

IMPLICIT THINKING KNOWLEDGE INJECTION FRAMEWORK FOR
SOFTWARE REQUIREMENTS DOCUMENTATION IN AGILE
METHODOLOGY

KAISS ALI ABD ELGHARIANI

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy

Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia

JUNE 2019

DEDICATION

To my beloved Mother, Father, Brothers, Sisters, Friends and my country Libya.

ACKNOWLEDGEMENT

All thanks and prayers are due to Allah who has assisted me to complete this whole project.

First of all, I would like to express my sincerest appreciations to my beloved Supervisors, Assoc. Prof. Dr. Nazri Kama, Dr. Nurulhuda Firdaus and Dr. Nur Azaliah Binti Abu bakar for their greatest support, recommendations and advices throughout working on this project. I would like to thank all Razak Faculty lecturers who have taught me the best practices of Software Engineering and their guidance and acknowledgement, which they have provided and shared with me.

I would like to express my fully gratitude to Mr. Naser Abunaama and Mr. Osama Alhoush for their encouragement to proceed with my PhD study, also, Dr. Abdulmajid Hussain for his ideas, advices, guidance and encouragement in completing this research. My sincere appreciation also extends to, Mr. Ismaiel, Dr. Ibrahim and Dr. Mohammed Taha who have provided assistance at various occasions.

Last but not least, special thanks to my beloved parents and brothers and sisters and friends for their moral and spiritual support. May Allah bless them.

ABSTRACT

Software engineering is knowledge-intensive work, and how to manage software engineering knowledge has received much attention. Agile is a common software development methodology among software developers. Requirements documentation (RD) is a challenging task for agile software developers. The existing agile RD does not incorporate the implicit thinking knowledge with the values it intends to achieve in the software project. Moreover, there is no clear framework that incorporates the implicit thinking knowledge of software developers. Therefore, this study developed a framework for Injecting the Implicit Thinking Knowledge in Agile Requirements Documentation (IITKARD). In doing so, a systematic literature review was conducted to identify the challenges of agile requirements engineering from 28 primary studies. A survey administered to 25 software engineering experts was conducted to ascertain the identified challenges of agile requirements engineering. Responses from the experts highlighted that implicit thinking knowledge in agile requirements documentation as one of the challenges. An evaluation was conducted to validate and verify the proposed IITKARD framework using an experiment based on focus group of 10 experts. The feedback from the experts indicated that the injecting of the implicit thinking knowledge in agile RD is important. The experiment with the experts in agile software engineering was carried out to validate and verify the IITKARD and its prototype tool by using two measurement aspects, which were efficiency and usability. The results obtained from the experiment showed that IITKARD was able to assist the experts to inject the implicit knowledge in agile RD measured in efficiency and usability. In addition, the results showed that the IITKARD framework achieved the highest level of experts' satisfaction. In conclusion, this research contributes to developing the IITKARD, which assists the software developers in injecting their implicit thinking knowledge in agile requirements documentation.

ABSTRAK

Kejuruteraan perisian adalah pengetahuan intensif, dan bagaimana untuk mengurus pengetahuan berkaitan kejuruteraan perisian telah mendapat banyak perhatian. Kaedah Agile adalah salah satu metodologi perisian yang biasa dalam kalangan para pembangun perisian. Dokumentasi keperluan (RD) adalah tugas yang mencabar untuk pembangunan perisian Agile. Agile RD yang sedia ada tidak menggabungkan pengetahuan pemikiran tersirat dengan nilai-nilai yang ingin dicapai dalam sesuatu projek perisian. Lebih-lebih lagi, tidak ada kerangka yang jelas yang menggabungkan pengetahuan pemikiran tersirat terhadap pembangunan perisian. Oleh itu, kajian ini telah membangunkan satu rangka kerja untuk Menyuntik Pengetahuan Pemikiran Tersirat dalam Dokumentasi Keperluan Agile (IITKARD). Dengan berbuat demikian, *Systematic Literature Review* (SLR) dijalankan untuk mengenal pasti cabaran-cabaran dalam kejuruteraan keperluan Agile daripada 28 kajian utama. Tinjauan turut dilakukan kepada 25 pakar kejuruteraan perisian yang dijalankan untuk menentukan cabaran kejuruteraan keperluan Agile. Maklum balas daripada pakar menekankan bahawa pengetahuan pemikiran yang tersirat dalam Agile RD sebagai salah satu cabaran. Penilaian telah dijalankan untuk mendapatkan keesahan kerangka kerja IITKARD yang dicadangkan dengan menggunakan eksperimen berdasarkan 10 pakar dari kumpulan berfokus. Maklum balas daripada pakar menunjukkan bahawa penyuntikan pengetahuan pemikiran tersirat dalam dokumentasi keperluan Agile adalah penting. Eksperimen dengan pakar kejuruteraan perisian Agile dilakukan untuk mendapatkan keesahan ke atas IITKARD dan alat prototaipnya dengan menggunakan dua aspek pengukuran, iaitu yang terdiri daripada kecekapan dan kebolehgunaan. Hasil daripada eksperimen ini menunjukkan bahawa IITKARD dapat membantu para pakar untuk menyuntik pengetahuan tersirat dalam RD Agile yang diukur berdasarkan kecekapan dan kebolehgunaan, di samping itu, keputusan menunjukkan bahawa rangka kerja IITKARD berjaya mencapai tahap tertinggi kepuasan dalam kalangan pakar. Sebagai kesimpulan, penyelidikan ini dapat memberi sumbangan kepada pembangunan IITKARD yang membantu pembangun perisian dalam menyuntik pengetahuan pemikiran tersirat dalam kalangan pembangun perisian bagi dokumentasi keperluan Agile.

TABLE OF CONTENTS

	TITLE	PAGE
	DECLARATION	iii
	DEDICATION	iv
	ACKNOWLEDGMENT	v
	ABSTRACT	vi
	ABSTRAK	vii
	TABLE OF CONTENTS	viii
	LIST OF TABLES	xiv
	LIST OF FIGURES	xvi
	LIST OF ABBREVIATIONS	xviii
	LIST OF APPENDICES	xix
CHAPTER 1	INTRODUCTION	1
1.1	Overview	1
1.2	Background of the Research	5
1.3	Motivation of the Research	6
1.4	Statement of the Problem	6
1.5	Research Questions	7
1.6	Research Objectives	8
1.7	Scopes of Research	8
1.8	Significance of the Research	9
1.9	Organization of the Thesis	10
CHAPTER 2	LITERATURE REVIEW	13
2.1	Introduction	13
2.2	Knowledge Management	14

2.2.1	Knowledge Management Activities	15
2.3	Knowledge Management in Software Engineering	17
2.3.1	KM activities in SE	19
2.3.2	Knowledge Management Issues and Challenges	21
2.4	Software Development Methodologies	22
2.4.1	Waterfall	23
2.4.2	Rational Unified Process (RUP)	24
2.4.3	Agile Methodology	27
2.4.4	Summary of Software Development Methodologies	32
2.5	Agile Requirements Engineering	34
2.5.1	Agile Requirements Engineering Practices	40
2.5.2	The Limitation of Traditional RE that are Resolved by Agile RE	46
2.5.3	The Practical Challenges of Agile Requirements Engineering	51
2.5.4	Tools Support Agile Requirements Engineering	57
2.5.5	Agile Requirements Documentation	64
2.6	Implicit Thinking Knowledge in Agile Methodology	65
2.7	Analysis on the Need of Implicit Thinking Knowledge in Agile RE	71
2.7.1	Demographic Analysis (Q1 – Q6)	71
2.7.2	Respondent Views on Agile Methodology (Q7– Q13)	71

2.8	Research Gap in Agile Implicit Thinking Knowledge Frameworks and Models	76
2.9	Summary	79
CHAPTER 3	RESEARCH METHODOLOGY	81
3.1	Introduction	81
3.2	Conceptual Framework	81
3.3	Research Roadmap	84
3.4	Operational Framework	86
3.4.1	Phase 1: Systematic Literature Review	88
3.4.1.1	Review Process	88
3.4.1.2	Planning the review	88
3.4.1.3	Conducting the review	92
3.4.1.4	Writing the review	95
3.4.2	Phase 2: Research Design and Procedure	96
3.4.3	Phase 3: Data Collection	97
3.4.3.1	Primary Data	97
3.4.3.2	Secondary data	98
3.4.3.3	Research Instrument	98
3.4.3.4	Questionnaire	98
3.4.3.5	Questionnaire Design	100
3.4.3.6	Survey Validity	101
3.4.3.7	Pilot study and Reliability Test	102
3.4.3.8	Data Analysis	103

3.4.4	Phase 4: Framework and Prototype Tool Development	103
3.4.4.1	Framework Construction	104
3.4.4.2	IITKARD Prototype Tool Development	105
3.4.5	Phase 5: Findings and Evaluation	105
3.4.5.1	Experts' Evaluation Objectives	106
3.4.5.2	Evaluation Procedures	106
3.4.5.3	Focus Group Experiment	107
3.4.5.4	Evaluation Methodology and Principles	109
3.4.5.5	Evaluation Metric	110
3.4.5.6	Evaluation Design	110
3.4.5.7	Experimental Setting	111
3.4.5.8	First Experiment: Efficiency of IITKARD	112
3.4.5.9	Second Experiment: Usability of IITKARD	113
3.4.5.10	Experiment's Data Analysis process	114
3.4.6	Phase 6: Reporting	115
3.5	Summary	115
CHAPTER 4	IITKARD FRAMEWORK AND ITS PROTOTYPE TOOL	117
4.1	Introduction	117
4.2	IITKARD Framework	117
4.2.1	Step 1: Create User Story Card	122
4.2.2	Step 2: Admin First Argument	125
4.2.3	Step 3: Injecting the Implicit thinking of Software Engineers	127

4.2.4	Step 4: Documenting the Implicit Thinking of Software Engineers	130
4.3	IITKARD Framework Prototype Tool Development	132
4.3.1	Step 1: Create User Story	133
4.3.2	Step 2: Set Admin First Argument	134
4.3.3	Step 3: Set Team Members Arguments	135
4.3.4	Step 4: Documenting the Arguments of Each User Story	136
4.4	Summary	137
CHAPTER 5	IITKARD FRAMEWORK EVALUATION AND RESULTS	139
5.1	Introduction	139
5.2	The Experiment Findings on the Execution of the IITKARD Tool	139
5.3	Experiment Results	141
5.3.1	Reliability Analysis	141
5.3.2	Demographic of the Experiments' Participants	142
5.3.3	Efficiency Experiment Results	142
5.3.3.1	Descriptive Statistics Analysis on the Efficiency of IITKARD	142
5.3.3.2	Qualitative Analysis of the Feedback on the Efficiency of IITKARD	146
5.3.4	Usability Experiment Results	155
5.3.4.1	Descriptive Statistics Analysis on the Usability of IITKARD	156

5.3.4.2	Qualitative Analysis of the Feedback on the Usability of IITKARD	158
5.4	Discussion	164
5.4.1	The Efficiency of IITKARD Framework	164
5.4.2	The Usability of IITKARD Framework and Prototype Tool	166
5.5	Threats to Validity	166
5.5.1	Conclusion Validity	167
5.5.2	Internal Validity	167
5.5.3	Construct Validity	167
5.5.4	External Validity	168
5.6	Summary	169
CHAPTER 6	CONCLUSION	171
6.1	Introduction	171
6.2	Research Summary	171
6.3	Achievement of Research Objectives	173
6.4	Contribution and Significance of Study	177
6.5	Limitations of the Research	178
6.6	Future Work	179
	REFERENCES	181
	LIST OF PUBLICATIONS	254

LIST OF TABLES

TABLE NO.	TITLE	PAGE
Table 2.1	RUP's phases description	26
Table 2.2	Comparison of software development methodologies comparisons	33
Table 2.3	Summary of the related reviews on agile requirements engineering	37
Table 2.4	Summary of practices and studies analyzed them	40
Table 2.5	Summary of the limitations of traditional RE resolved by agile methodology	47
Table 2.6	Summary of RE practical challenges in agile	52
Table 2.7	Comparison between RE tools in AM	62
Table 2.8	Gaps analysis of existing works on agile implicit thinking knowledge	78
Table 3.1	PICO Framework	88
Table 3.2	Research Strategy	90
Table 3.3	Summary of study research selection	93
Table 3.4	Quality Assessment	94
Table 3.5	Reliability test for the pilot study	103
Table 3.6	The mapping between research objectives and research processes and procedures details of operational framework	104
Table 3.7	Experiments Evaluation	111
Table 3.8	Structure of question for the first experiment	113
Table 3.9	Structure of questions for the second experiment	114
Table 4.1	Agile techniques for identifying Requirements	123
Table 4.2	User story attributes	123
Table 4.3	User Story Card	124
Table 4.4	Type of arguments	126
Table 4.5	Example of user story card	134
Table 4.6	Example of set argument	135
Table 5.1	Experiment Procedure	140
Table 5.2	Reliability test results	142
Table 5.3	Feedback of the fastness of converting the implicit thinking knowledge to documentation	147

Table 5.4	Feedback on the fastness of IITKARD in displaying the implicit thinking knowledge of software engineers	148
Table 5.5	Feedback of the overall results of cost effort of the using IITKARD framework	150
Table 5.6	Feedback on coordinating agile RE	151
Table 5.7	Feedback on mentoring agile RE	153
Table 5.8	Feedback on supporting the software developers	154
Table 5.9	Feedback on the difficulties of IITKARD prototype tool	159
Table 5.10	Feedback on completing the tasks successfully	160
Table 5.11	Feedback on the overall usability of IITKARD framework	161
Table 5.12	Feedback on the limitation of IITKARD	163

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 2.1	Literature Review Structure	13
Figure 2.2	Waterfall Model (Royce, 1970)	24
Figure 2.3	Disciplines and phases of RUP model (Kruchten, 2004)	25
Figure 2.4	Delivery process in agile methodology (Tran, 2012)	28
Figure 2.5	Redmine project dashboard (Sarkan <i>et al.</i> , 2011)	58
Figure 2.6	Redmine user story screen (Sarkan <i>et al.</i> , 2011)	59
Figure 2.7	JIRA task board view (JIRA Website)	60
Figure 2.8	JIRA user story description (JIRA Website)	61
Figure 2.9	Tacit Knowledge Enablers (Kavitha and Irfan, 2011)	68
Figure 2.10	Proposed framework for capturing tacit knowledge in agile methodology (Kavitha and IRfan, 2011)	69
Figure 2.11	The bar chart of the overall results of the respondents not applying agile methodology	72
Figure 2.12	The bar chart of the overall results of the respondents applying agile methodology	72
Figure 2.13	The bar chart of the overall results of the challenges of agile RE	73
Figure 2.14	The bar chart of the overall results of the influence of agile RE	74
Figure 2.15	The bar chart of the overall results of user story best practice in agile RE	74
Figure 2.16	The bar chart of the overall results of the implicit thinking knowledge in agile methodology.	75
Figure 2.17	The bar chart of the overall results of the effect of the implicit thinking knowledge in the maintenance phase	76
Figure 3.1	Conceptual Framework	83
Figure 3.2	Research Roadmap	85
Figure 3.3	Operational Framework	87
Figure 4.1	The novelty in agile methodology life cycle (Tran, 2012)	119
Figure 4.2	Conceptual view of the IITKARD framework	120
Figure 4.3	IITKARD Framework Steps	121

Figure 4.4	Step 1: Create user story	122
Figure 4.5	Step 2: Set Admin First Argument	126
Figure 4.6	Step 3: Implicit Thinking Injecting	128
Figure 4.7	Step 4: Document Implicit Thinking	130
Figure 4.8	Dashboard of the prototype tool	132
Figure 4.9	Form of user story information	133
Figure 4.10	First argument of user story	134
Figure 4.11	Add new argument	135
Figure 4.12	User story arguments	136
Figure 5.1	The bar chart of the overall results of the fastness of IITKARD	143
Figure 5.2	The bar chart of the overall results of converting the implicit thinking knowledge using IITKARD	143
Figure 5.3	The bar chart of the overall results of cost effort of the using IITKARD framework	144
Figure 5.4	The bar chart of the overall results of the acceptance level of coordinating agile RD	145
Figure 5.5	The bar chart of the overall results of the acceptance level of monitoring agile RD	145
Figure 5.6	The bar chart of the overall results of IITKARD framework Support	146
Figure 5.7	The bar chart of the overall result of the easiness of IITKARD and its tool	156
Figure 5.8	The bar chart of the overall results of IITKARD completed tasks successfully	157
Figure 5.9	The bar chart of the results of IITKARD overall usability satisfactory	157
Figure 5.10	The bar chart of the overall results of IITKARD usability limitations	158

LIST OF ABBREVIATIONS

AM	- Agile Methodology
ARE	- Agile Requirements Engineering
ARD	- Agile Requirements Documentation
CRC	- Class Responsibility Collaboration
GSD	- Global Software Development
IITKARD	- Injecting of Implicit Thinking Knowledge in Agile Requirements Documentation
JAD	- Joint Application Development
LD	- Lean Development
RAD	- Rapid Application Development
RE	- Requirements Engineering
RUP	- Rational Unified Process
SDLC	- Software Development Life Cycle
SLR	- Systematic Literature Review
SPM	- Software Project Manager
TC	- Task Card
UML	- Unified Modeling Language
US	- User Stories
UTM	- Universiti Teknologi Malaysia
XP	- Extreme Programming

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
Appendix A	Survey Validation Content Form	198
Appendix B	Summary of Validation Content Form	201
Appendix C	Guideline of Using IITKARD Framework	202
Appendix D	Experimentation Briefing	213
Appendix E	SGIS Data Used for Implementing the IITKARD	219
Appendix F	The Outcome of IITKARD Framework	230
Appendix G	The Experiment Instrument	235
Appendix H	Initial Data Demographic Analysis	241
Appendix I	Focus Group Demographic Analysis	246
Appendix J	List of Publication	254

CHAPTER 1

INTRODUCTION

1.1 Overview

The most significant value in an organization is the knowledge, which affecting its competitiveness (Amin *et al.*, 2017). Adopting knowledge management system is assists organizations to capture knowledge and share it to all their members. In fact, software development is highly requested. To achieve delivering the software product requiring to concentrate on better quality and better productivity (Martin-de Castro, 2015). To meet these achievement, software organizations have tried to use one of its most significant resource, which is the structured software engineering knowledge (Sandra *et al.*, 2017). Previously, this knowledge was recorded on paper or kept in people's mind. This makes it difficult to be accessible. Moreover, the knowledge that kept in peoples' brains is immediately lost when individual leave (Jurado *et al.*, 2018). Furthermore, it is hardly to capture knowledge about some matter in large organizations. However, knowledge has to be systematically captured and stored in a corporate repository, and then can be easily shared among the organization members (Chugh *et al.*, 2019). To make knowledge sharing in practice, knowledge should be acquired knowledge from organizations' members and formalize it to be available on structured level. In this context, knowledge management systems can be very beneficial (Shim and Lee, 2017).

