase in number of clusters used to compose team. This is because of the 8 categories of LS that clusters were based on. When there are more than 8 team members (i.e. the case with team of size 9, 10), LS of individuals overlap causing the similar faults to be found.

We performed linear regression test to find the correlation between the dissimilarity of LS of inspectors and the number of unique faults found by inspection teams of different sizes. The results show that there is a significant positive correlation among dissimilarity in the LS of inspectors with the team effectiveness for team sizes 3 to 10 (as shown in Fig. 6). There is no significant correlation for team size of 2 inspectors, which was due to the fact that the team size ranging from 3 to 10 inspectors covered more diversity in LS. Diversity in the overall team LS led to different types of faults to be found (also discussed later) resulting in an increase in the team effectiveness.

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	$P=0.001;\ r^2=0.004$

Figure 6: Linear regression results - statistical analysis

We analyzed the impact LS dimensions had on inspection effectiveness and nature of fault found. We analyzed the impact LS dimensions had on inspection effectiveness and nature of fault found. These dimensions are made up of a combination of categories. This was done to gain further insight on whether certain LS are responsible for higher inspection effectiveness and detection of fault types. To perform this analysis, we captured the classification of faults according to their fault type (Section IV) found by the participants belonging to each cluster. It was found from raw data that none of the inspectors had a preference towards Verbal-VER. The remaining LS categories (Active–ACT, Reflective–REF, Sensing–SEN, Intuitive–INT, Sequential–SEQ, and Global–GLO) were analyzed.

To analyze the effect of each LS dimension on inspection effectiveness and fault types, we created groups of each LS using six LS categories across three LS dimensions. Therefore,

eight clusters with their respective number of members were: REF-SEN-GLO (five), ACT-SEN-SEQ (five), ACT-INT-SEQ (one), REF-SEN-SEQ (one), REF-INT-SEQ (two), ACT-SEN-GLO (one), ACT-INT-GLO (two), and REF-INT-GLO (one).

Fig. 7 shows the average inspection effectiveness (shown by solid line) and average fault type (shown by bars) correlation for each LS cluster during inspection. Results are ordered from most effective to least effective cluster. Left y-axis shows the average number of fault types detected and secondary y-axis on the right shows the average inspection effectiveness.

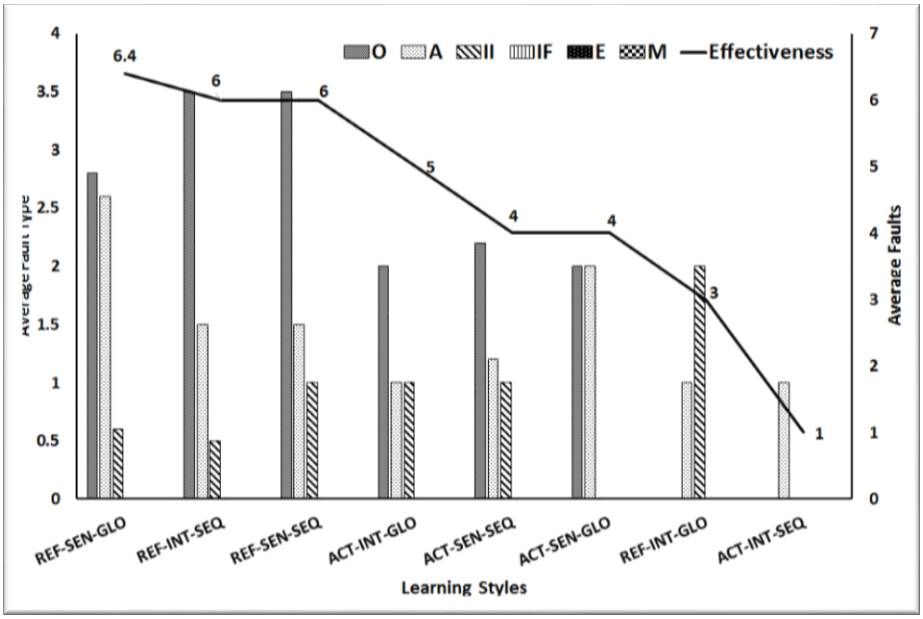

Figure 7: Effectiveness and Fault Types by each LS

Based on results in Fig 7, following observations were made:

a) Inspectors with REF-SEN-GLO LS had the maximum inspection effectiveness (6.4). REF-INT-SEQ LS cluster found the next maximum effectiveness (6). Upon analyzing the pre-study survey data, it was found that inspectors with REF-SEN-GLO LS preference had high experience of working with requirements analysis as compared to inspectors with a preference for REF-INT-SEQ LS.

