UNDERSTANDING HUMAN ERRORS TO IMPROVE REQUIREMENTS QUALITY

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

By

Kavitha Manjunath

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

Major Program:
Software Engineering

February 2018

Fargo, North Dakota

North Dakota State University
Graduate School

Title

UNDERSTANDING HUMAN ERRORS TO IMPROVE REQUIREMENTS QUALITY

By

Kavitha Manjunath

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 E. Nygard

Dr. Limin Zhang

Approved:

February 27, 2018 — Date

Dr. Kendall Nygard — Department Chair

ABSTRACT

Requirements engineering is the first and perhaps the most important phase of software life cycle. Software faults committed during the requirements development, if left undetected can affect downstream activities. While previous research has developed fault-detection techniques, they are inadequate because they lack an understanding of root cause of faults to be able to avoid its future occurrences. Our research is aimed at helping software engineers understand human errors (i.e., the root cause) that cause faults that are inserted into Software Requirements Specification (SRS). This can help software engineers become more aware and reduce the likelihood of committing human errors during requirements development. Using a retrospective error abstraction approach, this paper reports the results from an industrial study wherein software engineers were trained on human errors followed by their application of human error abstraction (based on fifteen requirement faults supplied to them) and recommendations on prevention techniques for those abstracted errors.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Gursimran Singh Walia, Associate Professor, Department of Computer Science and Software Engineering, North Dakota State University. His support was humungous, and he steered me in the right direction to achieve my goal. I would like to thank my committee members, Dr. Kendall Nygard, Department Chair, Assistant Professor, Department of Computer Science and Software Engineering, North Dakota State University and Dr. Limin Zhang, Associate Professor of Management, North Dakota State University accepting to be in my committee and supporting me.

I would like to thank my husband, Ram for his love, support, guidance and constantly cheering me to achieve greater heights. I would like to thank my parents, Manjunath and Vijayalakshmi for blessing and supporting me. I would like to thank my in-laws Moorthy and Vijayalakshmi for their encouragement. I would like to thank my entire family who never stopped believing in me and who encouraged and trusted the education system.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

RE ....Requirements Engineering.

SDLC ....Software Development Life Cycle.

SKR..Skill, Knowledge, Rule.

SRS ....Software Requirements Specification.

PGCS..Parking Garage Control System.

HE ....Human Error.

HEA ....Human Error Abstraction.

HEAA ....Human Error Abstraction Assist.

HET..Human Error Taxonomy.

EA ....Error Abstraction.

EAI...Error Abstraction and Inspection.

RIM ....Restaurant Interactive Menu.

RQ ....Research Question.

1. INTRODUCTION

Requirements engineering (RE) is the most important phase in software engineering and is often fuzzy due to interaction between different stakeholders that are involved during the eliciting, analysis, specification and management of user needs and requirements. Successful software projects require understanding of user needs which are translated into software requirements that in turn are engineered to deliver trustable, reliable and workable software product. Many critical and human oriented missions such as crash between two Boeing 747s and Chernobyl nuclear power plant incidents have failed due to mistakes in one of the stages of requirements engineering and have been life-threatening [12, 24]. There is also harmony on the nature of requirements engineering activities being diverse due to involvement of many stakeholders [25-28]. Also requirements are written in natural language, augmented by information in other representations, such as formulae and diagrams, which can cause ambiguity and misunderstanding.

The literature provides varied and often competing definitions for errors ranging from program error to service errors to human errors. Our research is more closely associated with human error (defect in the human thought process) as opposed to program error (execution failure). A human error is a failure in human thought process while attempting to think, analyze or understand the situation or utilize methodologies and tools. A fault is an embedded action of errors and can result from one or more errors [1].

The idea of using error information to detect faults in Software Development Life Cycle (SDLC) is not novel. Several techniques have focused on identifying the root cause of a sample of problems [2] while others have tried to perform error abstraction on a sample of representative faults [3]. However, these techniques are time consuming, focus only on a subset of faults (that are analyzed for root cause) and do not focus on underlying cognitive theory regarding the type of errors committed by human. Hence studies are conducted to utilize fault detection techniques to find the root cause of problems in software requirements in the early stage of development.

Fairley [5] had conducted a study to show the cost increases five times to correct a fault at design stage, ten times at coding stage, 20 to 50 times at acceptance phase, 100 to 200 times during actual operation. Requirements defects are expensive to correct in later stages and formal inspections of requirements documents yield only a low rate of fault detection [6]. Studies show that more than 40% of development efforts are spent in fixing identified issues in the later stages of SDLC [4].

Having a knowledge of where and what type of errors occur, how individual thinks and analyzes the system can draw a reasonable value to create quality requirements.

Human error research in Cognitive Psychologists concentrates on failures in the human mental process while focusing on different tasks. Our group focuses on identifying human errors that occur during the requirements development and educating software engineers on human error information to improve requirements quality. To understand the types of human error, a systematic study [7, 8, 9] and research was conducted in software engineering utilizing cognitive psychology dominion. The results produced a detailed taxonomy was built to identify the errors in requirements.

The results from previous studies showed that the educating on human error taxonomy helps understanding the most common mistakes committed during the requirements engineering process and help engineers to be mindful. Also, it helps in creating a quality document to remove all common errors in the early phase rather than focusing to remove and rework on them during development and testing phases, which will be more cost effective.

The scattered information about errors and error techniques should be gathered in a structured framework to identify the shortcomings. The structured framework is offered in the form of a taxonomy. This work is motivated by limited existing analysis and evaluation of suitable techniques for validating the root cause for errors in requirements phase, the human error. This paper aims at understanding the human errors to improve requirements quality. A study with industrial practitioners was conducted to determine how the human error taxonomy can be trained and if it can improve in abstracting the taxonomical categories of a series of faults introduced in Parking Garage Control Systems SRS.

Section II provides some background on the error abstraction process and the requirement error taxonomy. Section III describes the study design. Section IV describes the data analysis results. Section V discusses the threats to validity. Section VI focuses on the relevance of the results and summary. Section VII concludes the paper and presents ideas for future work.

2. BACKGROUND AND RELATED WORK

Developing software requirements has heavy human involvement and thought process which may lead to failure during the execution, gathering or problem-solving phase. This is known as human error. Human cognition plays a vital role in requirements engineering process [10, 11]. Most of research related to human error is primarily limited to areas such as aviation, nuclear power plant, medicine and manufacturing. Rather, minimal work on human error specifically concerns the software engineering, and there is even less devoted to the requirements engineering.