KM incorporates human resource, structural organization, and the information technology (Nonaka and von Krogh, 2009). Moreover, KM also includes methods and tools, which support its process (Ahmad, 2018). KM systems facilitate processes of knowledge creation and retrieving. KM systems aim to increase knowledge development and sharing. In software engineering, KM can be implemented to capture the knowledge and experience created during the software development lifecycle (Wang and Noe, 2010). Though there is a variety among

software projects, but experiences might be similar and assist software developers to practice their activities. Reusing knowledge might also help to avoid past failures repetition and provide the solution of frequent problems. The integration of software process and knowledge management systems make it more effective. Basically, Software Engineering Environments incorporate many tools supporting software engineering activities during software development lifecycle (Amritesh and Misra, 2014), so KM can be easily integrated in a SEE.

Agile is a software development methodology, which provides a teamwork support, face-to-face communication, customer collaboration and quick delivery (Rizvi, 2013). Agile development derived from the agile manifesto stated on 2001 by designers of agile approaches, such as Extreme Programming (XP), Scrum, Crystal and Dynamic Systems Development Method (DSDM), and other experts in software industry (Harris, 2006; Kapuruge *et al.*, 2010). Agile manifesto started by including a common set of significant standards and features for all of agile methodology approaches (Jin-Hua *et al.*, 2008). The manifesto includes four main standards to improve the effectiveness of teamwork, such as the interactions among team members, a quick software deliver, managing any unexpected changes, and customer involvement (Karlsen, Hagman and Pedersen, 2011). These features are included in each agile methodology approaches with a slightly different of each one, but all of agile approaches have its own process (Sandra *et al.*, 2017).

Agile Methodology has gradually improved in software engineering best practices. Recently, developers are looking for more flexibility to develop software systems, which can provide efficient services to their customers (Ernst et al, 2013). However, agile approaches are mostly having the same practices such as user story cards, face-to-face communication, iteration and user collaboration. Indeed, many software development models are designed to assist developers to build their software effectively (Kettunen, 2010). Agile methodology has its own features with regard to the concentration of productionizing the software system, starting from the first phase to the end phase. This includes the considering of iterations of each phase, and the small releases of the product that can make early product releases (Asghar *et al.*, 2017).

According to Martakis *et al.* (2013) Software Development Methodologies (SDM) have dissimilarities with agile methodology; agile methodology emphasizes the influence of software developers and clients who are playing significant roles in agile software development process. Using traditional software development methodologies, users mostly do not largely contribute in software development practices (Nerur *et al.*, 2005). However, customers in agile contribute with software developers as effective team members. For example, clients and developers together outline the system structures for software development lifecycle implementation. Boehm and Turner (2004) stated that agile methodology has changed the role of users who are supposed to be collaborated, represented, committed, and well-informed.

The aim of adopting agile methodology is to avoid common heaviness software development practices used in traditional software development methodologies, and to support software requirements changes management and fast product delivery (Erickson, Lyytinen and Siau, 2005). The philosophy of agile methodology approaches is to deliver software working editions in short iterations, then upgrade the edition of the software based on customers' feedback (Karlsen, 2011). By accepting requirements changes, quicker development, and clients will get the system they need (Hannay *et al.*, 2003). Therefore, agile methodology includes several common approaches such as Extreme Programming, SCRUM, Crystal methodologies family, Feature-Driven Development and Adaptive Software Development.

Software requirements engineering (RE) is the early practice of software development lifecycle (Karlsson, 2007; Panian, 2009). This practice identifies the user's requirements, which involves customer and developers' deliberations (Hurtado, 2013). The requirements engineering goal is to provide complete, unambiguous software project requirements (Talbot and Connor, 2011). An individual software requirement can be defined as a capability or a condition needed by customer to achieve software-facilitated tasks (Ghani *et al.*, 2014). Meanwhile, the requirements engineering process concerns about the identification, modeling and verification of the functionalities of a software product (V and Donn, 2009). Requirements engineering includes four main tasks, requirements elicitation,

negotiation, specification, and validation/verification (Carlson and Matuzic, 2010). There are many requirements elicitation techniques available, such as Joint Application Development (JAD) (Hughes and Cotterell, 2006), Storyboarding and Rapid Application Development (RAD) (Beynon-Davies, 2000). The objective of these techniques was to provide requirement engineers or system analysts a platform to conclude final list of requirements collaboratively. However, none of these techniques support capturing tacit knowledge and documenting the collaborative arguments held during the RE process (Inayat *et al.*, 2014).

RE is a traditional software engineering process, which includes identifying, analyzing, documenting and validating requirements for the developed software system (Liu *et al.*, 2010). In fact, more than one issue has been raised during the software development, such as requirements specification, software design, implementation and software testing (Martakis and Daneva, 2013). Scholars agree that it is difficult to manage and model unstructured elicited requirements from operational domain (Bano, 2014). Requirements need to be summarized and well-designed based on any standard requirement specification template (Donn, 2009). Besides, this assist stakeholders and maintenance team to understand requirements because it is a significant practice to be validated by stakeholders. Poor requirements specifications lead to ambiguity requirements and become difficult to understand and might be the cause of failure of software application (Ivari, 2010). Therefore, the issue of implicit thinking mismanagement forms a major threat for organizations. Though experts' know-how should be considered as part of the organizational memory, the organizations have no control on the experience knowledge kept in experts' minds (Hussain *et al.*, 20117). This is especially applicable to knowledge-intensive organizations such as software organizations. According to Hoffman *et al.*, (2008), such organizations are subjected to lose their ability to conduct business as their workforce ages and their knowledge will be lost once they leave the organization (Kang *et al.*, 2008).

A significant knowledge is usually exist during the software team deliberations. Generally, a part of shared knowledge is explicitly documented as a meeting minutes form, diagrams, test cases and other software documentations (Neves *et al.*, 2011). The explicitly documented knowledge is easy to be organized

and also can be shared easily among software team members (Jafarinezhad and Ramish, 2012). However, significant experience knowledge is still undocumented and tacitly kept in software engineers' brains (Nonaka and Krogh, 2009). This experience knowledge is categorized as an implicit thinking knowledge, which is usually observed from an orally communication. Capturing the implicit thinking knowledge has two main challenges. Firstly, it is unnoticed and secondly experts' knowledge is usually unconsciously exploit it. In other words, implicit thinking knowledge is not easy to be explained. This feature is reflected by personal knowledge of Polanyi's theory "we know more than we can tell" (Sandra *et al.*, 2017).

1.2 Background of the Research

Software development is a knowledge-intensive activity in which its success depends fundamentally on the developers' experience and skills (Kavitha and Irfan, 2011). According to Standish Group Report, one of project failure factors relate with requirements, which is simple requirements documentation (Inayat *et al.*, 2014). Agile software development has put a new focus on how to share knowledge among members of software development teams (Saini, Arif and Kulonda, 2018). In contrast to heavyweight, document-centric approaches, agile approaches rely on face-to-face communication for capturing implicit thinking of software engineers (Ahmed, 2018).

Several researches have figured out that the realization of ignoring implicit thinking documentation has led to increased interest in observing the ways in which knowledge of software engineers could be effectively determined, identified, organized and documented (Elghariani and Kama, 2016). The field of implicit thinking injection in agile requirements documentation has emerged to address this need (Shim and Lee, 2017). Therefore, this research aims to provide a framework to inject implicit thinking knowledge in agile methodology. This framework is supported by a prototype tool to assist software developers to understand and analyze the requirements.

1.3 Motivation of the Research

Given the range and variety of software methodologies, it is becoming increasingly to adopt one of agile approaches. This adoption has shown the significance of involving the client while developing the software. Moreover, agile has mentioned clearly that it could accept any addition features to its practices without losing the term of agility (Srifastava, Bhardwaj and Sarswat, 2017).

This work on dimension and smoothness yields a variety of new understandings, which this study suggests a framework to capture and document the implicit thinking knowledge of software engineers during requirements engineering phase. These understandings have also allowed to derive the framework which can self-tune optimally to both dimension and smoothness, simultaneously at all points in the requirements engineering in agile methodology. The new framework can help and assist software developers to manage and track the implicit thinking knowledge during developing the software.

Since agile approaches have been commonly used, scholars have increased their focus on challenges of agile requirements engineering. Researchers aim to provide a framework that allows software developers to manage agile requirements documentation and helps to resolve the issue of minimal documentation in agile approaches, by providing a unified documentation including implicit thinking knowledge of agile software developers.

1.4 Statement of the Problem

The success of software development projects depends critically on knowledge quality, which software organizations apply to their development processes (Andriyani, Hoda and Amor, 2017). The significant challenge is how to capture and share this knowledge. Agile methodology implies that software developers have focused on delivering software products (Ahmed, 2018). The simplicity of agile documentation has been considered as one of agile methodology

issues (Fannoun and Kerins, 2017). However, agile requirements are usually documented in the form of user story cards and task description. Implicit rationale of software developers is almost ignored by many software methodologies including agile approaches (Kavitha and Irfan, 2011).

Explicit documentation is commonly captured in most software development methodologies (Shim and Lee, 2017). Unlike explicit, implicit thinking knowledge is always hidden and not clearly stated. It is all about software engineers thinking knowledge such as their assumptions, views, suggestions and opinions, and explicit coded documenting governed the software engineer's decisions in the software engineering process (Saini, Arif and Kulonda, 2018). The lack of such implicit knowledge experience could lead to more difficulty during the software maintenance phase (Sandra *et al.*, 2017).

A framework for injecting implicit thinking knowledge of software engineers in agile requirements documentation is proposed. The aim of this framework is to assist software engineers to manage requirements and provide unified requirements documentation including implicit thinking knowledge of each team member. Requirements engineering practices are activities that assist developers to manage requirements with recording the implicit thinking of team members by giving views, assumptions, and observations during requirements engineering practices.

1.5 Research Questions

This study aims to answer the following questions:

- (a) RQ1: What are the issues and challenges in agile requirements documentation?
- (b) RQ2: How to develop a framework that injects implicit thinking knowledge in agile requirements documentation?
- (c) RQ3: How to develop a prototype tool that can support the injection of implicit thinking knowledge in agile requirements engineering?

- (d) RQ4: How to evaluate the efficiency of the developed framework?
- (e) RQ5: How to evaluate the usability of the developed framework?

1.6 Research Objectives

The objective of this research is to explore and investigate issues and challenges of RE in Agile Methodology and to provide a solution to minimize these challenges. Overall, the objective has five parts:

- (a) RO1: To analyze issues and challenges in agile requirements documentation.
- (b) RO2: To develop a framework that injects implicit thinking knowledge in agile requirements documentation.
- (c) RO3: To develop a prototype tool to support the framework of injecting the implicit thinking of requirement engineering in agile methodology.
- (d) RO4: To evaluate the efficiency of the developed framework.
- (e) RO5: To evaluate the usability of the developed framework.

1.7 Scopes of Research

Focusing on the research area is the significance need to emphasize the boundaries and constraints of the study. The scope in this study is limited to the following:

- i. The study limited to agile software development methodology. The study only focused on Extreme programming approach (XP) as it is commonly used among agile approaches.
- ii. Since, the purpose of this study is to develop a framework for injecting the implicit thinking knowledge of software developers during agile requirements engineering, this study focused only on agile requirements documentation.
- iii. A prototype tool is developed using Microsoft Visual Studio. Net (C#) and SQL Server as programming platform to support the framework process of injecting the implicit thinking knowledge in agile requirements documentation.

1.8 Significance of the Research

This research aims to develop a framework that injects the implicit thinking knowledge in agile requirements documentation to enable software developers to manage agile requirements documentation as part of an agile software development methodology. The proposed framework helps to resolve the issue of minimal documentation by providing a unified documentation, which incorporates implicit thinking knowledge of agile software developers.

Therefore, the importance of capturing the implicit thinking knowledge of software developers during requirements engineering phase, assists to understand how software project is built up (Saini, Arif and Kulonda, 2018). In addition, the proposed framework provides the ability of software developer's community during software maintenance phase. Based on researches made by (Sandra *et al.*, 2017), few models and tools are designed and developed to support software project management phases, but there is no attention paid for the implicit thinking knowledge documentation, and that is clearly mean that providing a framework to solve this problem is needed.

1.9 Organization of the Thesis

This thesis is structured into six chapters. Chapter 1 gives an overview of the research. It begins with introducing the overview of the research, research background, which briefly introducing the agile software development methodology, agile requirements engineering and implicit thinking knowledge. Then, it describes the statement of the problem, motivation of the research, research questions, and research objectives. Then, it continues with describing the research scope and the significance of the research.

Chapter 2 discusses the literature review of the research and highlights the knowledge gaps in extant research to justify the novelty of this research. The chapter starts with a discussion of the related works in the common software development methodologies. Subsequently, the chapter describes a systematic literature review (SLR) that have been conducted in identifying related theories of Agile requirements engineering practices and the practical challenges of Agile RE. Then, the chapter discusses the tools support agile RE and the implicit thinking knowledge in agile methodology. The review mainly focuses on identifying the strength and weaknesses of the previous studies that drive to the identification of gaps to be explored.

Chapter 3 discusses the research methodology, which refers to the overall process involved in the research in fulfilling the research objectives and obtaining the expected deliverables. It starts with a discussion of conceptual framework, research, research roadmap design, and operational framework phases.

Chapter 4 discussed the results on the steps to formulate IITKARD. In addition, the IITKARD together with the prototype tool development are explained and discussed.

Chapter 5 presents the evaluation procedures and processes of IITKARD framework and its prototype tool. Also, presents the discussion of empirical findings of the evaluation process

Chapter 6 consists of the thesis summary, contribution and significance of the study, limitations of the research and the works that can be extended from the proposed IITKARD.

REFERENCES

- Achimugu, P., Selamat, A., Ibrahim, R., and Mahrin, M. N. (2014). A systematic literature review of software requirements prioritization research. *Information and Software Technology*. 2014. 56(6), 568-585.
- Agile Manifesto. <http://agilemanifesto.org/principles.html>.
- Ahmed, U. (2018). A review on knowledge management in requirements engineering. in 2018 International Conference on Engineering and Emerging Technologies (ICEET), Lahore, Pakistan, PP 1-5.
- Alenljung, B., Person, A. (2008). Portraying the Practice of Decision-making in Requirements Engineering: a Case Study of large Scale Bespoke Development. *Requirements Engineering Journal*. 2008. 13, 257–279.
- Ambler, S.W. (2002). *Agile Modeling - Effective Practices for eXtreme Programming and the Unified Process*. Wiley, New York.
- Amin, A., Basri, S., Hassan, M. F., & Rehman, M. (2017, June). A snapshot of 26 years of research on creativity in software engineering-A systematic literature review. In *International Conference on Mobile and Wireless Technology*(pp. 430-438). Springer, Singapore.
- Amritesh, & Misra, S. (2014). Conceptual modeling for knowledge management to support agile software development. *The Knowledge Engineering Review*, 29(4), 496-511. doi:10.1017/S0269888914000198.
- Andriyani, Y., Hoda, R., & Amor, R. (2017, August). Understanding Knowledge Management in Agile Software Development Practice. In *International Conference on Knowledge Science, Engineering and Management* (pp. 195-207). Springer, Cham.