b) Inspectors belonging to the REF-SEN-GLO (i.e. most effective cluster) found the maximum number of Ambiguous Information fault (A). It can be inferred from this that inspection performance relies on a combination of categories along each LS dimension and a single LS category cannot help detect all types of faults. Hence, there should be an inspection team with inspectors of diverse LS's.

c) Inspectors from the REF-INT-SEQ and REF-SEN-SEQ cluster found the maximum number of Omission (O) faults. Also, REF-INT-GLO cluster found maximum number of Inconsistent Information (II) faults.

These results reinforce that a combination of different LS's enabled inspectors to find faults of different types and also that a difference in the LS of inspectors enable a higher coverage of faults present in an artifact. However, it also revealed that inspectors belonging to the five LS clusters (i.e., REF-SEN-GLO, REF-INT-SEQ, REF-SEN-SEQ, ACT-INT-GLO, ACT-SEN-SEQ) out of 8 clusters were able to uncover all the three types (O, A, and II) of faults present in requirements document.

5. THREATS TO VALIDITY

In this experiment, we were able to address some validity threats.

Conclusion Validity: The participants in this survey came from different backgrounds (within software engineering) and were at different levels of experience. Elements in different work environment include number of years of work experience (with an average of 3 years of work experience), job profiles and the companies they are working for.

There was no time limit to take LS survey and 60 minutes was provided to perform requirement inspection exercise. They were allowed to work in an environment suited to them. In this way, we addressed the fatigue effect. The same training material on inspection and fault finding technique was provided to each participant. However the participants trained themselves which could have caused training bias. The participants were professionals in industry representing the effect of LS on individuals in the real work environment. However, we did not control the group heterogeneity. Participants were from different companies with different experience and job types (managers, developers from different fields and software projects) and the experience they have. The effect of LS of author(s) of SRS on the inspection process could not be controlled as the requirement document was obtained from an external source and access to the authors LS was not available. Also, for larger teams (e.g. team size 10), there was not enough data to form teams from a few numbers of clusters (e.g., 1 or 2). These issues with respect to the generalization of the results will be addressed in future studies.

6. SUMMARY AND DISCUSSION OF RESULTS

The major focus of the study was to investigate the effect LS has on the performance of inspection teams during requirement inspection. Discussion of the results related to the hypotheses are as follows:

Hypothesis: The teams where members have dissimilar LS are more effective than the teams where members have similar LS for the requirement inspection phase.

The hypothesis is focused on understanding whether the inspection team found a higher number of unique faults during the inspection when the participating inspectors had variations in their LS preferences and strengths. The results from Section 4 (Fig. 5) show that the dissimilarity in the LS of participating inspectors was directly proportional to the effectiveness (fault count) of the inspection teams. This means that for an inspection team, with increased dissimilarity in the LS of inspectors, more number of unique faults were found during the requirement inspection. After statistical analysis, test results show a significant correlation between the LS dissimilarity and effectiveness, which supports the hypothesis. Therefore, based on the results of the study, using LS's of inspectors to guide the development of inspection teams is beneficial. Results also revealed that certain LS favor inspection positively (i.e. high effectiveness and detection of particular fault type). Based on the results provided in this paper, the concept of LS is applicable in software inspection domain and can help to manage the quality of software by creating high performance inspection team(s).

.