There exist several human error classification systems in literature. Few are Reason's Swiss cheese model which is extensively used in aviation and medicinal industry [12], Human Factor Analysis and Classification System [13], Rasmussen's Taxonomy [14].

In 1983, Jens Rasmussen [15] expanded the cognitive aspects of error to include skill, knowledge and rule based classification, also known as SKR classification. In 1993, Rasmussen reinforced this theory that human functioned in one of the three levels of cognitive processing, based on the type of task and level of experience for that condition.

In 1988, Norman [16], studied both cognitive and motor aspects of error and differentiated between two types of error: slips and mistakes. This classification is also known as hybrid classification. According to Norman et.al, Slips were triggered by person's experiences and memory, and strategic knowledge. Slips occurred in everyday scenarios because of not applying full attention to the task. E.g., pouring orange juice to a bowl of cereal instead of milk while reading a newspaper. The act was unintentional but the required task was not achieved because the attention was focused on the newspaper. Mistakes occurred when a person's cognitive activities lead to decisions that are conflicting to what was intended. Mistakes result from the inadequacies of insight and decision-making and result in the failure to formulate the plan.

James Reason et.al, further expanded the type of errors. In 1990, Reason [12] described error as "*a generic term to encompass all those occasions in which a planned sequence of mental or physical activities fails to achieve its intended outcome, and when these failures cannot be attributed to the intervention of some chance agency*". The foundation for Human error comes from Reason's classification that is derived as part of our research. Section 2.1 describes Reason's error classification and its derivations in our research.

2.1. Classification of Human Errors by James Reason

Although slips and mistakes bid clear differences between two kinds of errors one is prone to make in daily lives, they are not precise enough for the classification of errors in operational environment. A much-enlarged definition of errors to distinguish and classify the error types was provided by Reason and his work considered to be the base in validating human errors particularly in operational environments such as aviation and medicinal fields. It provides a structure to understand how human errors are classified during information and thought process. According to Reason, the human errors occur during cognitive activities such as planning and execution.

Errors can generally occur during planning or execution phase. Plans can be sufficient or scarce and outcomes or behavior of the plan can be intentional or unintentional. If the plan is sufficient to achieve the required outcome and it is followed, a desired outcome. If the plan is not followed, desired outcome is not achieved. If the plan is inadequate, desired outcome is not achieved.

Errors can be classified into execution errors and planning errors. Human errors occur due to inattentiveness/carelessness are known as slips, whereas lapse happen due to forgetfulness. Inadequate knowledge to generate a plan to fulfill a certain action is called as mistake.

A ***slip*** of action is an unintentional and this occurs at the point of task execution. The examples can include transposing a number as 0.31 instead of 0.13 due to fat fingering. A ***lapse*** occurs after plan formulation and before task execution. This includes forgetting to execute a sequence in the plan or the plan. An example can be forgetting to switch the gear to take reverse of the car. In RE, slips and lapses occur in all four phases such as analyzing the requirements, eliciting them, writing the specifications, managing all requirements using traceability. A mistake occurs unintentionally, when a plan is set to achieve intended outcome, however due to inexperience or inadequate knowledge of the person, it is not formulated well. An example for a mistake can be bad weather is not considered during flight departure.

Slips and lapses tend to occur when (a.) people confuse between two similar tasks (b.) there are distractions while performing the task (c.) tasks are too complicated or time consuming to cause forgetfulness between two steps (d.) tasks are very familiar that the plan is memorized and requires little thought. Mistakes are result of time pressures or inadequate knowledge or doing too many things at the same time with poor judgement to formulate a plan.

Reason' theory is used in several safety-critical domains such as healthcare, aviation and nuclear plant for training on operator's safety. One such example is a document published by Duke University for medical training which can be found in http://patientsafetyed.duhs.duke.edu/module_e/definitions.html. The training documents based on Reason's theory are used in software industry and academics. Thus we can infer this theory is suitable for domains like software engineering for training individuals in academics and industry in RE.

2.2. Development of Human Error Taxonomy (HET)

Schneider [6] and coworkers identified fault detection using formal inspection techniques for requirements inspection which yielded low rate. Lanubile [3] and coworkers augmented the fault detection process by including an additional step to identify human errors that helped to find additional faults. Reviewers were asked to backward-look identify the errors by abstracting the human error from discovered faults. This additional information of human errors can help inspectors to find overlooked or missed faults in the requirements document.

Using this theory, our group developed human error taxonomy (HET) to support error abstraction (EA), where HET is utilized to provide a structured list of the most commonly occurring requirements phase human errors. Based on their experience, understanding and creativity, each inspector would inspect the human errors based on one's perspective. Hence, a taxonomy was created to help inspectors to use common terms and not natural language, which would lead to ambiguity and confusion. Thus, HET acts as a tool to help inspectors to abstract the errors using a concrete list.

Walia and Carver developed requirements error taxonomy (RET) a structured classification for error identification in development of SRS documents [22]. They also conducted experimental studies to validate the usefulness of RET [17]. Our group has conducted previous studies conducted by Walia and Carver [17], and Anu and co-workers [18] have shown error-based inspection (EAI) showed better results over fault-based inspection. Another experiment conducted by them [19] showed that EAI can help in identifying 150% of more faults compared to traditional methods. The research [20] also provided evidence that abstracting error correctly will have positive impact on additional fault detection. The research is conducted to validate if there is a prior understanding of the type of human errors that are committed during requirements engineering will be less leading to better quality SRS. Anu and coworkers [21] have worked on analyzing the misjudgments and conducted retrospective analysis and found faults for Restaurant Interactive Menu SRS.

A human error abstraction assist was developed to classify the errors under Reason's theory of slips, lapses and mistakes. HET was collected after rigorous collection of data, studying software engineering aspects and literature, human errors in RE. These were applied to Reason's classifications. Slips, Lapses and Mistakes were studied for every RE phase (Analysis, Elicitation, Specification and Management) and HET was structured for all the phases. For e.g., considering various definitions for same subject is a slip and leads to lack of inconsistency in requirements specification. Detailed taxonomy of commonly occurring errors as described by Anu [23] and coworkers is provided in Figure 1.