- Aniche, M. and Silveira, G. (2011). Increasing learning in an agile environment: Lessons learned in an agile team. In Proceedings of the Agile Conference, Salt Lake City, UT, pp 289–295.
- Arsanjani, A., Zhang, J., Ellis, M., Allam, A., and Channabasavaiah, K. (2007). S3: A service-oriented reference architecture. *IT professional*. 2007. vol. 9, no. 3, pp. 10-17.
- Asghar, A., Bhatti, S., Tabassum, A. and Shah, S. (2017). The Impact of Analytical Assessment of Requirements Prioritization Models: An Empirical Study. *International Journal of Advanced Computer Science and Applications*. 2017. Vol. 8, No. 2.
- Augustine, S. (2005). *Managing Agile Projects*. Prentice-Hall, Englewood Cliffs.
- Azham, Z., Ghani I. and Ithnin, N. (2011). Security backlog in Scrum security practices. *Software Engineering (MySEC)*. 2011 5th Malaysian Conference in, 414-417.
- Bano, M. (2014). Aligning Services and Requirements with User Feedback. *Requirements Engineering Conference (RE)*, 2014 IEEE 22nd International.
- Bano, M., Ikram, N., and, Niazi, M. (2013). Requirements Engineering Challenges in Service Oriented Software Engineering: an exploratory online survey. *International Journal of Software Engineering* 07/2013; 6(2).
- Bano, M., Zowghi, D. (2013). Users' Involvement in Requirements Engineering and System Success. *Empirical Requirements Engineering (EmpiRE)*, 2013 IEEE Third International Workshop on Empirical Requirements Engineering (EmpiRE). DOI: 10.1007/s10664-016-9465-1.
- Barney, S., Aurum, A., Wohlin, C. (2008). A Product Management Challenge: Creating Software Product Value through Requirements Selection. *Journal of Software Architecture*. 2008. 54, 576–593.

- Bartlett, J. E., Kotrlik, J. W., and Higgins, C. C. (2001). Organizational research: Determining appropriate sample size in survey research appropriate sample size in survey research. *Information technology, learning, and performance journal*, 19(1), 43.
- Beck, K. (2000). *eXtreme Programming Explained: Embrace Change*. Addison-Wesley, Reading.
- Bettis, R. A., Ethiraj, S., Gambardella, A., Helfat, C., & Mitchell, W. (2016). Creating repeatable cumulative knowledge in strategic management: A call for a broad and deep conversation among authors, referees, and editors. *Strategic Management Journal*, 37(2), 257-261.
- Bjarnason, E., Wnuk, K. and Regnell, B. (2011b). Requirements are slipping through the gaps—A case study on causes & effects of communication gaps in large-scale software development. In 2011 IEEE 19th international requirements engineering conference (pp. 37–46).
- Bjørnson, FO. (2008). Knowledge management in software engineering: a systematic review of studied concepts, findings and research methods used. *Inform Softw Technol.* 2008. 50:1055–1068.
- Boden, A., Avram G., Bannon L., Wulf V. (2009). Knowledge management in distributed software development teams does culture matter? In: *Global Software Engineering. Fourth IEEE International Conference on*, IEEE, pp 18–27.
- Boehm, T. (2004). *Balancing Agility and Discipline: A Guide for the Perplexed*. Addison-Wesley, Boston, MA.
- Bourque, P., and Fairley, R. E. (2014). *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. Los Alamitos, CA, USA: IEEE Computer Society Press.
- Brereton, P. (2004). The software customer/supplier relationship. *Communications of the ACM*, vol. 47, no. 2, p. 81.

- Brodie, L., and Woodman, M. (2011). Prioritization of Stakeholder Value Using Metrics. In L. A. Maciaszek & P. Loucopoulos (Eds.), *Evaluation of Novel Approaches to Software Engineering* (pp. 74-88). Berlin Heidelberg: Springer.
- Cabral, AY., Ribeiro, MB., Lemke, AP., Silva, MT., Cristal, M., Franco, C. (2009). A case study of knowledge management usage in agile software projects. *International Conference on Enterprise Information Systems (ICEIS)*, pp 627–638.
- Campanelli, S., and Parreiras, S. (2015). Agile methods tailoring—A systematic literature review. *Journal of Systems and Software*, 110, 85-100.
- Cao, L., Mohan, K., Xu, P., and Ramesh, B. (2009). A framework for adapting agile development methodologies. *European Journal of Information Systems* 18, 332-343 (August 2009) | doi:10.1057/ejis.2009.26.
- Carlson, D. and Matuzic, P. (2010). Practical agile requirements engineering. In *13th Annual systems engineering conference*.
- Chau, T., Maurer, F. (2010) Knowledge sharing in agile software teams. *Logic Approx* 3075:173–183.
- Choo, CW., Alvarenga RCD (2010) Beyond the ba: managing enabling contexts in knowledge organizations. *J Knowl Manage* 14(4):592–610.
- Chugh, M., Chanderwal, N., Upadhyay, R., & Punia, D. K. (2019). Effect of knowledge management on software product experience with mediating effect of perceived software process improvement: An empirical study for Indian software industry. *Journal of Information Science*, 0165551519833610.
- Cockburn, J. H. (2001). Agile software development: the people factor: *IEEE Computer*. 34 (11), 131–133.

- Conboy, K. (2009). Agility from first principles: reconstructing the concept of agility in information systems development. *Inform Syst Res.* 20(3):329–354.
- Conboy, K., Fitzgerald, B. (2010). Method and developer characteristics for effective agile method tailoring: a study of xp expert opinion. *ACM Trans Softw Eng Methodol* 20(1).
- Corbucci, H., Goldman, A., Katayama, E., Kon, F., Melo, CO., Santos, V. (2011) .Genesis and evolution of the agile movement in brazil - a perspective from the academia and the industry. In: Proc. of 25th Brazilian Symposium on Software Engineering (Track: SBES is 25), pp 98–107.
- Creswell, J.W. (2002). *Research Design, Qualitative, Quantitative, and Mixed Methods Approaches.* Sage Publications, 2002.
- Dayan, R., Heisig, P., & Matos, F. (2017). Knowledge management as a factor for the formulation and implementation of organization strategy. *Journal of Knowledge Management*, 21(2), 308-329.
- De Lucia, A. and Qusef, A. (2003) Requirements Engineering in Agile Software Development. *Journal of Emerging Technologies in Web Intelligence* 01/2003; 2(3). DOI: 10.4304/jetwi.2.3.212-220.
- Dingsøyr, T., Nerur, S., Balijepally, V., Moe, N. (2012). A decade of agile methodologies: towards explaining agile software development. *J Syst Softw* 85(6):1213–1221.
- Donate, MJ., Canales, JI. (2012). A new approach to the concept of knowledge strategy. *J Knowl Manage.*16(1):22–44.
- Donate, MJ., Guadamillas, F. (2011). Organizational factors to support knowledge management and innovation. *J Knowl Manage* 15(6):803–814.

- Du, M. (2007). Knowledge management: what makes complex implementations successful Knowledge Management. 11(2):91–101.
- Dybå, T., Kampenes, V. B., and Sjøberg, D. I. (2006). A systematic review of statistical power in software engineering experiments. *Information and Software Technology*, 48(8), 745-755.
- Easterby-Smith, M., Lyles, MA. (2011). *Handbook of organizational learning and knowledge management*, 2nd edn. Wiley.
- Elghariani, K., and Kama. (2016). Review on Agile requirements engineering challenges, in 3rd International Conference on Computer and Information Sciences, Kuala Lumpur, Malaysia, pp. 507-512.
- Erickson, J., K. Lyytinen and K. Siau, Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research. In *Journal of Database Management*, 16(4), 2005, 88-100.
- Ernst, N. a., Borgida, A., Jureta, I. J., & Mylopoulos, J. (2013). Agile requirements engineering via paraconsistent reasoning. *Information Systems (June)*, 1–17.
- Fannoun, S., & Kerins, J. (2017). Evaluating current practice and proposing a system to enhance knowledge assets within a small software development unit. in 2018 4th International Conference on Information Management (ICIM). Oxford, UK.
- Firdaus, A., Ghani, I., and Jeong, SR. (2014). Secure Feature Driven Development (SFDD) Model for Secure Software Development *Procedia-Social and Behavioral Sciences*. 2014. 129, 546-553.
- Firdaus, A., Ghani, I., and Yasin, NIM. (2013). Developing Secure Websites Using Feature Driven Development (FDD): A Case Study. *Journal of Clean Energy Technologies Issue*. 2013. 1 vol(4).
- Fraser, S., Rising, L., Ambler, S., Cockburn, A., Eckstein, J., Hussman, D., Miller, R., Striebeck, M., Thomas, D. (2006). A fishbowl with piranhas

coalescence, convergence or divergence: The future of agile software development practices some assembly required. Proceedings of the Conference on Object Oriented Programming Systems Languages and Applications OOPSLA. pp 937–939.

Fuggetta, A., and Di Nitto, E. (2014). Software process. Paper presented at the Proceedings on the Future of Software Engineering 2014 (FOSE'14), May 31 - June 7, Hyderabad, India.

Gerald, M., Weinberg, L., Basili, C., Victor, R. (2003). Iterative and Incremental Development: A Brief History.. Computer 36.

Ghani, I., and Yasin, NIM. (2013). Software Security Engineering in Extreme Programming Methodology: A Systematic Literature. Sci.Int.(Lahore),25(2),215-221.

Ghani, I., Azham, Z., and Jeong, SR. (2014). Integrating Software Security into Agile-Scrum Method. KSII Transactions on Internet an Information Systems. 2014. (TIIS) 8 (2), 646-663.

Gottesdiener, E. (2009). AView To Agile Requirements. E EBG Consulting, Inc. [www.ebgconsulting.com, http://ebgconsulting.com/Pubs/Articles/AViewToAgileRequirements-gottesdiener.pdf](http://ebgconsulting.com/Pubs/Articles/AViewToAgileRequirements-gottesdiener.pdf).

Hannay, J., Dyba, T., Arisholm, E., Sjøberg, D. (2009). The effectiveness of pair programming: a meta-analysis. Inform Softw Technol 51(7):1110–1122.

Harris, R.S., Cohn, M. (2006). Incorporating Learning and Expected Cost of Change in Prioritizing Features on Agile Projects. In: Abrahamsson, P., Marchesi, M., Succi, G. (eds.) XP 2006. LNCS, vol. 4044, pp. 175–180. Springer, Heidelberg.

Hissen, A. (2008). Capturing Software-Engineering Tacit Knowledge. 2nd European Computing Conference.

- Hurtado, J. A., Bastarrica, M. C., Ochoa, S. F., and Simmonds, J. (2013). MDE software process lines in small companies. *Journal of Systems and Software*, 86(5), 1153-1171.
- Hussain, S. J., Rashid, K., Ahmad, H. F., & Hussain, S. F. (2007). Effective software management: where do we falter?. In *Proceedings of the 6th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems* (pp. 13–18).
- IEEE Standard for Information Technology--System and Software Life Cycle Processes--Reuse Processes. (2010). IEEE Std 1517-2010 (Revision of IEEE Std 1517-1999), 1-51. doi: 10.1109/ieeestd.2010.5551093.
- Iivari, J., Iivari, N. (2010). Organizational culture and the deployment of agile methods: the competing values model view. *Springer*, pp 203–222.
- Inayat, I., Salim, S., Marczak, B., Daneva, M., & Shamshirband, S. (2014). A systematic literature review on agile requirements engineering practices and challenges. *Computers in Human Behavior*. <http://dx.doi.org/10.1016/j.chb.2014.10.046>.
- ISO/IEC-25010. (2011). ISO/IEC 25010: 2011 Systems and software engineering— Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models.
- Jafarinezhad, O., & Ramsin, R. (2012). Development of Situational Requirements Engineering Processes: A Process Factory Approach. In *Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual* (pp. 279–288). IEEE.
- Jeners, S., Clarke, P., O'Connor, R. V., Buglione, L., and Lepmets, M. (2013). Harmonizing Software Development Processes with Software Development Settings—A Systematic Approach. In F. McCaffery, R. V. O'Connor & R. Messnarz (Eds.), *Systems, Software and Services Process Improvement* (Vol. 364, pp. 167-178). Springer Berlin Heidelberg: Springer.

- Jin-Hua, L., Chang-Jiang, W., Jing, L., and Qiong, L. (2008). Earned Value Project Management of Model-Centric Software Development. Paper presented at the 4th International Conference on Wireless Communications, Networking and Mobile Computing, 2008. WiCOM'08. , October 12-14, Dalian, China.
- Jun, L., Qiuzhen, W. (2010). Application of Agile Requirement Engineering in Modest-sized Information Systems Development. Software Engineering (WCSE), 2010 Second World Congress on (Volume: 2).
- Jurado, J. L., Garces, D. F., Paredes, L. M., Segovia, E. R., & Alavarez, F. J. (2018, October). Model for the improvement of knowledge management processes based on the use of gamification principles in companies in the software sector. In International Conference on Software Process Improvement(pp. 142-151). Springer, Cham.
- Kang, D., Song, I.-G., Park, S., Bae, D.-H., Kim, H.-K., and Lee, N. (2008). A case retrieval method for knowledge-based software process tailoring using structural similarity. Paper presented at the 15th Asia-Pacific Software Engineering Conference, 2008. APSEC'08. , December 3-5, Beijing, China.
- Kapuruge, M., Han, J., & Colman, A. (2010). Support for business process flexibility in service compositions: An evaluative survey. In 21st Australian Software Engineering Conference (pp. 97–106).
- Karlsen, JT., Hagman, L., Pedersen, T. (2011). Intra-project transfer of knowledge in information systems development firms. *J Syst Inform Technol* 13(1):66–80.
- Karlsson, C. (2016). *Research Methods for Operations Management*. New York: Routledge.
- Karlsson, L., Dahlstedt, AA. G, Regnell, B., Natt och Dag, J., & Persson, A. (2007). Requirements engineering challenges in market-driven software

development-An interview study with practitioners. *Information and Software Technology*, 49(6), 588–604.

Kasunic, M. (2005). *Designing an effective survey*. Pittsburgh: Carnegie Mellon Software Engineering Institute, CMU/SEI-2005-HB-004.

Kavitha, R. and Irfan, R., Ahmed. (2011). A Knowledge Management Framework for Agile Software Development Teams, in *2011 International Conference on Process Automation, Control and Computing*. Coimbatore, India, PP 1-5.

Kettunen, O. (2010). *Agile product development and strategic agility in technology firms*. Master thesis, Helsinki University of Technology, Finland.

Kitchenham, B. A. (2004). *Procedures for performing systematic reviews* NICTA Technical Report 0400011T.1 Keele, UK: Keele University.

Kitchenham, B. A., and Pfleeger, S. L. (2008). Personal opinion surveys. In F. Shull, J. Singer & D. I. K. Sjøberg (Eds.), *Guide to Advanced Empirical Software Engineering* (pp. 63-92). London: Springer.

Kitzinger, J. (2005). Focus group research: using group dynamics to explore perceptions, experiences and understandings. In I. Holloway (Ed.), *Qualitative research in health care* (Vol. 56). Maidenhead: Open University Press.

Komar, M., Shukla, M., and Agarwal, S. (2013). A hybrid Approach of Requirement Engineering in Agile Methodology, Machine Intelligence and Research Advancement (ICMIRA), *2013 International Conference on*.

Kontio, J., Lehtola, L., and Bragge, J. (2004). Using the focus group method in software engineering: obtaining practitioner and user experiences. Paper presented at the *International Symposium on Empirical Software Engineering, 2004. ISESE'04*.