7. CONCLUSION AND FUTURE WORK

With the results obtained during this study, we can conclude that the concept of LS is applicable in software inspection domain and should be used to determine the team of inspectors to maximize the inspection effectiveness. Although the data size used is small, the results show that an inspection team consisting of inspectors with different LS analyzes requirements with a different perspective. This results in less overlap of analysis in a team and leads to varied output identifying higher number of risks. Our software tool can be used by researchers and project managers to manage the inspection process depending on their need of overall fault coverage. It also helps inspectors focus on a particular type of requirement fault. Our future focus would be on replicating our study for larger data sets at an industrial level. The results were narrowed down for the requirement inspection process, however this concept can be extended for quality improvement processed of other activities involved in the SDLC process (like design review, code review, best practices review etc.). The data shall be analyzed to evaluate the correlation (positive or negative) among inspectors with certain LS preferences (e.g., *Active vs. Reflective)* may have on their performance during the requirements inspections. These results are interesting and motivate us for further investigation.

8. REFERENCES

[1] Andrew, J.N, and Bytheway (1999). Successful Software Projects and How to Achieve Them. *IEEE Software*, 16(3), pp. 15-17.

[2] Belassi, W., and Tukel, O. I. (1996). A New Framework for Determining Critical Success/Failure Factors In Projects. *International Journal of Project Management*, 14 (3), pp. 141-151.

[3] Geethalakshmi, S.N. & Shanmugam, A. & Subbarayan, V. (2007). Critical Success and Failure Factors of Software Projects. *The International Journal of Technology. Knowledge & Society*, 3(1) pp. 103-106

[4] Irja Hyvari, (2006). "Success of Projects in Difference Organizational Conditions". *Project Management Journal*, 37(4), pp. 31-40.

[5] Rupinder Kaur, Dr. Jyontsna Sengupta. (2011, Feb). Software Process Models and Analysis on failure of Software Development Projects. *International Journal of Scientific and Engineering Research* 2 (2).

[6] Berry, D.M., and Kamsties, E. (Springer 2004). Ambiguity in requirements specification,.*Perspectives on software requirements* (Springer, 2004), pp. 7-44

[7] Boehm, B., and Basili, V. (2001). Software Management Software Defect Reduction Top 10 List, *COMPUTER-LOS ALAMITOS*-, 34 (1), pp. 135-137

[8] Chillarege, R., Bhandari, I.S., Chaar, J.K., Halliday, M.J., Moebus, D.S., Ray, B.K., and Wong, M.-Y. (1992). Orthogonal defect classification-a concept for in-process measurements, *Software Engineering, IEEE Transactions,* 18 (11), pp. 943-956

[9] Aceituna, D., Do, H., Walia, G.S., and Lee, S. W. (2011). Evaluating the use of model-based requirements verification method: A feasibility study. *Empirical Requirements Engineering (EmpiRE)*, *First International Workshop,* pp. 13-20

[10] Parnas, D.L., and Lawford, M. (2003). The role of inspection in software quality assurance. *Software Engineering, IEEE Transactions*, 29(8), pp. 674-676

[11] Shull, F., Rus, I., and Basili, V. (2000). How perspective-based reading can improve requirements inspections. *Computer*, 33(7), pp. 73-79

[12] Porter, A.A., Votta Jr, L.G., and Basili, V.R. (1995). Comparing detection methods for software requirements inspections: A replicated experiment. *Software Engineering, IEEE Transactions*, 21(6), pp. 563-575

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

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

[15] Goswami A., Walia G. (2013, November). An Empirical Study of the Effect of Learning Styles on the Faults found during the Software Requirements Inspection. *Software Reliability Engineering (ISSRE)*, pp 330-339

[16] Ackerman A. Frank, Buchwald Lynne S., Lawski Frank H. (1989, May/June). Software Inspections: An Effective Verification Process, *IEEE Software*, 6(3), pp. 31-36

[17] Leszak, M., Perry, D.E., and Stoll, D. (2000). A case study in root cause defect analysis, in Editor (Ed.)^(Eds.). *Book A case study in root cause defect analysis*, ACM, pp. 428-437

[18] Miller, J., Zhichao Yin. (2004, Nov.). A congnitive-based mechanism for constructing software inspection teams. *Software Engineering, IEEE Transactions*, 30(11), pp. 811-825