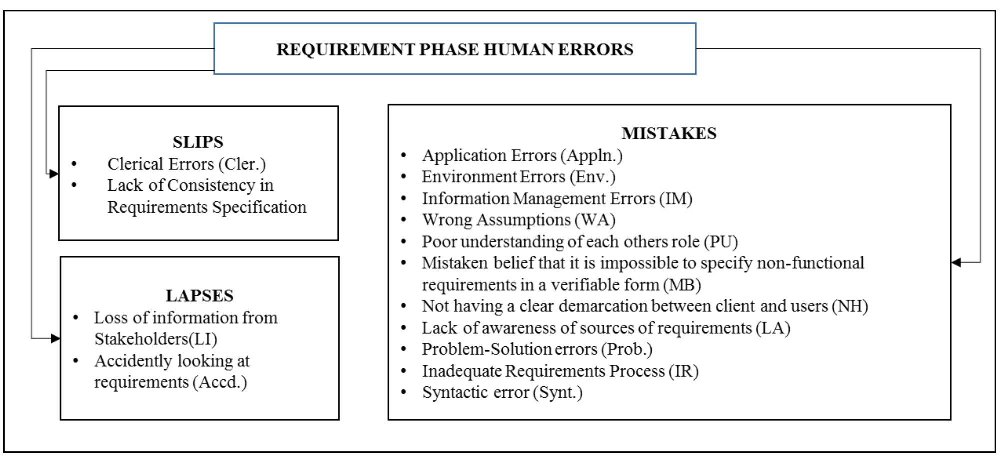


Figure 1. Human Error Taxonomy developed by Anu and co-workers

2.3. Error Abstraction using HEAA

Subjects were asked to abstract the error in three steps. Step 1 included abstraction for Level 1 – RE Activity. Step 2 included abstraction for Level 2 – Error Type and Step 3 was for

Level 3 – Error Class. The RE activities that were provided for abstraction are: (a.) Requirement elicitation is a phase where the requirements are discovered through consultation with the stakeholders, (b.) Requirements Analysis: a phase where the requirements are analyzed and the differences identified are resolved through negotiations with stakeholders, (c.) Requirements Specification: a phase where a precise and formal requirements documents are produced (d.) Requirements Management: a phase where requirements are managed because requirements and contexts evolve based on needs. The Figure 2 shows HEAA for abstracting RE activity.

To map error type, a decision tree was created and the definitions for slips, lapses and mistakes were explained with examples. In the decision tree, the situation or scenario is analyzed, plan is created and error type was classified. Figure 3 shows the HEAA for error types.

Further deep into the taxonomy, 15 types of error levels were defined and they were classified under each error type and RE activity. Figure 4 shows the error levels as defined by Anu and co-workers using software engineering studies and cognitive psychology as base that was provided as a guide for this experiment.

The Human Error Abstraction Assist (HEAA) aided the subjects to abstract the fault to RE activity, Error Type and Error Class Level. Since abstraction is retrospective activity, HEAA will help incline towards common terms instead of using creativity. 15 faults from PGCS SRS was provided to the subjects and were asked to abstract based on the training provided using video and taking help of Human Error Abstraction Assist (HEAA) document.

Step 1:

Choose one of the following options to decide where the fault originated (i.e., where the human error occurred and caused the fault to be inserted):

(a) Did the fault occur:

- While the system was being analyzed?
- While a large system was being divided into smaller parts?
- While system functionalities (functional requirements) and system behavior (performance and other non-functional requirements) were being determined?

(b) Did the fault occur during interviews or discussions with the stakeholders (end users, project sponsors, etc.)? This is where the user needs are gathered.

(c) Did the fault occur when the system information/requirements were being documented to create a formal software requirement document?

(d) Did the fault occur:

- During the *management* of the activities in a), b), or c) above?
- As requirements evolved or changed (i.e., traceability, version control, etc.)

RE activity associated to each option:

Option (a) – Requirement Analysis. **Option (b)** – Requirement Elicitation.

Option (c) – Requirement Specification. **Option (d)** – Requirement Management

Figure 2. Human Error Abstraction Assist for RE Activity

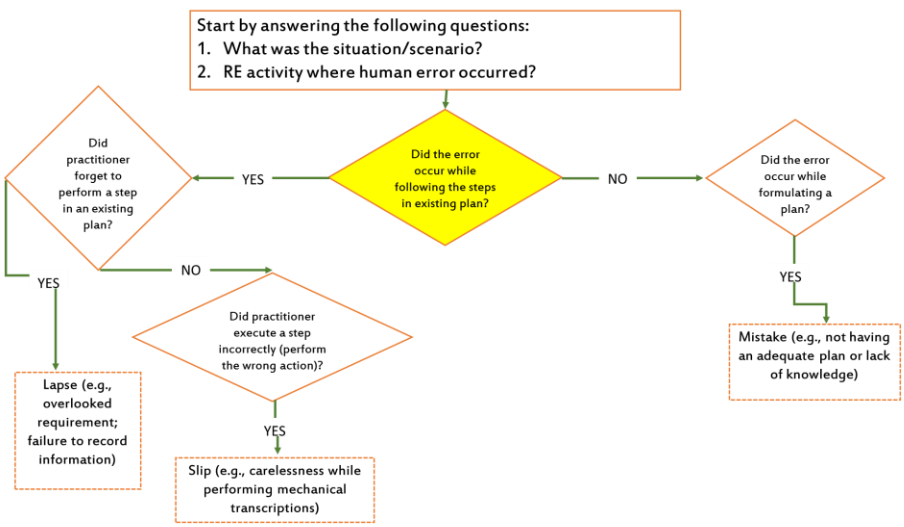


Figure 3. Decision tree to abstract Error Type

Step 3: Pick the appropriate Human Error

Requirement Analysis

Slips:

- **Clerical Error:** Carelessness while analyzing elicited requirements

Mistakes:

- **Application error:** analyst's misunderstanding or lack of knowledge of a part of (or the whole) system or problem
- **Environment error:** misunderstanding or misuse of the requirement analysis tools available for use in the project
- **Wrong assumptions** made by requirement analyst about user/stakeholder needs or opinions or any incorrect assumptions by RE analysts
- **Low understanding of each other's roles:** RE analyst does not understand the roles of all end users, stakeholders and other RE analysts.
- **Mistaken belief** of RE analysts that it is impossible to specify non-functional requirements in a verifiable form
- **Problem-Solution errors:** Lack of knowledge of the requirement analysis process and general requirement engineering know-how

Requirement Elicitation

Slips:

- **Clerical Error:** Carelessness while recording user needs

Lapses:

- **Loss of information from stakeholders:** Forgetting, discarding or failing to store information or documents provided by stakeholders.
- **Accidentally overlooking requirements:** Overlooking a requirement or some information that is crucial to the requirement