- Krueger, R. A., and Casey, M. A. (2014). *Focus groups: A practical guide for applied research*. Thousand Oaks, California: Sage publications.
- Lakulu, M., Abdullah, R., Selamat, M., Ibrahim, H., Mohd Nor, M. (2010). A Framework of Collaborative Knowledge Management System in Open Source Software Development Environment. *Computer and Information Science*. 3(1):81-90.
- Lee, W., Park, S., Lee, K., Lee, C., Lee, B., Jung, W., Kim, T., Kim, H., and Wu, C. (2005). Agile Development of Web Application by Supporting Process Execution and Extended UML Model. *Software Engineering Conference, 2005. APSEC '05. 12th Asia-Pacific*.
- Lee, W., Park, S., Lee, K., Lee, C., Lee, B., Jung, W., Kim, T., Kim, H., and Wu, C. (2013). Adoption of Agile Methodology in Software Development., *Information & Communication Technology Electronics & Microelectronics (MIPRO), 2013 36th International Convention Ye Wang, Liping Zhao, Xinyu Wang , Xiaohu Yang , Sam Supakkul " PLANT: A pattern language for transforming scenarios into requirements models"* *International Journal of Human-Computer Studies*.
- Lichtenstein, S., Nguyen, L., and Hunter, A. (2005). Issues in IT service-oriented requirements engineering. *Australasian Journal of Information Systems*, vol. 13, no. 1, p. 176, 2005.
- Liu, K., Valerdi, R., & Laplante, P. A. (2010). Better requirements decomposition guidelines can improve cost estimation of systems engineering and human systems integration. *8th Annual Conference on Systems Engineering Research*. Hoboken, NJ.
- Liu, L. L. (2006). Software Maintenance and CMMI for Development: A Practitioner's Point of View. *Journal of Software Engineering*, 1(2), 68–77.

- Maranzato, RP., Neubert, M., Herculano, P. (2011). Moving back to scrum and scaling to scrum of scrums in less than one year. In: The ACM international conference companion on Object oriented programming systems languages and applications companion (SPLASH), pp 125–130
- Martakis, A., & Daneva, M. (2013). Handling Requirements Dependencies in Agile Projects: A Focus Group with Agile.
- Martín-de Castro, G. (2015). Knowledge management and innovation in knowledge-based and high-tech industrial markets: The role of openness and absorptive capacity. *Industrial Marketing Management*, 47, 143-146.
- Mikulenias, G., and Kapocius, K. (2011a). An approach for prioritizing agile practices for adaptation. In W. W. Song, S. Xu, C. Wan, Y. Zhong, W. Wojtkowski, G. Wojtkowski & H. Linger (Eds.), *Information Systems Development* (pp. 485-498). New York: Springer.
- Mikulenias, G., and Kapocius, K. (2011b). A Framework for Decomposition and Analysis of Agile Methodologies during their Adaptation. In W. W. Song, S. Xu, C. Wan, Y. Zhong, W. Wojtkowski, G. Wojtkowski & H. Linger (Eds.), *Information Systems Development* (pp. 547-560). New York: Springer.
- Morisio, M., Seaman, C. B., Basili, V. R., Parra, A. T., Kraft, S. E., and Condon S. E. (2004). COTS-based software development: Processes and open issues. *The Journal of Systems & Software*. 2004. vol. 61, no. 3, pp. 189-199, 2002. S. Crew, *Service Centric System Engineering—EU/IST Integrated Project*.
- Nerur, R., M., and Mangalaraj. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM* 48 (5) (2005) 73–78.
- Neves, FT., Correia, AMR., Rosa, VN., Neto, MC. (2011). Knowledge creation and sharing in software development teams using agile methodologies:

key insights affecting their adoption. In: 6th Conferência Ibérica de Sistemas e Tecnologias de Informação.

Nonaka, I., von Krogh, G. (2009). Tacit knowledge and knowledge conversion: controversy and advancement in organizational knowledge creation theory. *Organ Sci* 20(3):635–652
Pais, S., Talbot, A. & Connor, A.M. (2009) "Bridging the research-practice gap in requirements engineering", *Bulletin of Applied Computing and Information Technology*, 7.

Ouriques, R., Wnuk, K., Svensson, R. B., & Gorschek, T. (2018, November). Thinking Strategically About Knowledge Management in Agile Software Development. In *International Conference on Product-Focused Software Process Improvement* (pp. 389-395). Springer, Cham.

Pandey, D., Suman, U., & Ramani, A. (2010). An Effective Requirement Engineering Process Model for Software Development and Requirements Management. In *Advances in Recent Technologies in Communication and Computing (ARTCom), 2010 International Conference on* (pp. 287–291).

Pandey, D., Suman, U., & Ramani, A. K. (2010). Performance Measurement of Different Requirements Engineering Process Models: A Case Study. *International Journal of Computer Engineering & Technology (IJCET)*, 1(2), 1–15.

Pandey, U. S., & Ramani, A. K. (2009). Social-Organizational Participation difficulties in Requirement Engineering Process-A Study. In *National Conference on Emerging Trends in Software Engineering and Information Technology*, Gwalior Engineering College, Gwalior.

Panian, Z. (2009). User Requirements Engineering and Management in Software Development. In *Proceedings of the European Computing Conference* (pp. 609–620).

- Petersen, K., Wohlin, C. (2009). A Comparison of Issues and Advantages in Agile and Incremental development between State of the Art and an Industrial Case. *Journal of Systems and Software* 82, 1479–1490.
- Pettersen, R., Michael, M., Monfils, FF., Dingsøy, T., Saadaoui, S., Bjørnson, FO., Neophytou, K., Hadjioannou, A. (2012). Practical knowledge management - Techniques for small and medium sized companies, 1st edn. EXTRA Consortium Polanyi, M. (1997). "The Tacit Dimension," in *Knowledge in Organizations*, L. Prusak, Ed. Boston, MA: Butterworth-Heinemann, 1997, pp. 135-146.
- Pinto, D., Bortolozzi, F., Sartori, R., & Tenório, N. (2017). Investigating Knowledge Management within Software Industry: A Systematic Literature Review. *International Journal of Development Research*, 7(12), 17672-17679.
- Racheva, Z., Daneva, M., Herrmann, A. (2010). A Conceptual Model of Client-driven Agile Requirements Prioritization: Results of a Case Study. In: *IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, Bolzano, Italy.
- Racheva, Z., Daneva, M., Sikkel, K. (2009). Value creation by agile projects: Methodology or mystery? In: Bomarius, F., Oivo, M., Jaring, P., Abrahamsson, P. (eds.) *PROFES 2009*. LNBIP, vol. 32, pp. 141–155. Springer, Heidelberg.
- Racheva, Z., Daneva, M., Sikkel, K., Herrmann, A., Wieringa, R. (2010). Do we Know Enough about Requirements Prioritization in Agile Projects: Insights from a Case Study. In: *The Proceedings of Requirements Engineering 2010*, Australia (2010).
- Ramesh, B., Baskerville, R., & Cao, L. (2010). Agile requirements engineering practices and challenges: An empirical study. *Information Systems Journal*, 20(5), 449–480.

- Ruy, F. B., de Almeida Falbo, R., Barcellos, M. P., Costa, S. D., & Guizzardi, G. (2016, November). SEON: A software engineering ontology network. In *European Knowledge Acquisition Workshop* (pp. 527-542). Springer, Cham.
- Ryan, S. and O'Connor, R. (2013). *Acquiring and Sharing Tacit Knowledge in Software Development Teams An Empirical Study*. *Information and Software Technology*. Vol. 55, No. 9, pp. 1614-1624, 2013.
- Rehman, T., Khan, M. N. A., & Riaz, N. (2013). *Analysis of Requirement Engineering Processes, Tools/Techniques and Methodologies*. *International Journal of Information Technology and Computer Science (IJITCS)*, 5(3), 40.
- Rizvi, B. (2013). *A systematic review of distributed agile software engineering*. Alberta: Athabasca University.
- Roger, A. E., Marcel, F. N., & Lopez, A. C. (2010). *Business Process Requirement Engineering*. *International Journal on Computer Science and Engineering*, 2(9).
- Sandra L., Buitrun Francisco J., Pino, Brenda L., Flores-Rios, Jorge E., Ibarra-Esquer, Astorga-Vargas. (2017). "A Model for Enhancing Tacit Knowledge Flow in Non-functional Requirements Elicitation", in *2017 5th International Conference in Software Engineering Research and Innovation (CONISOFT)*. Mérida, Mexico.
- Saini, M., Arif, M., Kulonda, D. (2018). *Critical factors for transferring and sharing tacit knowledge within lean and agile construction processes*. *Construction Innovation*, Vol. 18 Issue: 1, pp.64-89, <https://doi.org/10.1108/CI-06-2016-0036>.
- Santoro, G., Vrontis, D., Thrassou, A., & Dezi, L. (2018). *The Internet of Things: Building a knowledge management system for open innovation and knowledge management capacity*. *Technological Forecasting and Social Change*, 136, 347-354.

- Santos, V., Goldman, A., Santos, C. (2012). Uncovering steady advances for an extreme programming course. *CLEI Electron J* 15(1):1–20.
- Shim, W. & Lee, S. (2017). An Agile Approach for Managing Requirements to Improve Learning and Adaptability. in 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW). Lisbon, Portuga, PP 435-438.
- Stettina, CJ., Heijstek, W., Fægri, TE. (2012). Documentation work in agile teams: the role of documentation formalism in achieving a sustainable practice. In: Agile Conference (AGILE), pp 31–40.
- Talbot, A. & Connor, A.M. (2011). Requirements engineering current practice and capability in small and medium software development enterprises in New Zealand. *Proceedings of the 9th ACIS Conference on Software Engineering Research, Management & Applications*. 2011.
- Topouzidou, S. (2012). SODIUM, Service-Oriented Development In a Unified framework. Final report ISTFP6-004559. <http://www.atc.gr/sodium>.
- Tran, M. (2012). Strengths and weaknesses of moving to a more agile approach: CIO's perspective.
- V, L., Donn, J. (2009). Writing Software Requirements Specifications. Website:<http://www.techwrl.com/techwhirl/magazine/writing/softwarerequirementspecs.html> Access date: January 2009.
- Vinay, S., Aithal, S., and Sudhakara, G. (2013). A quantitative approach using goal-oriented requirements engineering methodology and analytic hierarchy process in selecting the best alternative. In A. K. M., S. R. & T. V. S. Kumar (Eds.), *Proceedings of International Conference on Advances in Computing*. India: Springer.
- von Meyer-Höfer, M., Nitzko, S., and Spiller, A. (2015). Is there an expectation gap? Consumers' expectations towards organic: An exploratory survey in mature and emerging European organic food markets. *British Food Journal*, 117(5), 1527-1546.

- Wang, S., Noe, R.A. (2010). Knowledge sharing: a review and directions for future research. *Human Resour Manage Rev* 20:115–131.
- Wang, X., Maurer, F., Morgan, R. (2010). Tools for supporting distributed agile project planning. *Agility Across Time and Space*, pp 183–200.
- Warburton, R. D. (2011). A time-dependent earned value model for software projects. *International Journal of Project Management*, 29(8), 1082-1090.
- Whitworth, E., Biddle, R. (2007). The social nature of agile teams. *Agile Conference (AGILE)*, Washington, DC, pp 26–36.
- Wirsing, M, & Hölzl, M. (2010). *Rigorous Software Engineering for Service-Oriented Systems—Results of the Sensoria project on Software Engineering for Service-Oriented Computing*. Springer.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Heidelberg: Springer Science & Business Media.
- Yong, A. G., and Pearce, S. (2013). A beginner's guide to factor analysis: Focusing on exploratory factor analysis. *Tutorials in Quantitative Methods for Psychology*, 9(2), 79-94.
- Zhang, Z., and Awasthi, A. (2014). Modelling customer and technical requirements for sustainable supply chain planning. *International Journal of Production Research*, 52(17), 51315154.

APPENDIX A
SURVEY VALEDATION CONTENT FORM



CONTENT VALIDITY SURVEY FORM

Research Title : Injecting Implicit Thinking in Agile Requirements Documentation

Author : Kaiss Ali Elghariani

Supervisors : Dr. Mohd Nazri Kama and Dr. Nurulhuda Firdaus Mohd Azmi

Introduction

The research is about to inject implicit thinking of software engineers as part of agile requirements documentation. Implementing the injection of the implicit thinking shall be carried out in two real software projects consists of few practices, which will lead to explore the challenges of agile requirements engineering documentation.

Procedures to be followed

We have classified the research questions and grouped into 4 sections, which should be answered by the participants of the software projects. The questions are divided into four (4) sections based on research requirements, which are Demographic Profile of Respondents, Agile methodology, Agile Requirements Engineering and Framework effectiveness evaluation. The experts are requested to rate the relevancy of each group of questions by rating from 1-5. The experts can also add any comments or suggestions can be added to the group of questions.

Relevancy Rates:

Strongly 1	Not Relevant 2	Relevant (but not important) 3	Relevant 4	Strongly Relevant 5
---------------	-------------------	-----------------------------------	---------------	------------------------

Thank You

Section		Rate Scale					Comment
		1	2	3	4	5	
A. Demographic Profile of Respondents							
A.1 Respondent Profile							
Q1	What is your age?						
Q2	What is the highest degree or level of school you have completed?						
A.2 Software Project Profile							
Q3	Which of the following best describes your position while engaged in this project?						
Q4	How many years have you worked with an outsourcing company?						
Q5	Which of the following categories best describes the type of software developed for this project?						
Q6	Which of the following application domains does/did the project apply to?						
Q7	What is/was the duration of the project (from inception to delivery)?						
Q8	How would you estimate the size of the project in terms of lines of code?						
B. Agile Methodology							
B.1 Not applying Agile Methodology							
Q9	Which of the following software development methodology best describes the one you are using/did use in the project?						
B.2 Applying Agile Methodology							
Q10	If you are using agile methodology, which of the following agile approaches describes the one you are using/did use in the project?						
C. Agile Requirements Engineering							

Section		Rate Scale					Comment
Questions		1	2	3	4	5	
Q11	The following statements are the challenges of requirements engineering in agile methodology. (Please rate these statements by clicking one box with the following scales)?						
Q12	Are the requirements engineering practices affects analysis and design phases in agile software methodology? (Please rate these statements by clicking one box with the following scales)?						
Q13	Do the user stories and task cards have provided the basic practice of requirement engineering?						
Q14	Is Implicit thinking of software engineers included in agile requirements engineering?						
Q15	Does the Implicit thinking of software engineer's affects software maintenance phase?						

APPENDIX B

SUMMARY OF SURVEY VALEDATION CONTENT FORM

Expert's Name	University/ORG	Overall Evaluation			Comments
		Not Relevant	Relevant	Strongly Relevant	
Faizura Haneem	UTM		√		Last section's questions are mostly depends on Q14
Dr. Mazidah	UUM		√		Outsourcing company shouldn't be specified
Dr. Mazni	UUM		√		Outsourcing company shouldn't be specified Sections B1&2 are confused, so better simplify them
Dr. Abdulmajid	IIUM		√		Project scale not necessary
Dr. Akram	IIUM		√		Questions in section B need further explanation
Dr. Elammari	Apple Company		√		Question shouldn't be stated Need further explanation to the participant about implicit thinking of software developers

APPENDIX C

GUIDELINE OF INJECTING THE IMPLICIT THINKING KNOWLEDGE IN AGILE REQUIREMENTS DOCUMENTATION FRAMEWORK



*User Guideline for Implicit Thinking Knowledge
Injection for Software System Requirements
Documentation in Agile Methodology (IITKARD)*

TABLE OF CONTENTS

1	INTRODUCTION	204
	1.1 Intended Readership	204
	1.2 Purpose	204
	1.3 Conventions	205
2	OVERVIEW	205
3	INSTRUCTIONS	205
	3.1 Signing in	205
	3.2 System Menu and Dashboard	206
	3.3. Add User Story	206
	3.4 Add First Argument	208
	4.5 Add Team Members Arguments	209
	4.6 Search for documented user story/Argument.	210
	4.7 Exit System	211
4	DOCUMENT CONTROL	212
5	DOCUMENT SIGNOFF	212

1 Introduction

1.1 Intended Readership

The users of the tool that supports IITKARD framework are classified as follow.

- **Team Leader (Admin)**
 - *Expert, accessing the tool for a significant to manage software project requirements (User Stories)*
 - *Team member, create and interact with the requirements arguments.*
- **Team members, including**
 - *Experts, accessing the tool to interact with the arguments.*
- **The level of experience of agile methodology needed is minimum 1 year.**
- **Please follow section 3 for using the tool.**

Then language used for the tool is understandable for non-speaker English.

1.2 Purpose

The purpose of IITKARD framework tool is to inject implicit thinking knowledge of software engineers in agile requirements engineering, and this user guide describing how to use the tool.

The processes supported by the tool of IITKARD framework as follow:

- 1- Create User Story
- 2- Set Admin First Argument
- 3- Inject implicit hiking knowledge of software engineers
- 4- Document implicit thinking knowledge

1.3 Conventions

As is the purpose of this tool is to support the implantation of IITKARD framework, the implicit thinking knowledge has been categorized into 4 types as follow:

- 1- Issue
- 2- Assumption
- 3- Opinion
- 4- Question

2 Overview

IITKARD is an extension of JIRA software framework of issue and project tracking for agile team provided by Atlassian Company, which is an enterprise software company that builds software products for software engineers, project management. IITKARD includes the existing practices in agile requirements engineering and the novelty feature of adding the software engineers' implicit thinking in agile software development methodology.