[19] Porter, A., Siy, H., Mockus, A., and Votta, L. (1998). Understanding the sources of variation in software inspections. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 7(1), pp. 41-79

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

[21] Felder, R.M., and Silverman, L.K. (1988). Learning and teaching styles in engineering education. *Engineering Education*, 78(7), pp. 674-681

[22] Felder, R.M., and Henriques, E.R. (1995). Learning and teaching styles in foreign and second language education. *Foreign Language Annals*, 28(1), pp. 21-31

[23] Pashler, H., McDaniel M., Rohrer, D. and Bjork, R. (2008, Dec.). Learning Styles Concepts and Evidence. *Psychological Science in the Public Interest*, Dec 2008, 9(3) pp. 105-119

[24] Barry L. B., Theresa P. M. (2004). Evaluating the scope of learning styles instruments used in studies published in the journal of agricultural education, *Journal of Southern Agricultural Education Research*, 54(1)

[25] Kolb, D.A. (1984). Experiential learning: Experience as the source of learning and development. (Prentice-Hall Englewood Cliffs, NJ)

[26] Kolb, A. Y., & Kolb, D. A. (2005). *The Kolb learning style inventory – version 3.1: 2005 Technical specifications*. Boston: Hay Resources Direct

[27] Mumford, Alan. (1995). Learning Styles and Mentoring. *Industrial and Commercial Training*, 27(8)

[28] Zwanberg, N. V., Wilkinson, L. J. and Anderson, A. (2000). Felder and Silverman's Index of Learning Stylesand Honey and Mumford's Learning StylesQuestionnaire: How do they compare and do they predict academic performance? *Educational Psychology*, 20(3), pp. 365-380

[29] Farkas,R. D. (2003, Sep./Oct.). Effects of Traditional Versus Learning-Styles Instructional Methods on Middle School Students. *Journal of Educational Research*, 97(1), pp. 42-51

[30] Solomon, B.A., and Felder, R.M. (1999). Index of learning styles, Raleigh, NC: North Carolina State University. Website: http://www.engr.ncsu.edu/learningstyles/ilsweb.html

[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), pp. 103-112

[32] W. Paul. Vogt. (1999). Dictionary of Statistics & methodology: A Nontechnical guide for the Social Sciences

[33] Craig A. Mertler, Rachel A. Vannatta, (2002). *Advanced and Multivariate Statistical Methods*, Second edition, Chapter 2: Guide to Multivariate Techniques, Los Angeles, CA: Pyrczak Publishing

[34] Svante Wold, Kim Esbensen, Paul Geladi. (1987, Aug). Principal Component Analysis, Chemometrics and Intelligent Laboratory Systems, 2(1-3), pp 37-52

[35] Anderson, T.W., Anderson, T.W., Anderson, T.W., and Anderson, T.W. (1958). An introduction to multivariate statistical analysis. (Wiley New York)

[36] 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), pp. 408-417

[37] Steinbach, M., Ertöz, L., and Kumar, V. (2004). The challenges of clustering high dimensional data. *New Directions in Statistical Physics. Springer*, pp. 273-309

[38] Tan, Steinbach, Kumar, (2005). *Introduction to Data Mining (First Edition)*, chapter 8: Cluster Analysis: Basic Concepts and algorithms, Boston, MA: Addision-Wesley Longman Publishing Co.

[39] Galbraith, J., and Lu, J. (2001) *Inequality and Industrial Change: A Global View*. Chapter 16, New York, NY: Cambridge University Press

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

[41] F. Shull, J. Carver, and G. Travassos. (2001, Sept.). An Empirical Methodology for Introducing Software Processes. *Proc. Joint Eighth European Software Eng. Conf. and Ninth ACM SIGSOFT Symposium. Foundations of Software Eng.*, pp. 288-296

[42] Jeffrey C. Carver, Nachiappan Nagappan, Alan Page. (2008, Nov.). The Impact of Educational Background on the Effectiveness of Requirements Inspections: An Empirical Study. *IEE Transactions on Software Engineering*, 34(6)