Mistakes:

- **Application error:** stakeholder's or requirement gathering person's misunderstanding of a part of (or the whole) system or problem
- **Environment error:** misunderstanding or misuse of the requirement gathering tools available for use in the project
- **Wrong assumptions** made by requirement gathering person about user/stakeholder needs or opinions or any incorrect assumptions made by requirement gathering person.
- **Low understanding of each other's roles:** Requirement gathering person does not understand the roles of all end users and stakeholders.
- **Mistaken belief** of requirement gathering person that it is impossible to specify non-functional requirements in a verifiable form
- **Not having a clear demarcation between client and users:** Requirement gathering person's misunderstanding of the difference between clients and users
- **Lack of awareness of sources of requirements**
- **Problem-Solution errors:** Lack of knowledge about the elicitation process (and/or general RE know-how)

Requirement Specification

Slips:

- **Clerical Error:** Carelessness while documenting specifications from elicited requirements.
- **Lack of consistency In Requirement Specifications:** Lack of logical coherence in the requirement specification documentation, which makes it difficult to be interpreted correctly

Mistakes:

- **Environment error:** misunderstanding or misuse of the requirement specification tools available for use in the project
- **Syntactic error:** Misunderstanding of grammatical rules of natural language (English) or grammatical rules of a formal requirement specification language.

Requirement Management

Mistakes:

- **Inadequate Requirements Process:** All steps required to ensure a robust requirement engineering process are not followed
- **Information Management error:** lack of knowledge about standard procedures and practices defined by the organization

Figure 4. Human Error Abstraction Assist for Error Class

3. STUDY DETAILS

The experiment was conducted to validate the accuracy with which the industrial practitioners perform human error abstraction using the developed and researched method of error abstraction (EA) and Human Error Taxonomy (HET). Also, the experiment data allowed us to validate HET in an industrial setting and gather feedback on prevention strategies for errors contained in HET.

3.1. Study Objectives

The major objective was to educate software developers on human error (using HET and HEAA that has been developed by Anu et al., [18, 19, 20]) and evaluate the usability of HET by having professionals' abstract errors (from faults provided to them) and map them to the errors contained in HET. This study investigates if subjects could abstract the correct RE activity where error originated, the type of error (Slips vs. Lapse vs. Mistake) and the error class (within each error type). Each subject was supplied with 15 faults (contained in Parking Garage Control Systems SRS) along with Human Error Abstraction Assist and training video to guide the error abstraction process. A total of 10 subjects performed the error abstraction (for same 15 faults). The error report form helped us to validate and compare the EA results for same 15 faults for all 10 subjects.

The following research questions were formulated:

RQ1: How well the subjects understood the Human error abstraction and mapped Error Type and Error class level accurately?

RQ2: Which errors in HET have been encountered by participating subjects at their workplace?

RQ3: What strategies are suggested to prevent the future occurrence of human errors?

3.2. Subjects and Artifacts

Ten (10) industrial practitioners working in software service and consulting firm participated in the study. An artifact that described the requirements for Parking Garage Control Systems (PGCS) that contained several faults was used in the study. PGCS system allows control and supervise entries and exits into and out of a parking garage. The system allows or rejects entries into the parking garage depending on number of available parking spaces. The fault seeding was done by Microsoft employees and does not bias the study results.

3.3. Experimental Procedure

A pre-recorded video and power point presentation was created to train the subjects on Human Error Taxonomy. The video was 28 minutes long and it provided in-depth training on human errors in everyday life and human errors committed in requirements engineering. The training provided step-by-step instruction on how to use the Human Error Abstraction Assist (HEAA) to abstract errors from requirement faults. Figure 5 shows the experiment procedure which included training video and fault list supplied to subjects and report of human errors (and their prevention strategies) as output.

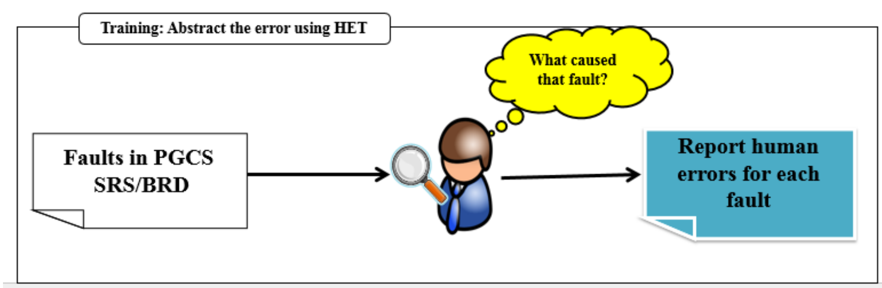
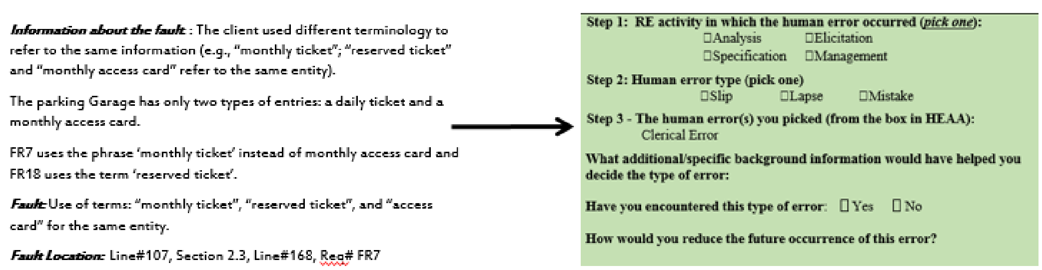


Figure 5. Study design to understand human errors

Sample error form provided as part of experiment is shown in Figure 6. Error form included series of steps and questions to collect data for 15 faults in Parking Garage SRS. Fault section contained information about the fault along with functional requirement numbers, fault explaining what exactly the issue is and line numbers from PGCS SRS document. The analysis part included 3 steps and additional questions. The three steps to abstract errors are mentioned in section 2.3. The subjects were asked to abstract RE activity in the first step by understanding the differences between each RE activity. Step 2 was error type abstraction using decision tree in Figure 3. Step 3 was to abstract error class using HEAA in Figure 4.