3 Instructions

3.1 Signing in

A user ID and password is required to log onto web interface.


- 1- Key in username
- 2- Key in password
- 3- Choose software project (dropdown list)
- 4- Click sign button

Username or Email

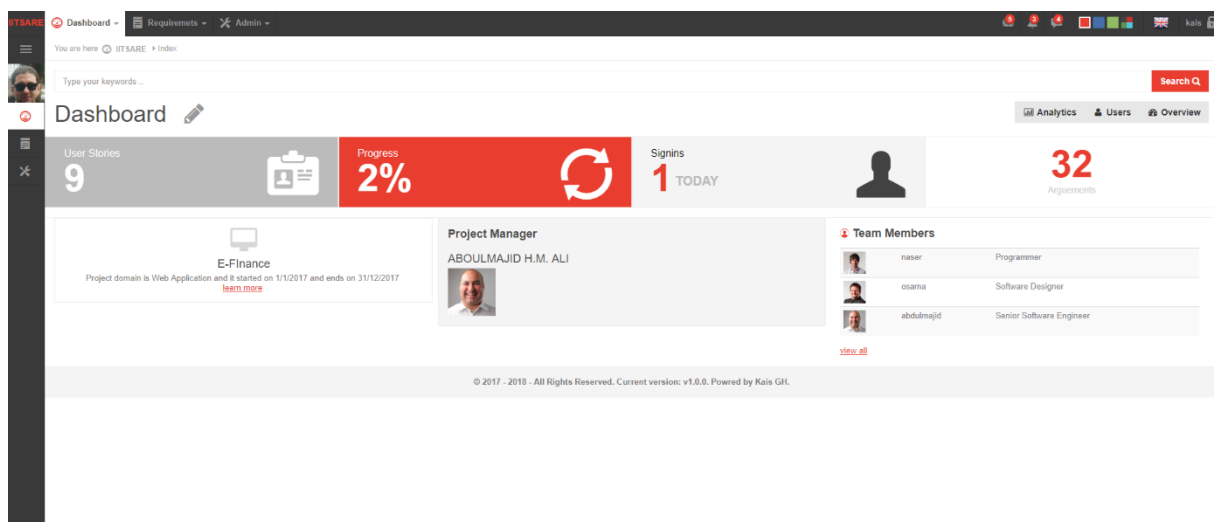
Password

Project

Remember me

 Please enter your username and password ...

3.2 System Menu and Dashboard



3.3. Add User Story

This function assists user to create a user story as a software requirement, the team leader should fill up the form based on user story attributes.

- 1- Key in user story title
- 2- Choose task engineer (Developer) (dropdown list)
- 3- Choose Requirement Priority
- 4- Click Next
- 5- Key in user story description

The screenshot shows a web application interface for maintaining user stories. The main heading is 'Maintain User story'. On the left, there is a sidebar with navigation options: Dashboard, Requirements, and Admin. A red notification box in the sidebar states: 'This is a prototype tool of ITSARE Framework, for PhD research'. The main content area displays a form with four steps:

- Step 1: Story Info
- Step 2: Story details (Active)
- Step 3: Admin Comment
- Step 4: Confirmation

The 'Story details' step includes the following fields:

- Title:** Enter Story title ...
- Developer:** KAIS ELGHARIANI
- Priority:** High

Navigation buttons at the bottom of the form include 'First', 'Last', 'Previous', and 'Next'.

For example: a user story called user login. This is considered as a requirement, so following is a user story details:

Title: Delete Student Record.

Developer: TAUFEEQ

Priority: High.

Description: As a user, I want to delete student record any time I need.

3.4 Add First Argument

Before finishing adding the user story admin shall

- 1- Choose one of the argument types such as issues, assumption, suggestion, question and opinion.
- 2- Choose effort (High, Normal or low)
- 3- Choose one of the argument type (Issue, Assumption, Suggestion, Question)
- 4- Insert argument's text.
- 5- Click next
- 6- Click confirmed

The screenshot shows a web application interface for 'Maintain User story'. The interface is divided into four steps:

- Step 1: Story info** - Includes radio buttons for Effort: High, Normal (selected), Low.
- Step 2: Story details** - Includes radio buttons for Argument: Issue (selected), Assumption, Suggestion, Question.
- Step 3: Admin Comment** - A large text area for entering an admin comment.
- Step 4: Confirmation** - A section for confirming the entry.

Navigation buttons 'First', 'Last', 'Previous', and 'Next' are located at the bottom of the form.

Based on the example given in the previous section (Delete Student Record), the admin or the team member shall set the following info:

Effort: The expected effort is normal.

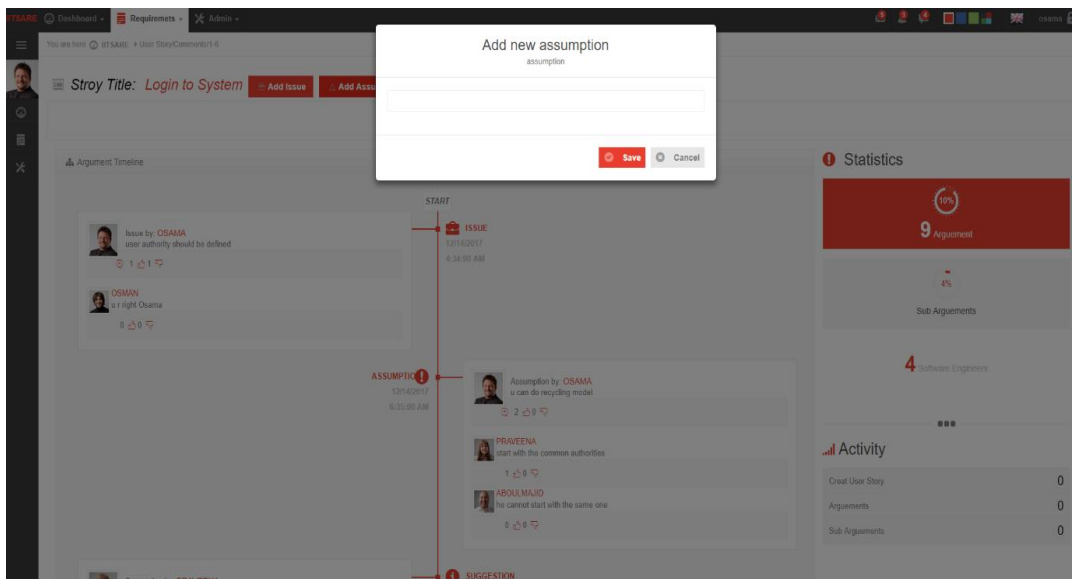
Argument: in this example the team leader will ask a question “Student record should be physically deleted or we better set a flag as a deactivated student?”

3.5 Add Team Members Arguments

Team members can interact with arguments such as Issues, Assumptions, Suggestions, Questions and Opinions related to user story.

- 1- click on the icon shown on the user story list
- 2- choose one of the arguments type and then pop-up windows appeared
- 3- Key in the argument text.

The following figure shows the form of adding argument.



Following the previous example (Delete student record), one of the team members given a suggestion: “better you set a flag to show student status is active or inactive!”. Another team member gave an

assumption: “if you physically delete student record, student data will be lost”.

3.6 Search for documented user story/Argument.

All user stories and arguments are documented in IITKARD tool.

From the dashboard page user can follow the below step:

- 4- In the search form, key in in keywords related to what are you looking for
- 5- List of user stories contain the specific keywords will be displayed.
- 6- Click on the user story argument
- 7- Arguments will be displayed includes all arguments details.

A list of stories contains any of the key word will be displayed in sort of storyline. Also, to display in stories contains any key word will be displayed It contains the title and its arguments member’s photo, name, text and icon of the argument type whether it is an issue, assumption or suggestion, the following shows implicit thinking knowledge documentation. The following figure shows the list of user stories.

ITKARD Dashboard - Requirements - Admin - You are here ITKARD - User StoryComments/3-1

Story Title: CMS sub-modules

Add Issue Add Assumption Add Suggestion Add Question

Note: This page is optimized for print. Try print the implicit thinking knowledge and check out the preview. For example, this note will not be visible.

5/27/2018 12:17:28 AM Statistics

Print Arguments

Argument Timeline

START

QUESTION

Question by Mohd 25/2018 6:10
All Requirements should be under each modules???

OSAMA lets follow dr azalaha suggestion

SUGGESTION

Suggestion by NUI 25/2018 6:12
better if we classify the requirements based on the modules

ASSUMPTION

Assumption by OSAMA 25/2018 6:14
we can re use shared requirements between the SIGS modules

5 Arguments

6 Software Engineers

Activity

User Story Cards 0

Arguments 0

Sub Arguments 5

ITKARD Dashboard - Requirements - Admin - You are here ITKARD - User StoryList

Maintain User Story

Copy CSV Excel PDF Print Show 10 Search:

	Story No.	Story Title	Story Priority	Story Effort	Story Date	Developer	Action
<input checked="" type="checkbox"/>	1-10	fgpndg	High	Normal	2-5-2018	KAIS	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 2
<input checked="" type="checkbox"/>	1-9	test	Low	Normal	29-3-2018	ABOULMAJID	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 1
<input checked="" type="checkbox"/>	1-5	delete comment	Low	Normal	27-11-2017	OSMAN	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 10
<input checked="" type="checkbox"/>	1-7	Update Student Record	High	on	14-12-2017	OSMAN	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 3
<input checked="" type="checkbox"/>	1-8	Print Report	High	Normal	14-12-2017	TAUFEEQ	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 4
<input checked="" type="checkbox"/>	1-11	cccccc	Low	High	2-5-2018	PRAVEENA	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 1
<input checked="" type="checkbox"/>	#11	Login to System	High	Normal	26-11-2017	PRAVEENA	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 0
<input checked="" type="checkbox"/>	#1-2	Login to System	High	Normal	26-11-2017	PRAVEENA	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 0
<input checked="" type="checkbox"/>	#1-3	Login to System	High	Normal	26-11-2017	PRAVEENA	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 1
<input checked="" type="checkbox"/>	1-4	Add Comment	High	High	26-11-2017	BAIZURA	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> 6

Showing 1 to 10 of 12 entries

-- First -- Previous 1 2 Next -- Last --

3.7 Exit System

Click Logout

#1 Document Control

Title: User Guideline of IITKARD framework
Date: 28/3/2018
Author: Kais Ali Elghariani
Distribution: Project Sponsor
Project Team

DOCUMENT SIGN OFF

Nature of Signoff	Person	Signature	Date	Role
Author	Kaiss Elghariani		24th March 2018	PhD Candidate
Reviewer	Prof Madya. Dr. Nazri Kama		28th March 2018	Reviewer
Reviewer	Dr. NurZalaiah Binti AbuBaker		28th March 2018	Reviewer
Reviewer	Dr. NurulHuda Ferdaous			Reviewer

APPENDIX D

EXPERIMENTATION BRIEFING



Experiment of

Injecting Implicit Thinking Knowledge in Agile Requirements Documentation (IITKARD)

Kaiss Elghariani

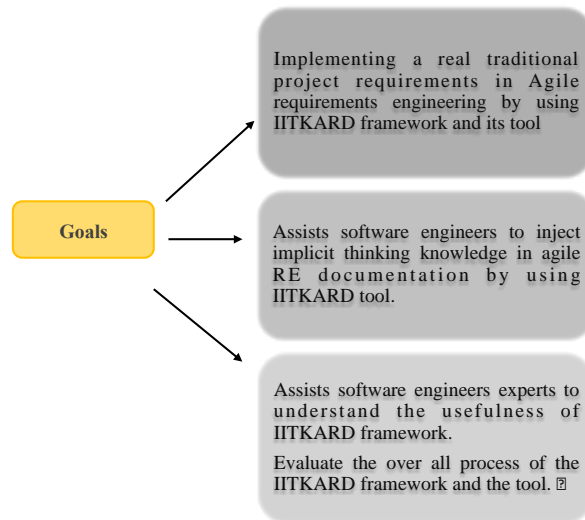


inovatif • entrepreneurial • global

Outlines

- Experiment Goals
- Agile Methodology
- Agile RE
- Implicit Thinking Knowledge
- Experiment Process
- Experiment Evaluation

Experiment Goals



Agile Methodology

Main Principles of Agile :

- Individuals and interactions over software development processes
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile RE

- Agile RE documented as a user story card:

What is a user story?

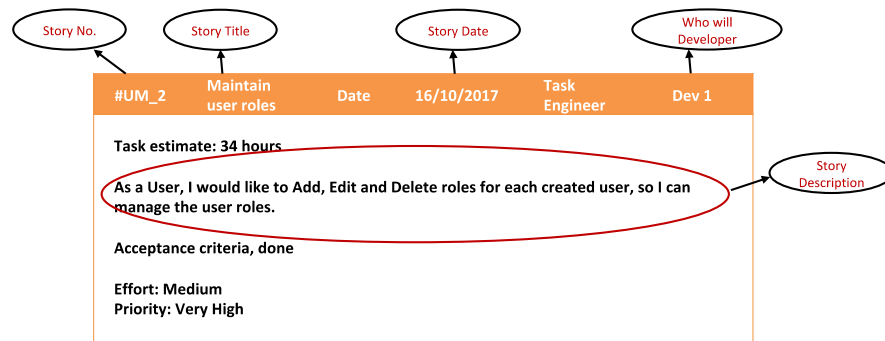
- User stories represent customer requirements in a card, leading to conversation and confirmation (Jeffries, 2001)
- User stories only capture the essential elements of a requirement:
 - *who* it is for
 - *what* it expects from the system
 - *why* it is important (optional) (Yu and Mylopoulos, 1994)

Simple format:
 As a **role**, I want to **action**, (so that **benefit**) (Mike Cohn)



Agile RE

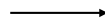
- Sample of User Story Card:



Implicit Thinking Knowledge

Implicit Thinking Knowledge in SE helps to:

- Implicit Thinking Knowledge



- Understanding how software is developed
- Implicit thinking knowledge tends to transfer knowledge directly from one person to another
- Implicit thinking knowledge returns great investment and it increases workplace efficiency



- It provides the ability of software developer's community during software maintenance phase.
- Support the cause of learning software organization



www.utm.my

inovatif • entrepreneurial • global

Implicit Thinking Knowledge

Success Factors for Implicit Thinking Knowledge:

- **Personal Interaction:** person has to interact with team members for knowledge sharing.
- **Good Management Leader:** as a role model for organization to follow by members.
- **Human Encouragement:** by the people in an organization. Motivation rewards and hopes.
- **Relation** should be good between **sender and receiver** when knowledge is in sharing. Such as trust and openness.
-

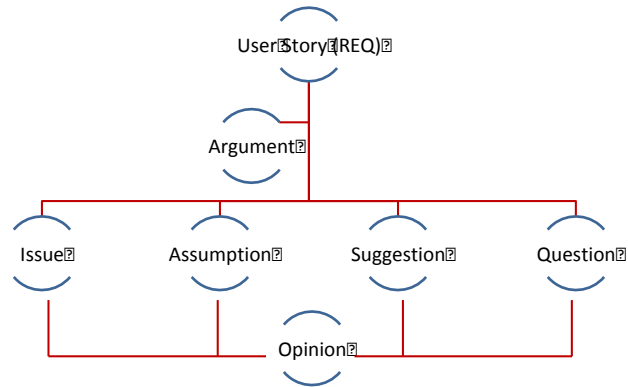


www.utm.my

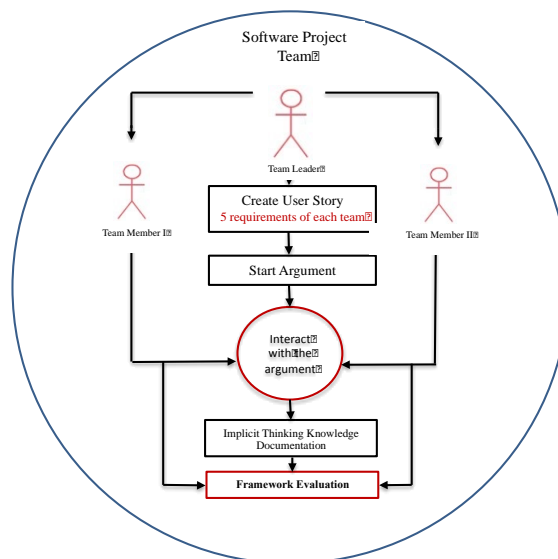
inovatif • entrepreneurial • global

Implicit Thinking Knowledge

Implicit Thinking knowledge categorizations:



Experiment Process



SGIS Requirements

Teams	Requirements Module	Requirements' Module & Titles
Group 1 3 Members	User Administration	<ul style="list-style-type: none"> • Maintain user roles • Managing main administrator roles • Managing primary administrator roles • Managing user roles • Send Email
Group 2 3 Members	Knowledge Management	<ul style="list-style-type: none"> • Acquire data from agencies • Types of agencies' data • Obtaining Method • Display latest content • Add Agency's page tabs
Group 3 3 Members	Religious Questions and Answers	<ul style="list-style-type: none"> • Religious Questions and Answer sub module • Religious Questions and Answer bulk search • Add Questions and Answers • Delete Questions and Answers • Publish / un publish questions & Answers
Group 4 3 Members	Portal Life Event management	<ul style="list-style-type: none"> • Control visual elements • Include upload and settings • Show current slider • delete slider • Upload slider image
12 Members	4 Modules	20 Requirements



Evaluation Instrument

- Questionnaire will be distributed for each team member for evaluation purpose



APPENDIX E
SGIS DATA USED FOR IMPLIMINTING THE
EXPERIMENT EVALUATION

GROUP 1

Module: User Administration

#UM_2	Maintain user roles	Date	16/10/2017	Task Engineer	Developer 1
<p>Task estimate: 34 hours</p> <p>As a User, I would like to Add, Edit and Delete roles for each created user, so I can manage the user roles.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#UM_3	Managing main administrator roles	Date	16/10/2017	Task Engineer	Developer 1
<p>Task estimate: 34 hours</p> <p>As a user, I would like the main administrator to interpret have full access to the CMS module and its sub modules as follow:</p> <ul style="list-style-type: none"> • User Management • User Administration • Role Configuration • Knowledge Management • Post Management • Management of Forms and Links • File Management • Content Management of Religious Questions • Portal Life Event Management • Reports • Settings & Configuration • Audit Trail Activity <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#UM_4	Managing user roles	Date	16/10/2017	Task Engineer	Developer 1
<p>Task estimate: 34 hours</p> <p>As a User, I would like the user has a specific role in a system, so I only be able to see the parts marked as 'viewable'.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#UM_5	Managing primary administrator roles	Date	16/10/2017	Task Engineer	Developer 1
<p>Task estimate: 34 hours</p> <p>As a User, I would like, the primary administrator can do the actions below:</p> <ul style="list-style-type: none"> 8- Add Administrator User (webmasters) 9- Delete User 10- Assign / Convert user role <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#UM_7	Send Email	Date	16/10/2017	Task Engineer	Developer 1
<p>Task estimate: 34 hours</p> <p>As a User, I would like, users created in the system receive an email that aims to tell them how to create their own username and change the user's password (if necessary). The user can then log in to CMS by using the username and password that was previously created.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

GROUP 2

Module Title: Knowledge Management

#KM_1	Acquire data from agencies	Date	16/10/2017	Task Engineer	Developer 2
<p>Task estimate: 1 week</p> <p>As a User, I would like this module to acquire and collect various types of data from different agency portals, all into CMS. Data collected in the Knowledge Management module will be published to the Life Portal.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#KM_2	Types of agencies' data	Date	16/10/2017	Task Engineer	Developer 2
<p>Task estimate: 34 hours</p> <p>As a user, I would like data from other agencies include information content from their website, forms and downloads and direct links to the agency's website.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#KM_3	Obtaining Method	Date	16/10/2017	Task Engineer	Developer 2
<p>Task estimate: 1 week</p> <p>As a User, I would like the data from different agencies can be obtained and updated through the following methods:</p> <p>11- Content manager manually entering content other users include content that has been provided by agencies</p> <ul style="list-style-type: none"> • Content that has been found in the template file (excel) will be uploaded, which files have been filled out by webmaster SGIS or content manager from agencies. • Extracting content from the website specific agencies and content submissions to SGIS CMS • Extract data for agencies' website via feeds, using JSON endpoints <p>Through the method of uploading content, the excel template file will be uploaded, and once processed, the contents will be available in relevant sections / sections</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#KM_4	Display latest content	Date	5/6/2013	Task Engineer	Developer 2
<p>Task estimate: 48 hours</p> <p>As a User, I would like, the list of agencies that are now integrated with the latest content, will appear. Administrators can click on any agency to go to the agency's website within the CMS link.</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Very High</p>					

#KM_5	Add Agency's page tabs	Date	16/10/2017	Task Engineer	Developer 2
<p>Task estimate: 34 hours</p> <p>As a User, I would like, agency page contains 3 tabs, namely: 1) Content / Information 2) Download 3) Link</p> <p>Acceptance criteria, done</p> <p>Effort: Medium Priority: Normal</p>					

GROUP 3

Module Title: Religious Questions and Answers

#SJ_01	Religious Questions and Answer sub module	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to have sub module under Knowledge Management module called the Religious Questions and Answer. This module enables relevant users to add and organize content in the Religious Questions and Answer section of Portal Life Event.</p> <p>Effort: Medium Priority: Very High</p>					

# SJ_04	Religious Questions and Answer bulk search	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to have more than one choice, so I can do bulk research on Religious Questions and Answers</p> <p>Effort: Medium Priority: Very High</p>					

# SJ_05	Add Questions and Answers	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to add a Q & A entry, so the user will be taken to the form where they will be asked to provide the following information:</p> <ul style="list-style-type: none"> ○ Questions ○ Answers ○ Categories, Tags & Keywords <p>When clicking on Submit, this new Question and Answer entry will appear in the full list of entries in the Religious Questions page</p> <p>Effort: Medium Priority: Very High</p>					

# SJ_06	Delete Questions and Answers	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to post entries, and also delete them</p> <p>Effort: Medium Priority: Very High</p>					

# SJ_07	Publish / un publish questions & Answers	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to The Religious Questions and Answer entry will have publish/un publish options. This will control the appearance of question-answer entries on the portal.</p> <p>Effort: Medium Priority: Very High</p>					

GROUP 4

Module Title: Portal Life Event management

#LP_01	Control visual elements	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like, administrators be able to control some visual elements of the Life Event portal that will be visible to the public. Visual element settings depend on their presence on the Portal (For example, if a theme is selected without a "slider", the settings are based on "Slider" will not be here)</p> <p>Effort: Medium Priority: Very High</p>					

#LP_02	Include upload and settings	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like this module includes sections for uploading and setting banners / images "sliders" for portals. Upon entering the module, the administrator will be able to upload / remove the "slider" image.</p> <p>Effort: Medium Priority: Very High</p>					

#LP_03	Show current slider	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like the page will display a table showing the current "slider" image that has been uploaded. On the side of the image there will be a button to delete, un publish.</p> <p>Effort: Medium Priority: Very High</p>					

#LP_04	delete slider	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like click delete button, the "slider" image will be deleted from the system. When clicking on un publish, users can control whether the "slider" image is visible on the Life Event portal or not.</p> <p>Effort: Medium Priority: Very High</p>					

#LP_05	Upload slider image	Date	16/10/2017	Task Engineer	Developer 3
<p>Task estimate: 34 hours</p> <p>As a User, I would like to have upload options. Here, users will be able to click the upload button, and add a "slider" image from their computer. When images are uploaded, the system will present the function of the size in which the user should measure the image to a certain dimension suitable for "slider".</p> <p>Effort: Medium Priority: Very High</p>					

APPENDIX F

THE OUTCOME OF IITKARD FRAMEWORK

The dashboard provides a comprehensive overview of the project. It features a navigation bar with 'Dashboard', 'Requirements', and 'Admin' options. A search bar is available for finding specific keywords. The main content area includes several key metrics: 15 User Stories, 2% Progress (indicated by a red progress bar and a refresh icon), 1 Signin today, and 49 Arguments. A section for the 'Selnagor Islamic Getway System' project is highlighted, noting its domain as Web & Mobile Application and its timeline from 16/10/2017 to 15/10/2018. The Project Manager, Mohd Nazri Bin Kama, is listed, along with a list of Team Members: farhan (Requirements Engineer), haneem (Software Requirements analysis), and jalal (Software Requirements analysis). A footer note states: 'Reserved© 2017 - 2018 - All Rights. Current version: v1.0.0. Powred by Kais GH. UTM'.

The table displays a list of user stories for the project. Each row includes a checkbox, story number, title, priority, effort, date, developer, and an action column with a flag icon and a count. The stories are as follows:

	Story No.	Story Title	Story Priority	Story Effort	Story Date	Developer	Action
<input checked="" type="checkbox"/>	3-10	Add Questions and Answers	Normal	Normal	27-5-2018	Farhan	🚩 - 4
<input checked="" type="checkbox"/>	3-4	Publish / Unpublished Questions & Answer	Normal	Normal	27-5-2018	Farhan	🚩 - 5
<input checked="" type="checkbox"/>	3-7	Delete Questions and Answers	Normal	Normal	27-5-2018	Farhan	🚩 - 3
<input checked="" type="checkbox"/>	3-18	Obtaining Method	High	Normal	27-5-2018	Amelia	🚩 - 3
<input checked="" type="checkbox"/>	3-5	Religious Questions and Answer	High	Normal	27-5-2018	Amelia	🚩 - 3
<input checked="" type="checkbox"/>	3-8	Religious Questions and Answer bulk search	High	Normal	27-5-2018	Amelia	🚩 - 2
<input checked="" type="checkbox"/>	3-11	Display latest content	High	Normal	27-5-2018	hazila	🚩 - 2
<input checked="" type="checkbox"/>	3-13	Add Agency's page tabs	Normal	Normal	27-5-2018	hazila	🚩 - 3
<input checked="" type="checkbox"/>	3-15	Acquire data from agencies	Normal	Normal	27-5-2018	hazila	🚩 - 4
<input checked="" type="checkbox"/>	3-9	Types of Agencies Data	High	Normal	27-5-2018	hazila	🚩 - 3

Showing 1 to 10 of 15 entries

Navigation: ← First ← Previous 1 2 Next → Last →

<input checked="" type="checkbox"/>	Story No.	Arguement User	Arguement Type	Arguement	Arguement Date	Action
<input checked="" type="checkbox"/>	3-4	farhan	Question	How many authorise user will be required?	27-5-2018	
<input checked="" type="checkbox"/>	3-9	haneem	Suggestion	Hyperlink to the original sources (agency's website)	28-5-2018	
<input checked="" type="checkbox"/>	3-11	haneem	Assumption	When updating the content, user can choose agencies from the list to be populated at the content.	28-5-2018	
<input checked="" type="checkbox"/>	3-13	haneem	Suggestion	The content of agency page categorized accordingly for easy retrieval.	28-5-2018	
<input checked="" type="checkbox"/>	3-15	haneem	Suggestion	The data includes agency's vision, mission and agency's services.	28-5-2018	
<input checked="" type="checkbox"/>	3-9	haneem	Issue	There is an issue if the link is broken. Can u add verification code before display the link. If the link is broken, please add notice to the user.	27-5-2018	
<input checked="" type="checkbox"/>	3-18	haneem	Issue	SGIS CMS used for what	28-5-2018	
<input checked="" type="checkbox"/>	3-13	jalal	Suggestion	user stories should be defined much clearly	27-5-2018	
<input checked="" type="checkbox"/>	3-18	jalal	Suggestion	i think user stories should be written more clearly	27-5-2018	
<input checked="" type="checkbox"/>	3-10	amelia	Suggestion	The Q&A need to be tagged by topic	27-5-2018	

Showing 1 to 10 of 49 entries

[← First](#)
[← Previous](#)
[1](#)
[2](#)
[3](#)
[4](#)
[5](#)
[Next →](#)
[Last →](#)

ITKARD Dashboard
Requirements
Admin

You are here: ITKARD > User Story/Story/3-10

Note:
This page is optimized for print. Try print the invoice and check out the preview. For example, this note will not be visible.

[Print User Card](#)

3-10	Add Questions and Answers	27/5/2018	Task Engineer	Farhan
------	---------------------------	-----------	---------------	--------

Estimated Time: 2 days

As a User, I would like to add a Q & A entry, so the user will be taken to the form where they will asked to provide the following information:

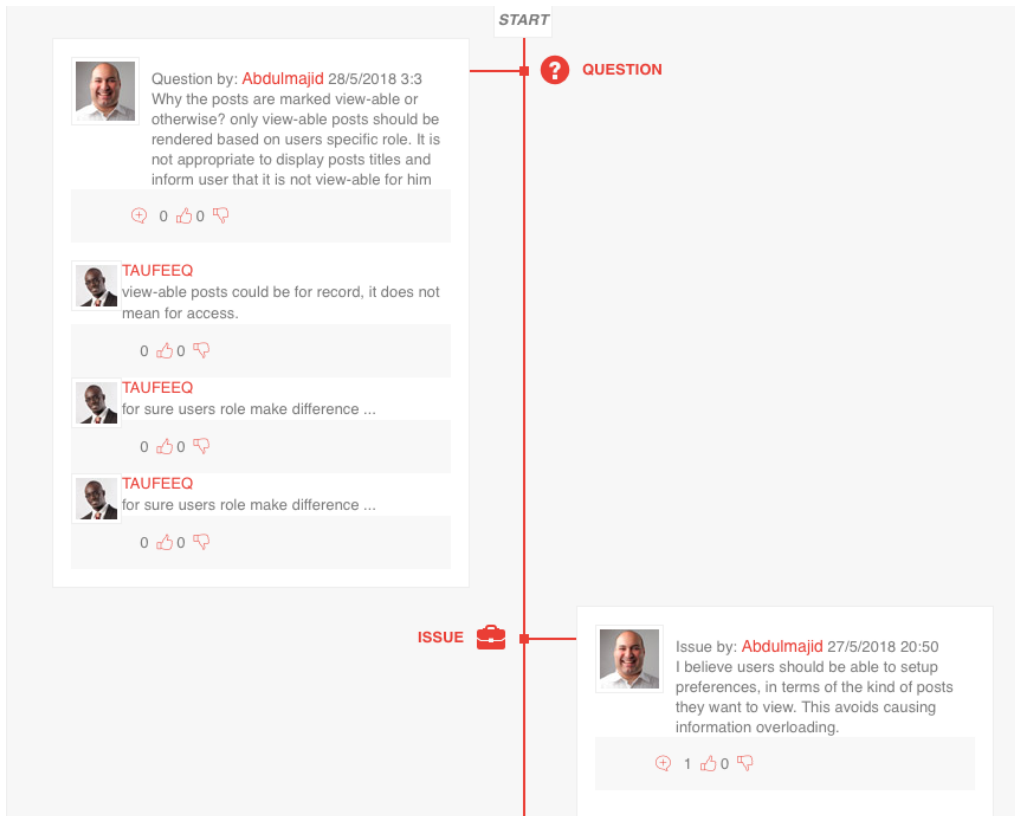
- *Questions
- *Answers
- *Categories, Tags & Keywords

When clicking on Submit, this new Questions and Answer entry will appear in the full list of entries in the Religious Questions page

Story Effort: Normal

Story Priority: Normal Arguments: 4

© 2017 - 2018 - All Rights Reserved. Current version: v1.0.0. Powered by Kais GH.



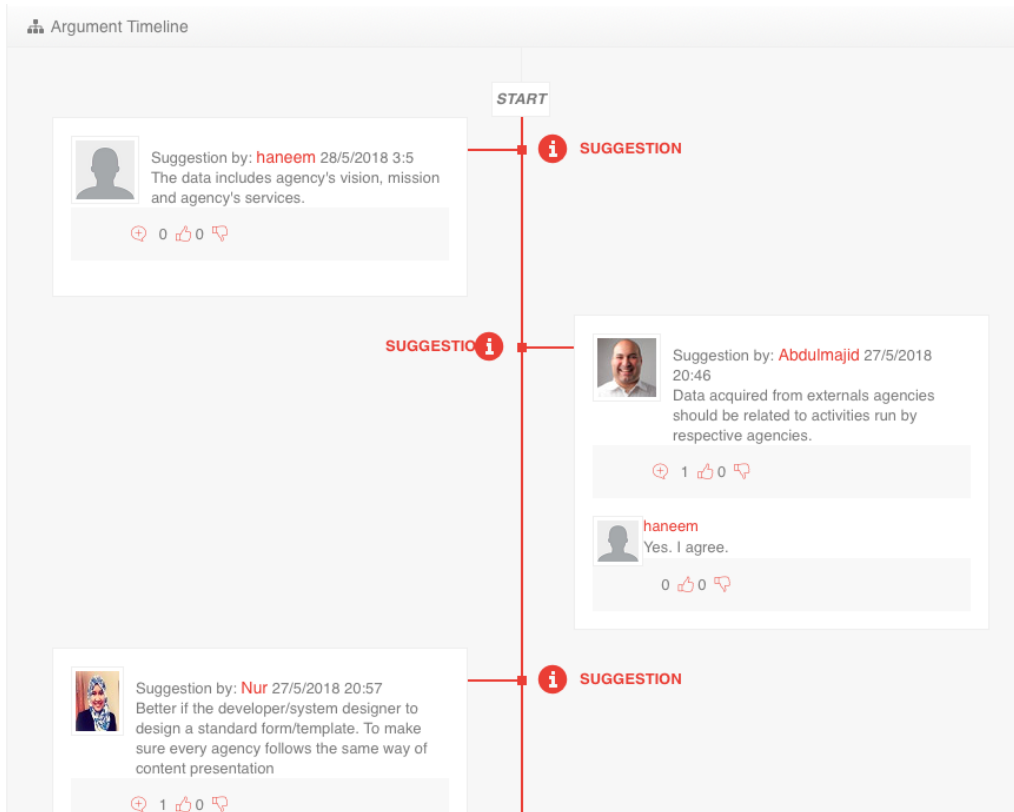
Story

Note:
This page is optimized for print. Try print the invoice and check out the preview. For example, this note will not be visible.