Error form includes additional questions such as: (a) what additional background information would have helped you to decide the type of error. This question was to understand if the training methodologies must be different. More than 90% of the subjects did not answer this question and skipped it. Hence this was not used for the analysis. Question (b) was have you encountered this type of error. A Yes/No type of question to understand if subjects are experiencing the type of error that they have abstracted in the error form. Question (c) how would you reduce the future occurrence of this error. This quantitative question helped us to gather prevention techniques for the errors occurred in requirements engineering process.


Information about the fault : The client used different terminology to refer to the same information (e.g., "monthly ticket"; "reserved ticket" and "monthly access card" refer to the same entity).

The parking Garage has only two types of entries: a daily ticket and a monthly access card.

FR7 uses the phrase 'monthly ticket' instead of monthly access card and FR18 uses the term 'reserved ticket'.

Fault: Use of terms: "monthly ticket", "reserved ticket", and "access card" for the same entity.

Fault Location: Line#107, Section 2.3, Line#168, Req# FR7

Step 1: RE activity in which the human error occurred (*pick one*):
☐Analysis ☐Elicitation
☐Specification ☐Management

Step 2: Human error type (pick one)
☐Slip ☐Lapse ☐Mistake

Step 3 - The human error(s) you picked (from the box in HEAA):
Clerical Error

What additional/specific background information would have helped you decide the type of error:

Have you encountered this type of error: ☐Yes ☐No

How would you reduce the future occurrence of this error?

Figure 6. Sample error form

4. ANALYSIS AND RESULTS

This section provides the analysis of the report form that we obtained from ten subjects who participated in this experiment. We also validated the error abstraction correctness at identifying human error type and human error class.

4.1. Accuracy of Error Abstraction

This section analyzes the accuracy of error abstraction for the fifteen (15) faults that were provided to the subjects (considered as inspectors). The accuracy was validated in all 2 levels: Level 2 – Error Type and Level 3 – Error Class. The goal was to evaluate if the subjects could validate all the faults correctly for different HEAA levels. Anu et al., [20] have worked at developing the HET and HEAA (with the help of Cognitive Psychologist Dr. Bradshaw) after conducting several academic studies. So, this paper adds new evidence to support the usability of HEAA in an industrial setting.

Table 1. Progressive EA Correctness at the levels 2 and 3 of HEAA

Fault #	Number of subjects who chose the correct Error Type (Level 2 of HEAA)	Number of subjects who chose the correct Error Class(Level 3 of HEAA)	Overall Correctness: Number of subjects who reported correct EA result for the fault(correct at 2 levels)
Fault 1	50% (5/10)	20%(1/5)	10%(1/10)
Fault 2	50%(5/10)	60%(3/5)	30%(4/10)
Fault 3	40%(4/10)	50%(2/4)	20%(2/10)
Fault 4	50%(5/10)	60%(3/5)	30%(3/10)
Fault 5	60%(6/10)	50%(3/6)	30%(3/10)
Fault 6	70%(7/10)	14%(1/7)	10%(1/10)
Fault 7	50% (5/10)	60%(3/5)	30%(3/10)
Fault 8	30%(3/10)	67%(2/3)	20%(2/10)
Fault 9	90%(9/10)	67%(6/9)	60%(6/10)
Fault 10	60%(6/10)	50%(3/6)	30%(3/10)
Fault 11	50%(5/10)	20%(1/5)	10%(1/10)
Fault 12	60%(6/10)	33%(2/6)	20%(2/10)
Fault 13	30%(3/10)	0%	0%(0/10)
Fault 14	60%(6/10)	17%(1/6)	10%(1/10)
Fault 15	60%(6/10)	33%(2/6)	20%(/10)

Table 1 provides collective results for EA correctness reported by ten (10) subjects working in software services. The experiment and HEAA required the inspectors to abstract the errors in two levels: Error Type at which human error has occurred (can be evaluated using the decision tree) and error class for deeper classification. Each row provides accuracy at the two levels the subjects were trained in to abstract the faults. The percentage of accuracy of selection for each subject was calculated based on whether the correct selection for error type was made. When we validated Error Type selection for 15 faults, all the subjects had chosen at least one Error Type. For the subjects who has chosen more than one Error Type, we counted the accuracy if the correct selection made, otherwise ignored for accuracy calculation. The same calculation and analysis was used for error class level. Thus, the validation of accuracy was hierarchical, where the correctness was required in both levels. Similar analysis is done for all fifteen (15) faults. Figure 6 shows the EA accuracies achieved for all three HEAA levels by ten subjects who participated in this experiment. Following are the analysis from Table 1 and Figure 7.

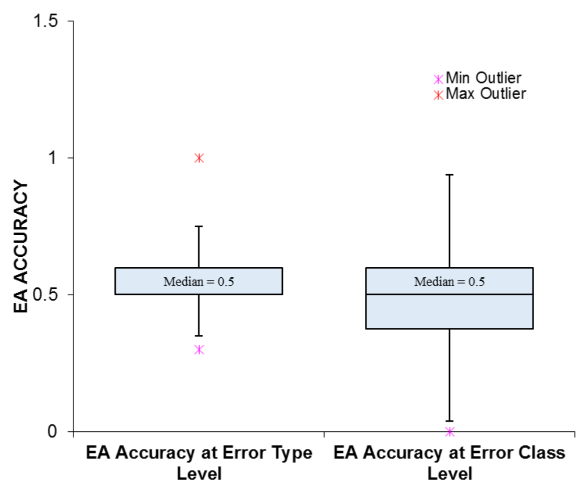


Figure 7. Accuracy for Error Abstraction (EA) for two HEAA Levels

Level 2 of HEAA – Analysis at Error Type: Based on the results from Table 1, Row 1 (*Number of subjects who chose the correct Error Type (Level 2 of HEAA)),* the results show subjects had less difficulty in abstracting the error for correct Error Type. The overall EA result percentage is 54% for ten subjects for fifteen faults. The result of abstracting the next level will be lesser or greater based on the accuracy of current level. Since the numbers are considered based on number of people selecting right RE activity, the analysis will differ in numbers and percentage. There were 12 faults with error type mistake, 50% of the overall responses were correctly chosen while other 50% was wrongly chosen for type mistake. Fault 4 is interesting because the error type can be both slip/lapse as per Dr. Bradshaw. The subjects who answered it correctly have chosen Slip. Thus, the overall accuracy for Slip/Lapse is 63.33%. For Fault 9, overall accuracy is 90%, 70% for Fault 6.