[Print User Card](#)

3-12	Managing main administrator roles	27/5/2018	Task Engineer	Abdulmajid
Estimated Time: 2 days				
I would like the main administrator to interpret have full access to the CMS module and its sub modules as follows:				
<ul style="list-style-type: none"> *User management -user Administration -Role Configuration *Knowledge Management -Post Management -File Management -Content Management and Religious Questions *Portal life Event Management -REports Settings & configuration -Audit Trail Activity 				
Story Effort: Normal				
Story Priority: Normal				Arguments: 2

© 2017 - 2018 - All Rights Reserved. Current version: v1.0.0. Powred by Kais GH.



ITKARD Dashboard Requirements Admin azaliah

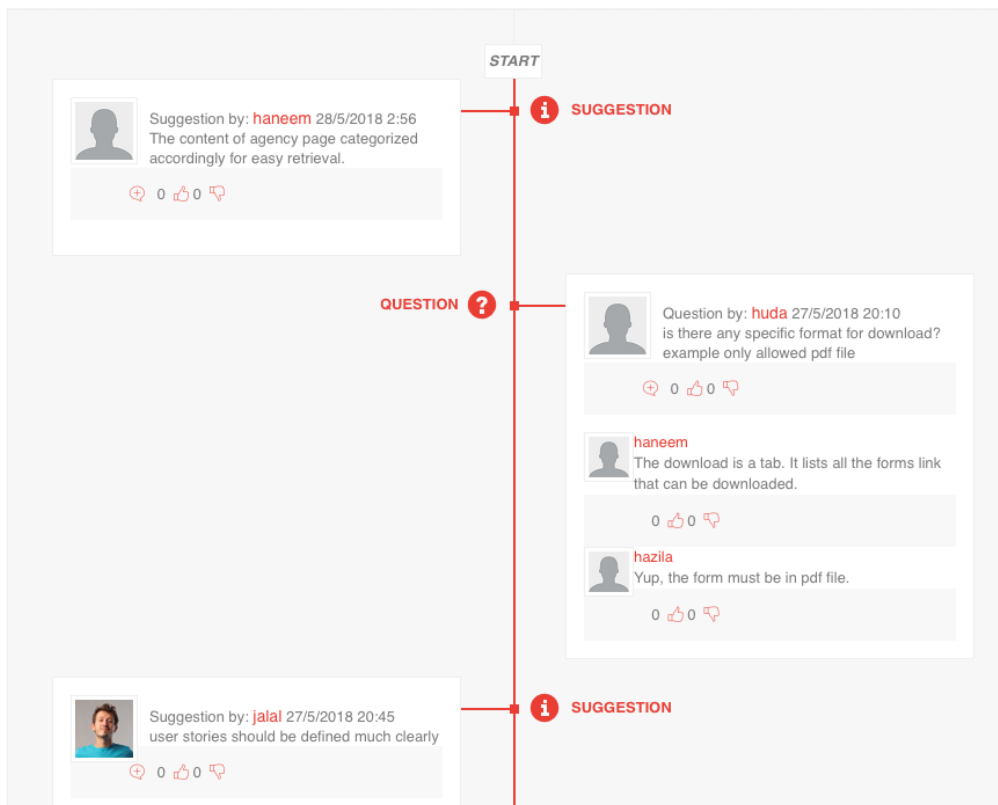
You are here ITKARD > User Story/Story/3-10

Story

Note: This page is optimized for print. Try print the invoice and check out the preview. For example, this note will not be visible. [Print User Card](#)

3-10	Add Questions and Answers	27/5/2018	Task Engineer	Farhan
Estimated Time: 2 days				
As a User, I would like to add a Q & A entry, so the user will be taken to the form where they will asked to provide the following information:				
*Questions				
*Answers				
*Categories, Tags & Keywords				
When clicking on Submit, this new Questions and Answer entry will appear in the full list of entries in the Religious Questions page				
Story Effort: Normal				
Story Priority: Normal				Arguments: 4

© 2017 - 2018 - All Rights Reserved. Current version: v1.0.0. Powred by Kais GH. UTM



APPENDIX G

THE EXPERIMENT INSTRUMENT



Implicit Thinking Knowledge Injection Framework for Software
System Requirements Documentation in Agile Methodology

PhD Candidate: Kaiss Elghariani, kais.gh@hotmail.com, +60178805576

Supervisors:

Assoc. Prof. Dr. Mohd Nazri Kama Main Supervisor mdnazri@utm.my	Dr. Nurulhuda Firdaus Mohd Azmi Co-supervisor I huda@utm.my	Dr. Nur Azaliah Bt Abu Bakar Co-supervisor II azaliah@utm.my
Advanced Informatics School, Level 5, Menara Razak, Jalan Sultan Yahya Petra, 54100 Kuala Lumpur.		

Purpose of the Study

We are conducting a research on integrating implicit thinking knowledge in agile software requirements documentation. Also, we are collecting experts' satisfaction level by using IITKARD framework and prototype tool, which supports the implementation of the framework. The information required is for evaluation purpose.

Procedures to be followed

The questionnaire is divided into 4 sections, 1) Demographic profile of respondents, which divided into 2 subsections 1.1) Respondent Profile and 2.2) Project profile. Followed by section 3) Agile Methodology and Section 4) IITKARD framework and its prototype tool evaluation.

Statement of Confidentiality

Your participation in this research is confidential. In the event of any publication or presentation resulting from the research, no personally identifiable information will be shared because your name is in no way linked to your responses. Your confidentiality will be kept to the degree permitted by the technology used.

A. Demographic Profile of Respondents

A.1 Respondent Profile

1. What is your age?

- Under 22 years old
- 23-30 years old
- 31-40 years old
- 41-50 years old
- 51-60 years old
- 61 years or older

2. What is the highest qualification you have completed?

- Doctorate degree
- Master's degree
- Bachelor's degree
- Diploma and Advanced Diploma
- Academic and Vocational and Technical Certificates
- Skills Certificates

3. What is your experience duration in Software Development Methodology?

- Less than 6 months
- 6 months - 12 months
- 1 year – 2 years
- 2 years - 3 years
- 3 years - 4 years
- 4 years - 5 years
- More than 5 years

4. What is your experience duration in Agile Software Development Methodology?

- Less than 6 months
- 6 months - 12 months
- 1 year – 2 years
- 2 years - 3 years
- 3 years - 4 years
- 4 years - 5 years
- More than 5 years

A.2 Software Project Profile

5. Which of the following scales is the project involved in?

- Small-scale software project
- Medium-scale software project

Large-scale software project

6. Which of the following categories best describes the type of software developed for this project?

Desktop

Web-based (not Web services)

2-Tier client/server

Database

Mobile Application

Other

If other please specify

7. Which of the following requirements engineering practices you involved in while engaged in this project?

Software Requirements analysis

Software Requirements Specification

Software Requirements Management

Software Requirements Validation

Software Requirements Documentation

Other

If other please specify

8. What is/was the duration of the project (from inception to delivery)?

Less than 6 months

6 months - 12 months

1 year – 2 years

2 years - 3 years

3 years - 4 years

4 years - 5 years

More than 5 years

B. Agile Methodology

1. If you are using agile methodology, which of the following agile approaches describes the one you are using/did use in the project?

SCRUM approach

Extreme Programming (XP) approach

Feature-Driven Development approach

Dynamic Systems Development Method

Lean and Kanban Software Development

Crystal

Other

If other please specify

2. What is your experience duration in Agile Requirements Engineering?

- Less than 6 months
- 6 months - 12 months
- 1 year – 2 years
- 2 years - 3 years
- 3 years - 4 years
- 4 years - 5 years
- More than 5 years

3. What is your experience duration in Agile Requirements Documentation?

- Less than 6 months
- 6 months - 12 months
- 1 year – 2 years
- 2 years - 3 years
- 3 years - 4 years
- 4 years - 5 years
- More than 5 years

***Note: Please answer this section after using the framework tool.**

C. IITKARD Framework Evaluation¹

For this section please refer to the following footnote

Please answer the following three questions with regards to this framework:

C.1 Efficiency

Efficiency of IITKARD relates to the use of all inputs in producing any given output, including personal time and energy. Also, efficiency minimizes the waste of resources such as physical materials, energy and time, while successfully achieving the desired output of IITKARD framework and its prototype tool.

1. Can you indicate how fast to convert the input of implicit thinking knowledge to output as a documentation after adopting the IITIKARD framework?

Very slow Slow Normal Fast Very fast

¹ The proposed framework shall be an extension JIRA framework & tool with additional following features:

- i- The ability of adding and managing software developers' issues, assumptions, suggestions, questions and opinions while detecting analyzing the requirements.
- ii- A formal documentation of the above feature to be tracked in the later software developing phases.

2. How fast is IITKARD displaying the implicit thinking knowledge of software engineers?

Very slow Slow Normal Fast Very fast

3. IITKARD inputs of implicit user stories and implicit thinking knowledge converted to outputs without cost of a user effort?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

4. IITKARD framework was coordinating agile requirements documentation in an efficient and appropriate way?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

5. IITKARD framework was monitoring agile requirements documentation in an efficient and appropriate way?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

6. IITKARD framework and the tool assist software engineers who are newly involved in the software project to understand how the requirement was developed?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

C.2 Usability

Usability is the degree of ease with which IITKARD framework tool can be used to achieve required goals effectively and efficiently. Usability assesses the level of difficulty involved in using IITKARD tool.

7. In a range of 1 to 5, what can you rate in terms of the difficulties of the usage of the IITKARD tool?

Very difficult Difficult Normal Easy to use Very easy to use

For answer 1, please specify your comment:

8. IITKARD framework and the tool completed the task successfully?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

9. The overall usability of the IITKARD framework and its prototype tool is satisfactory?

Not satisfied Moderately Satisfied Neutral Satisfied Very Satisfied

10. There are still many limitations to make the IITKARD framework and its tool usable?

Strongly Disagree Disagree Neither agree and disagree Agree Strongly agree

If you are agree or strongly agree, can you please specify your comment:

End of the questions & Thank you for your cooperation.

APPENDIX H

DEMOGRAPHIC RESULTS

(a) Respondent Age Rate

The overall age of respondents involved in this survey shows the highest of 76% age rate between 31-40 years old. While 16% of respondents' age ranged between 41-50 years old and 4% equally between 23-30 and 51-60 years old. The following table shows the total of respondents to this research questionnaire. Figure 1 summarizes the overall result of respondents' age.

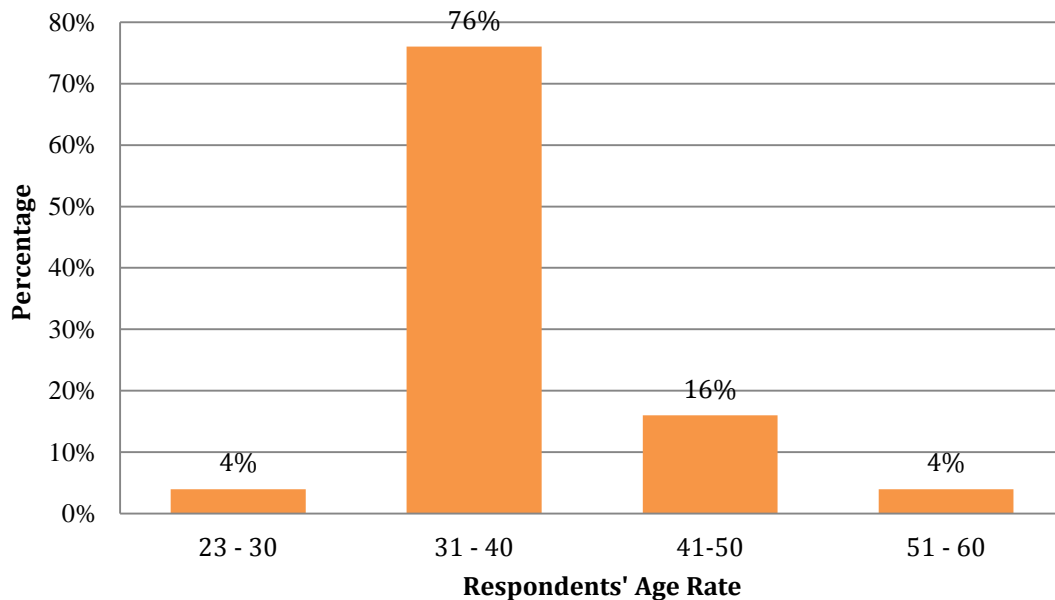


Figure 1 The bar chart of the overall results of the respondents' age rate

(b) Respondent Education Level

From Figure 2, it can be summarized that the majority of respondents were Master degree holders at 52%, followed by Doctorate at 36% and Bachelor Degree at 12%.

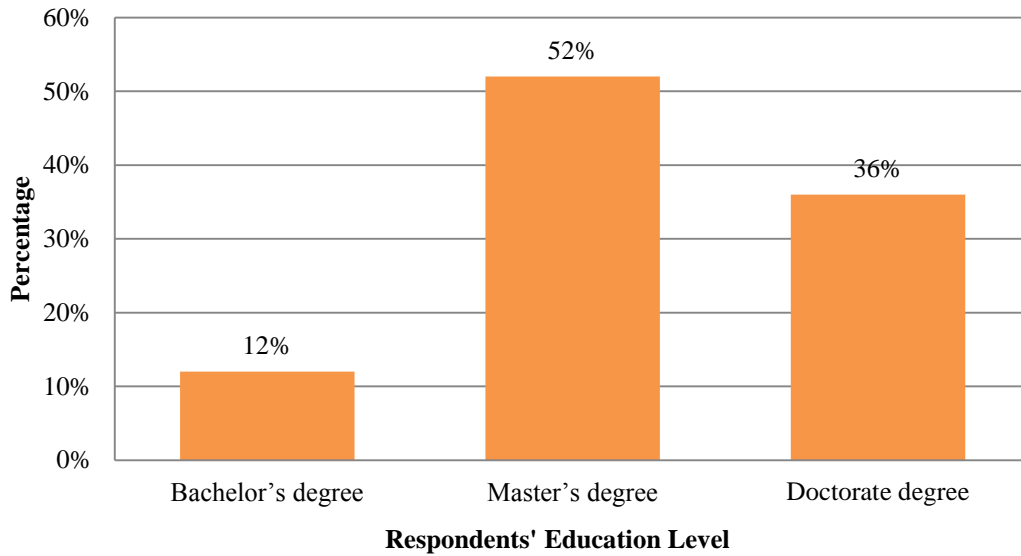


Figure 2 The bar chart of the overall results of the respondents' degree level

(c) Respondents' Position as a Software Engineer

Figure 3 shows that of the majority of the respondent hold a position of software engineers, some other were software architects, programmers and software testers.

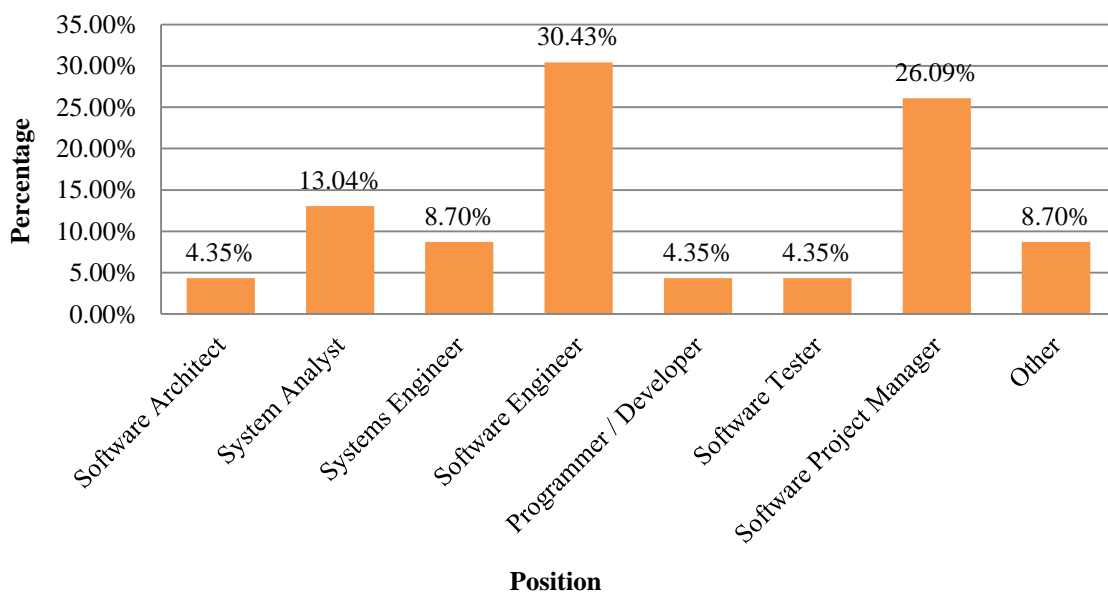


Figure 3 The bar chart of the overall results of the respondents' project position

(d) Respondents Software Project Type

The majority of the respondents in this survey previously participated in the software project of Web-Based (not Web services) at 47.83%. The detail of the findings on software project type is presented in Figure 4.