Level 3 of HEAA –Analysis at Error Class Level: Based on the results from Table 1 (*Number of subjects who chose the correct Error Class (Level 3 of HEAA)),* the results shows 40% of overall average number of faults were mapped correct Error Class. The number is lesser as compared to Error Type abstraction. For error level application error, the accuracy is 44% (8/15 faults belonged to application error), lack of consistency is 60% (1/15 faults), Clerical Error/Loss of Information is 63% (2/15 faults), Wrong assumptions is 25% (2/15 faults), Information management Errors is 0% (1/15 faults), Lack of consistency is 60% (1/15 faults), and Inadequate Requirement Process is 14% (1/15 faults). The above numbers are calculated per error level type for number of subjects abstracted the error level correctly.

Overall accuracy for all the 2 HEAA levels: The last column in Table 1 shows the overall accuracy for all the 2 HEAA levels. The accuracy is calculated based on the correctness in all the two levels. Each subject who abstracted correct Error Type in second level should have abstracted correct Error Class in third level for each fault correctly to obtain the overall accuracy. The overall average is only 22%. The highest value is 60% for fault 9 whereas the lowest is 0% for fault 13. This shows the training for industrial subjects must be modified significantly.

4.2. Frequency of Error Abstraction

Human error abstraction is perspective based and depends on one's understanding the taxonomy for error abstraction that includes RE activities, error types and error class. The analysis can be confused at various stages due to over-thinking, wrong analysis or stopping the analysis at different levels. Table 2 shows the frequency of the RE activities for all fifteen faults. The expected vs selected frequencies shows that the correct RE activity is confused with others.

Table 2. Frequencies with which the RE activities were confused with each other

Selected → / Expected ↓	Elicitation	Analysis	Specification	Management	Total expected occurrences of the RE activity in the abstraction data (for all 10 subjects)
Elicitation	(0.00%)	(0.00%)	(0.00%)	(0.00%)	0
Analysis	24 (24%)	38 (38%)	31 (31%)	7 (7%)	100
Specification	5 (16.67%	4 (13.33%	17 (56.67%)	4 (13.33%)	30
Management	8 (40%)	3 (15%)	5 (25%)	4 (20%)	20

The calculation of frequency is based on number of times all the ten subjects should have selected the correct RE activity for all 15 faults. Hence the actual number of analysis occurrence would be 100, specification would be 30 and management would be 20, totaling to 150 (i.e.10 subjects' times 15 faults). Correctly reported numbers for analysis is 38/100 while it was confused to be elicitation 24 times, specification 31 times and management 7 times. Thus, the subjects confused analysis to a different activity 62 times. Also, specification was correctly abstracted only 17 times out of 30 occurrences yielding ~57%. The most difficult abstraction was management which was abstracted correctly only in 20% of cases.

Table 2 also shows that specification was the activity that was frequently selected instead of other activities. There were 38 cases where specification was chosen instead of analysis or management. Table 1 shows 90% of the people abstracted correctly for specification for fault 9. This shows that training materials and instrumentation should be improved for other RE activities such as management and analysis for Error Abstraction. For industry set up, domain expertise and technology expertise can be included as part of training. For example, the software professionals who were involved in this experiment focuses on Dynamics 365 technology. Having training materials and SRS related to how the system behavior and CRM capabilities examples might bring better understanding to the subjects.

4.3. Prevention Strategies for Error Prevention

We wanted to validate the need for HET in industry and asked 2 questions to understand the needs.

Q1. Have you encountered this type of error?

Options: Yes/No

Q2: How would you reduce the future occurrence of this error?

We collected the data and analyzed it for each fault. Below is the table (Table 3.) which contains the total number of people who had answered Yes/No. The overall response for yes was 78% for 15 faults for 10 subjects. The overall response rate was calculated for 150 responses (15 faults * 10 subjects). We also calculated the overall percentage of people who experienced slip, lapses and mistakes for all the faults (Table 4.) and the results were as follows: Percentage of subjects that experienced Slips is 82%, Percentage of subjects that experienced Lapses is 100%, and Percentage of subjects that experienced Mistakes is 72%. Based on the data obtained, we concluded that a large percentage experience errors and educating the subjects on HET will probably reduce the error experience and result in quality improvements in RE documents.

Table 3. Responses for errors encountered for faults

Faults ↓ \ Subjects →	1	2	3	4	5	6	7	8	9	10
1	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
2	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
3	Yes	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes
4	Yes	No	No	Yes	Yes	Yes	No	No	Yes	Yes
5	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
6	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
7	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	Yes
8	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
9	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes
10	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	No Answer
11	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
12	Yes	No	Yes	No Answer	Yes	Yes	Yes	Yes	No	No
13	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
14	No Answer	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
15	Yes	No	Yes	Yes	Yes	No	No	Yes	No	No

Table 4. Overall response for errors encountered for error types

Slips		Lapses		Mistakes	
Yes	No	Yes	No	Yes	No
54	12	9	0	52	72

For Q2 analysis, we categorized the quantitative data that we obtained to 9 categories into problem areas. The categories are provided in table (Figure 8. – Color coded for each classification) below. The color codes are provided for easy classification of responses classified under error class selected by the subjects for each fault. The categories were obtained by validating the most common occurrence of words in the solutions provided for resolving the problem and most suitable category that can be built by using the words. We included the error level that was selected by the subject and calculated prevention techniques. Out of 150 responses (15 faults abstracted by 10 subjects = 150 responses), 8 were blank/no response/subjects did not know what to answer. Also 12 responses out of 150 could not be resolved because they were either single worded or did not fall into any of the categories. We ignored 12 miscellaneous and 8 unanswered from analysis and discussion of results.

CLASSIFICATION BASED ON ERROR LEVEL

ERROR TYPE → / CATEGORIZATION ↓	Clerical Error	Accidentally overlooking requirements	Lack of consistency in Requirements Specifications	Loss of information from stakeholders	Problem Solution Errors	Inadequate Requirements Process	Environment error	Application Error	Wrong assumptions	Syntactic Error	Information Management error	Lack of awareness of sources of requirements	Mistaken Belief
No proper tools for RE and review	25	2	9	0	2	7	2	9	8	4	0	1	1
No centralized system for validating data and requirements	4	2	7	0	1	1	0	1	2	1	0	0	0
No impact analysis, requirement traceability and critical path analysis	8	2	3	0	2	0	0	0	0	1	0	1	0
Not documenting Use Case Scenarios	1	0	0	2	0	0	0	3	0	0	0	0	0
No subject matter expert	2	1	0	0	3	0	0	5	0	0	0	0	0
No UML's/Flowcharts	0	2	0	0	1	1	0	0	0	0	1	0	0
No time frame for RE freeze	1	0	0	0	0	0	0	0	0	0	1	0	0

OVERALL CATEGORIZATION

No proper tools for RE and review	No centralized system for validating data and requirements	No impact analysis, requirement traceability and critical path analysis	Not documenting Use Case Scenarios	No subject matter expert	No UML's/Flowcharts	No time frame for RE freeze	Miscellaneous	Unanswered
70	19	17	6	11	5	2	12	8

Figure 8. Error level and overall categorization

In Figure 8, we identified 70 (70 out of 130 responses = 53%) out of 130 responses belonged to 'no proper tools for RE and review' or hereafter referred as category 1. Based on this, we identified that the problem lies in requirements engineering where almost all the identified errors occur. We also found that clerical error was the most selected error level (41 out of 130 responses = 31%) which was tied to only one error type - slip. In clerical error category, nearly 60% (25/41 responses for clerical error) belonged to category 1. This provided a result that there are slips occurring during requirements engineering, which needs to be addressed. One way to resolve this is to bring to notice the ripple effects that occurs and educate subjects on HET to avoid such happenings. 20% of the problem area was missing impact analysis/traceability. Other 20% belonged to all other categories.

Significant results were also seen for error-levels: Lack of consistency in requirement specification (tied to slips) and application error (tied to mistakes). Nearly 14% of the overall solutions was provided for both the error-levels. Nearly 50% of the problem area, resolved from the solutions provided, occurred in Category 1 – No proper tools for RE and review. 36% of the problem area for Lack of consistency of requirement specification occurred because of not having centralized place for validating the variables and formulas used, other 14% of the problems were due to missing impact analysis/traceability. For application error – which belonged to mistake – 27% of the problematic areas were due to lack of understanding of the system or absence of SME.

Error-Levels such as Wrong Assumptions (Mistakes), accidentally overlooking requirements (Lapses), Inadequate Requirements Process (Mistakes), Problem Solution Errors (Mistakes) contributed to nearly 7% each. Majority of the problem area was category 1 – No Proper tools for RE and review.

The prevention techniques provided closely related to HET education and following SDLC principles for RE. There is a substantial requisite for RE process improvement, avoiding errors by identifying concerns in RE, conduct review, have both functional and technology expertise. The essence of identifying the root causes to avoid such errors is the essence of HET. Thus, we believe that educating on HET is very important in industry. However, based on previous results/hypothesis, teaching methodologies should be improved or be different in industry.

4.4. Discussion of Results

In this section, we will discuss the error results and opportunity to improve the experiment.

Section 4.1 analyzes the accuracy of error abstraction found by subjects (participated in experiment) on PGCS SRS, developed by someone who did not belong to the same software company where the subjects worked. The accuracy of error abstraction depends upon sequential correct option selection in the analyzed levels. The results showed that subjects were confused with the definitions of error type and error class. Overall accuracy achieved for 15 faults by 10 subjects at Level 2 (Error Type) of HEA was 54% and at Level 3 (Error Class) of HEA was 40%. Overall accuracy for 150 responses (15 faults abstracted by 10 subjects) was 22%.

Section 4.2 analyzes the frequency at which the subjects were confused between choosing the correct human error to incorrect chosen one. The analysis done on all the 3 levels for 150 responses (15 faults abstracted by 10 subjects), showed that the subjects needed additional inputs to perform error abstraction. Thus, in both sections, we identified an opportunity to improve HET education in industry.

Also, an oral discussion happened when few subjects mentioned that the examples provided during the slip, lapse and mistake was difficult to make a correct judgement. Example that was discussed mostly was driving into the garage door to explain lapse. An example was included as part of training to explain the definitions for error types. The goal was to go to grocery store. The below planning steps included to achieve the goal; (A) Start the Car by inserting the car key (B) Put transmission in reverse. (C) Back down the driveway (D) Navigate the route to store. We explained that forgetting to back down is a lapse (forgetfulness), hence the car can run and hit the garage door (assuming the car is parked in front of the garage door). However, many subjects argued that the car hitting the garage door is carelessness and should be slip. Since this is cognitive psychology, each interpretation was different. At least 3 sets of different examples and allowing sometime to discuss was requested as part of experiment. The experiment was considered long and it was mentioned that it should happen in several steps, highlighting faults at each step so that the purpose of correct abstraction and understanding is achieved.

Section 4.3 provided us with both qualitative and quantitative results on percentage of subjects experiencing the error and prevention strategies mentioned to overcome the error. Percentage of subjects who experienced the error was relatively more. The prevention strategies were categorized at level 3 (error class). The strategies helped us understand that there is a need to educate the subjects on how to perform the requirements engineering process by utilizing HET to avoid error occurrence and improve software artifacts. Since, we categorized the prevention strategies based on error level, we could gather results which leaned towards having strong requirements engineering and review process, having proper subject matter expertise and including a correct software development process. This helped us understand that there is a scope for education in industry on SDLC process improvements, utilizing right resources, educate on frequently occurring error to get quality products. Software process should focus on step by step inclusion rather than including a big requirement to gather requirements to release a minimum viable product.

5. THREATS TO VALIDITY

The study faces the following threats

(1) All the participants were introduced to cognitive concepts such as slips, lapses and errors for the first time. We addressed this threat by providing same training video and materials required for the experiment.

(2) The participants were diverse in nature with respect to their backgrounds and experience which might contribute to variability in analysis.

(3) The study focused on real time professionals than students in classroom setting. This is an initial setup and this threat can be validated by providing real project SRS samples and including trainings and examples for real requirements.

(4) We did not include any faults for elicitation, thus failing to understand the actual accuracy and frequency levels for elicitation abstraction.

6. CONCLUSION AND FUTURE WORK

The major focus of the study was to understand the accuracy at which industrial professionals will be able to abstract the faults and where the misjudgment happens. This will help to instrument the training and SRS for industrial experiment to validate error abstraction.

Analysis for Research Question 1 (section 4.1) shows selecting incorrect error type renders incorrect error level abstraction. The results of the experiment yielded very low overall accuracy for all 2 levels of HEAA. However, the subjects who mapped Error Type correctly could map to correct error class to a greater extent.

Further analysis for RQ1 (section 4.2) shows that number of cases where the subjects were confused to choose correct RE activity for analysis and management was lesser compared to choosing to the other ones. However, for specification correctness was 57% as compared to 43% of incorrect cases. Since elicitation was not included in any faults, we could not analyze the numbers for elicitation. Based on analysis for RQ2, out of 3 RE activities, number of cases of choosing incorrect activities was more for both analysis and management (2/3 RE activities). The comparison shows people were confusedly chose specification instead of management or analysis. This shows the training materials must be improved to train on abstracting errors for analysis and specification.

RQ2 and RQ3 was analyzed in section 4.3. Based on the results obtained, number of people who encountered errors in slips, lapses and mistakes was respectively 82%, 100% and 72%. We concluded that the industry professionals have admitted that they are encountering human errors and the resolution provided by them to improve RE process and include SME is base for reducing human error by educating on HET. Additional methods and processes such as educating and utilizing right SDLC process, recording meeting to avoid forgetfulness, being aware of the system, technology and product can help industry professionals build quality requirement specification document.

Our future work aims at conducting retrospective analysis to validate if the professionals will be able to identify the errors further in the SRS document. Also identifying the differences for the same SRS document in academic and industrial areas can lead to identifying if education and teaching methods must be improved in both areas or if academia performs better, we can identify what should be improved in industry for strengthening the quality of requirements.

REFERENCES

[1] IEEE Std 610.12-1990, IEEE standard glossary of software engineering terminology. 1990.

[2] Lezak, M., Perry, D., and Stoll, D. "A Case Study in Root Cause Defect Analysis". In Proceedings of the 22nd International Conference on Software Engineering. Ireland. 2000. p.428-437.

[3] Lanubile, F., Shull, F. and Basili, V.R. 1998. Experimenting with Error Abstraction in Requirements Documents. In Proceedings of the 5th International symposium on software metrics (Bethesda, MD, USA, 1998).

[4] Boehm, B. and Basili, V.R., "Software Defect Reduction Top 10 List." IEEE Computer, 2001. 34(1): 135-137.

[5] Fairley, R. Software Engineering Concepts, McGraw-Hall, New York, 1985

[6] G. M. Schneider, J. Martin, and W. Tsai. An experimental study of fault detection in user requirements documents. ACM Transactions on Software Engineering and Methodology, 1 (2): 188- 204, April 1992.

[7] Walia, G.S., Carver, J., “Development of Requirement Error Taxonomy as a Quality Improvement Approach: A Systematic Literature Review,” Department of Computer Science and Engineering, Technical Report, MSU-070404, Mississippi State University, 2007.

[8] Walia, G.S., Carver, J., Philip, T., “Requirement Error Abstraction and Classification: An Empirical Study,” In Proceedings of the 5th International Symposium on Empirical Software Engineering, Rio de Janeiro, 2006, pp. 336-345.

[9] Walia, G.S., “Empirical Validation of Requirement Error Abstraction and Classification: A Multidisciplinary Approach,” M.S Thesis, Mississippi State University, MS, 2006.

[10] Huang F, Liu B. Systematically Improving Software Reliability: Considering Human Errors of Software Practitioners. In: 23rd Psychology of Programming Interest Group Annual Conference (PPIG 2011), York, UK; 2011

[11] Huang F. Software Fault Defense based on Human Errors. Ph.D., Beijing: School of Reliability and Systems Engineering, Beihang University; 2013

[12] Reason, J. 1990. Human error. Cambridge University Press.

[13] Shappell, S.A. and Wiegmann, D.A. 2000. The Human Factors Analysis and Classification System – HFACS. Security. (2000), 19.

[14] Rasmussen, J. and Vicente, K.J. 1989. Coping with human errors through system design: implications for ecological interface design. International Journal of Man-Machine Studies. 31, 5 (1989), 517–534.

[15] Rasmussen J. (1993). Skills, rules, and knowledge: signals, signs and symbols and other distinctions in human performance models. IEEE Transactions: Systems, Man & Cybernetics, 1983, SMC-13, pp.257-267. ISSN: 0018-9472.

[16] Norman Donald A (1988). The psychology of everyday things. Basic Books (New York, USA), 1988. ISBN: 0465067093.

[17] G. S. Walia and J. C. Carver, "Using error abstraction and classification to improve requirement quality: Conclusions from a family of four empirical studies," Empir. Softw. Eng., vol. 18 (4), pp. 625–658, 2013.

[18] V. Anu, G. Walia, G. Bradshaw, W. Hu, and J. C. Carver, "Usefulness of a Human Error Identification Tool for Requirements Inspection: An Experience Report," in 23rd International Working Conference on Requirements Engineering: Foundation for Software Quality, 2017.

[19] V. Anu, G. S. Walia, W. Hu, J. C. Carver, and G. Bradshaw, "Using A Cognitive Psychology Perspective on Errors to Improve Requirements Quality: An Empirical Investigation," in IEEE 27th International Symposium on Software Reliability Engineering (ISSRE'16), 2016.

[20] V. Anu, G. S. Walia, W. Hu, J. C. Carver, and G. Bradshaw, "Error Abstraction Accuracy and Fixation during Error-based Requirements Inspection," in 27th IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW '16)., 2016.

[21] Anu, V., Walia, G., Hu, W., Carver, J., and Bradshaw, G. "Issues and Opportunities for Human Error-based Requirements Inspections: An Exploratory Study", ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM 2017.

[22] G. S. Walia and J. C. Carver. A systematic literature review to identify and classify software requirement errors. Information and Software Technology, 51(7):1087–1109, 2009

[23] Anu, V., Walia, G., and Bradshaw, G. "Incorporating Human Error Education into Software Engineering Courses via Error-based Inspections", 48th ACM Technical Symposium on Computer Science Education, SIGCSE 2017. March 8 – 11, 2017

[24] J. Reason, Error management in aircraft engineering: a manager's guide, British Airways Engineering, Heathrow, 1994

[25] J. Bowers, J. Pycock, Talking through design: requirements and resistance in cooperative prototyping, Proceedings of CHI'94, Boston, MA, 1994, ACM Press, New York, pp. 299–305

[26] J.A. Goguen, Social issues in requirements engineering, Proceedings of the IEEE International Symposium on Requirements Engineering: RE'93, San Diego, CA, 1993, IEEE Computer Society Press, New York, pp. 194–195

[27] M. Jirotka, J.A. Goguen (Eds.), Requirements Engineering: Social and Technical Issues, Academic Press, London, 1994

[28] P. Quintas (Ed.), Social Dimensions of System Engineering: People, Processes, Policies and Software Development, Ellis Horwood, London, UK, 1993.