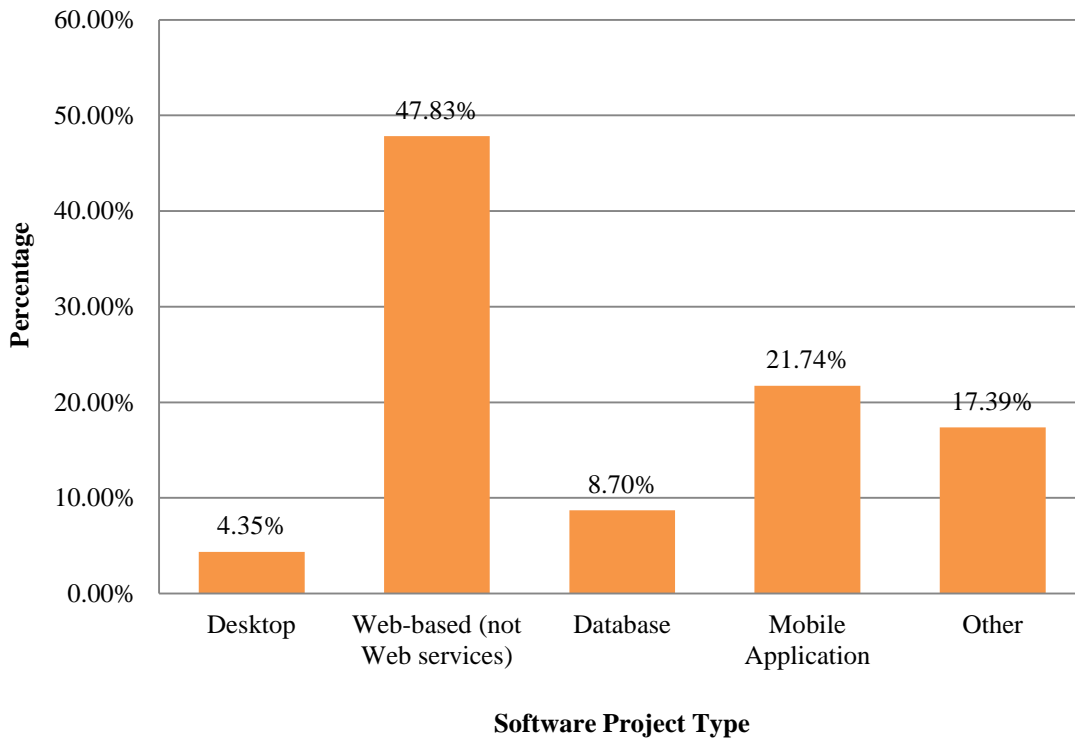


Figure 4 The bar chart of the overall results of the respondents' project type

(e) Respondent Software Application Domain

The total of 27.27% of the respondents are in finance/banking/insurance systems while 22.73% in education and 4.55% in health systems and human resource systems, and 40.91% in another software domain as shown in Figure 5.

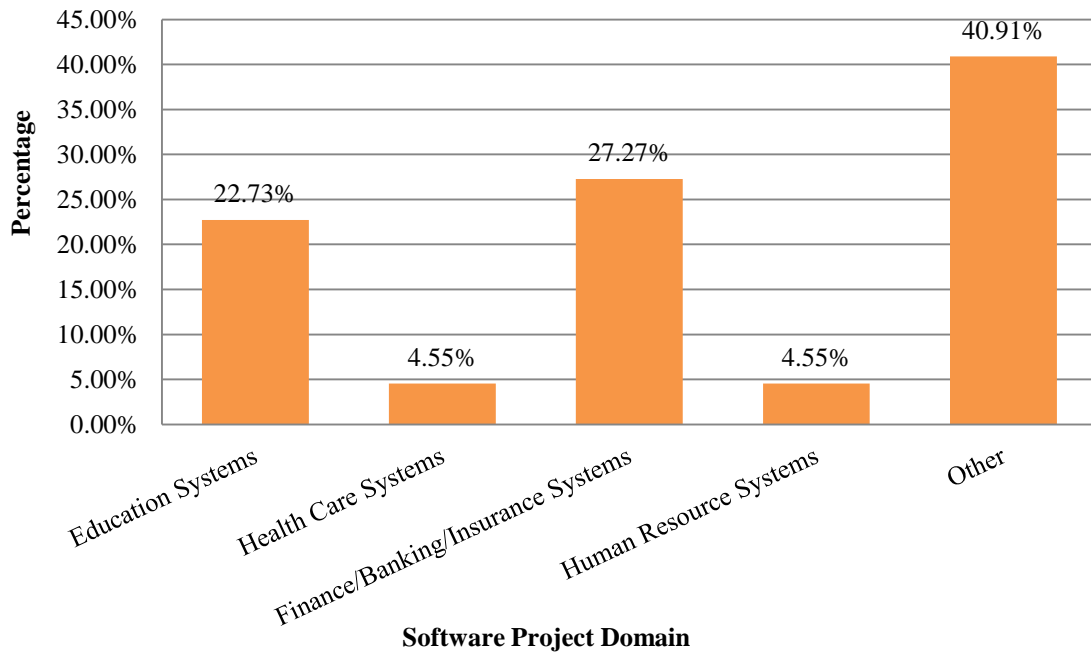


Figure 5 The bar chart of the overall results of the software application domain of the respondents

(f) Software Project Duration

The total of 56.52% of the respondents had experience in software project between 6 and 12 months, while 13.04% had less than 6 months and 17.39% had about 36 to 48 months experience²¹⁻, 8.70% between 48 to 60 months and 4.35% more than 60 months. Figure 6 summarizes the overall results of the software project duration.

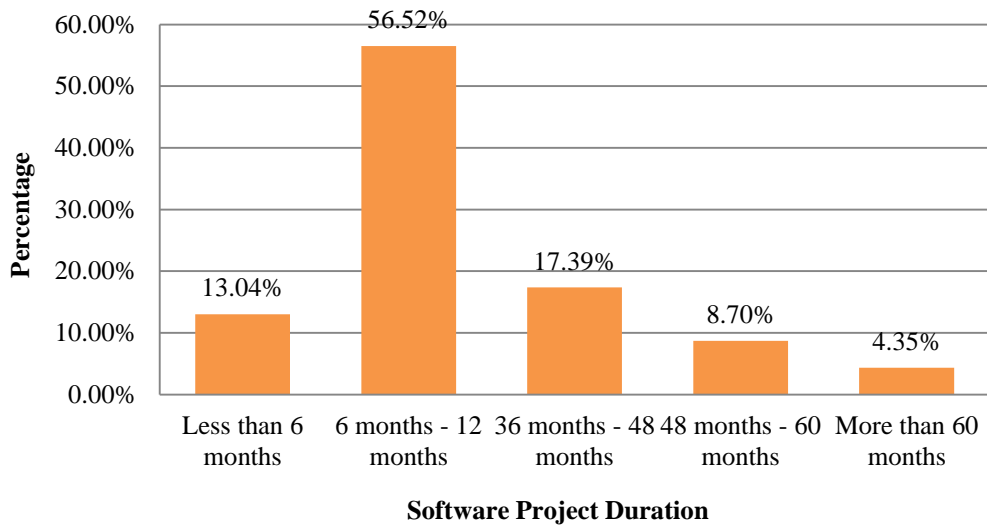


Figure 6 The bar chart of the overall results of duration of the software project

APPENDIX I

FOCUS GROUP DEMOGRAPHIC ANALYSIS

1. Respondents' Age

The overall experts' age rates were 50 % between 31-40 years old, while 20 % their age was between 41-50 years old, also 20% between 23-30 years and 10% was for 51-60 years rate as shown in Figure 1.

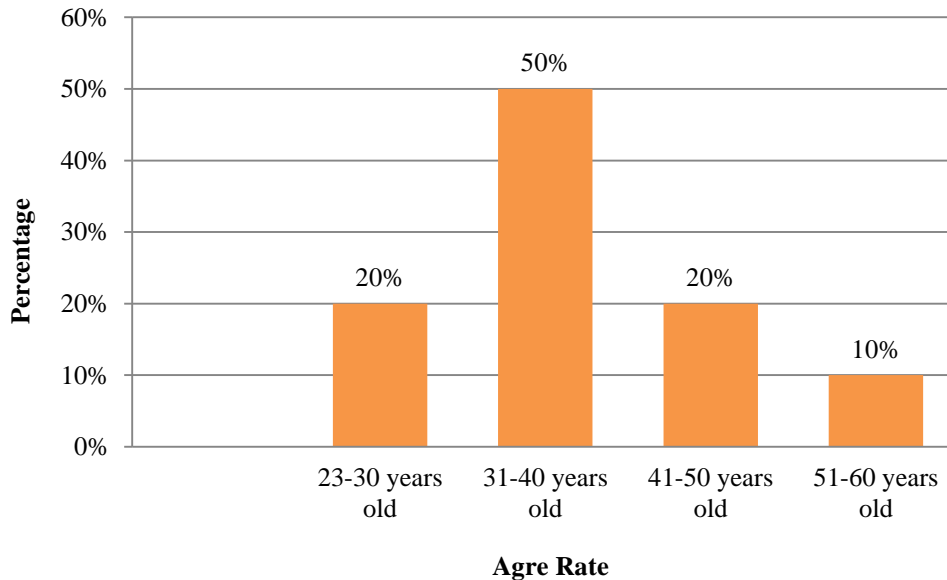


Figure 1 The bar chart of the overall results of the experts' age

2. Respondents' Qualification

Figure 2 shows that 50 % of experts were doctorate degree holders and 30 % were Master's degree while 10 % were Bachelor holders and 10 % were Diploma holders.

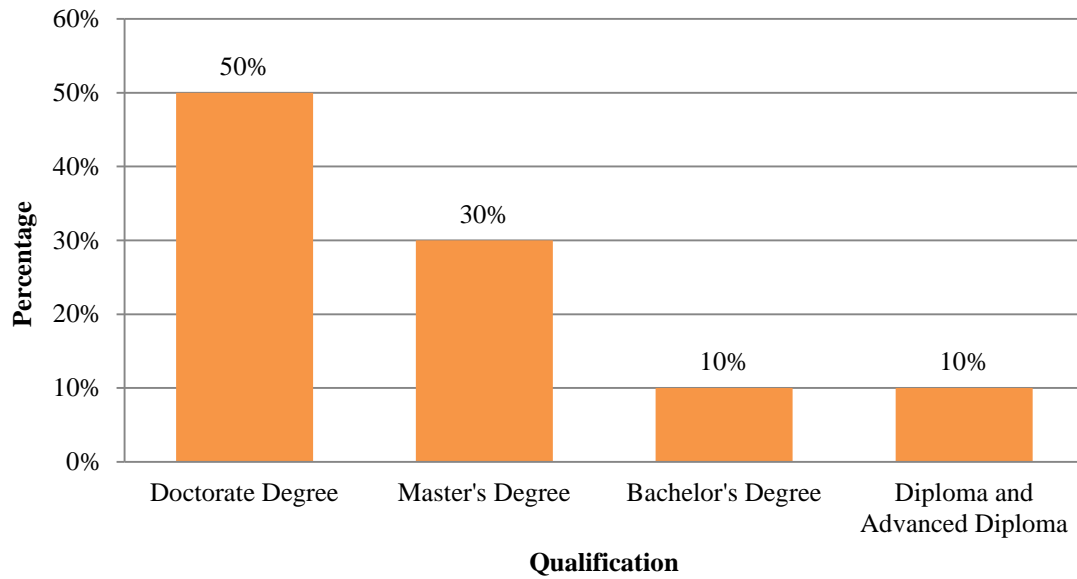


Figure 2 The bar chart of the overall results of the experts' qualification

3. Experience of Software Development Methodology

The majority of participants (80 %) had more than 5 years of experience in software industry, while 20 % had between 4 – 5 years as presented in Figure 3.

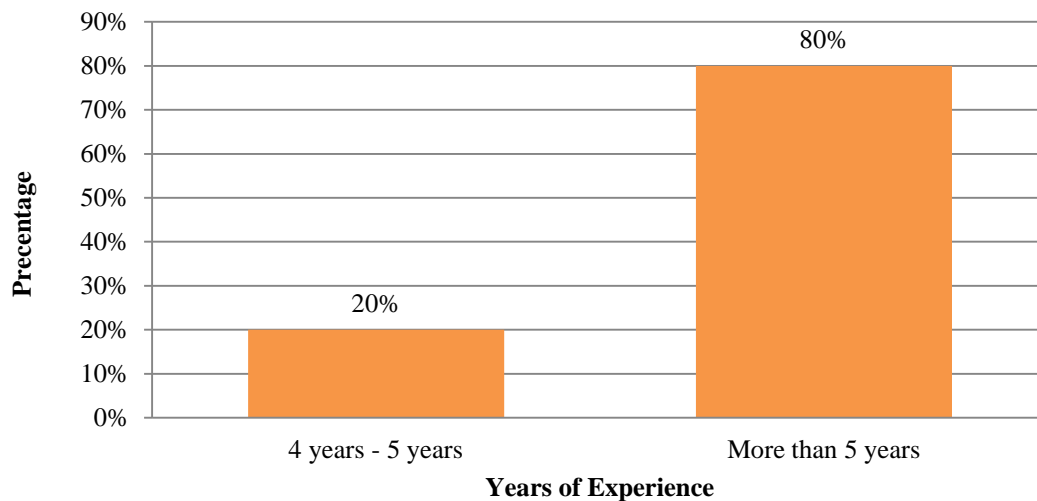


Figure 3 The bar chart of the overall results of the experts' experience in software development methodology

4. Experience of Agile Methodology

The total of 50 % of the participants had more than 5 years experience in agile methodology, while 30 % had between 4 – 5 years experience, and 20 % had between 3 – 4 years experience, as shown Figure 4.

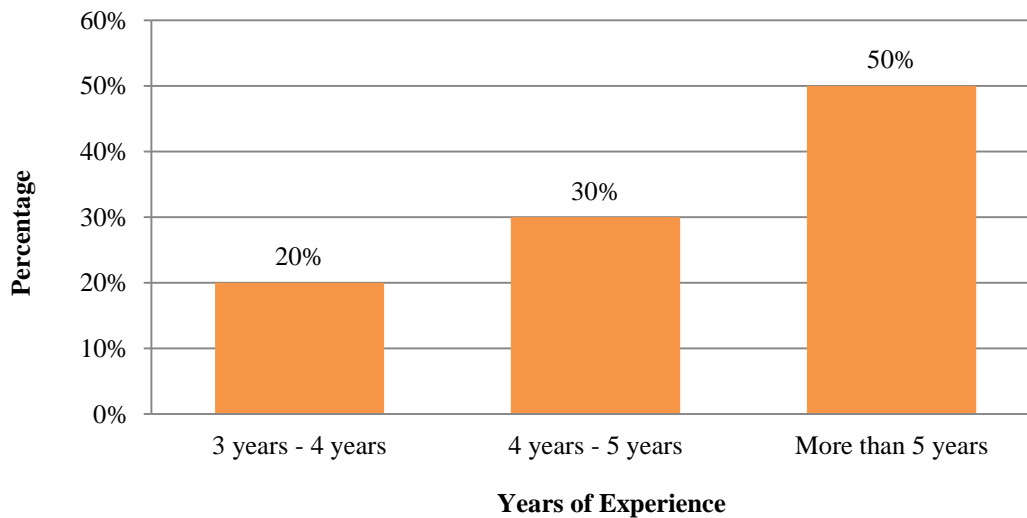


Figure 4 The bar chart of the overall results of the experts' experience in agile methodology

5. Scales of the Project

The total of 50 % of the experts involved in a medium-scale software project, while 30 % were involved in a small-scale project, and 20 % were involved in a large-scale software project, as shown in Figure 5.

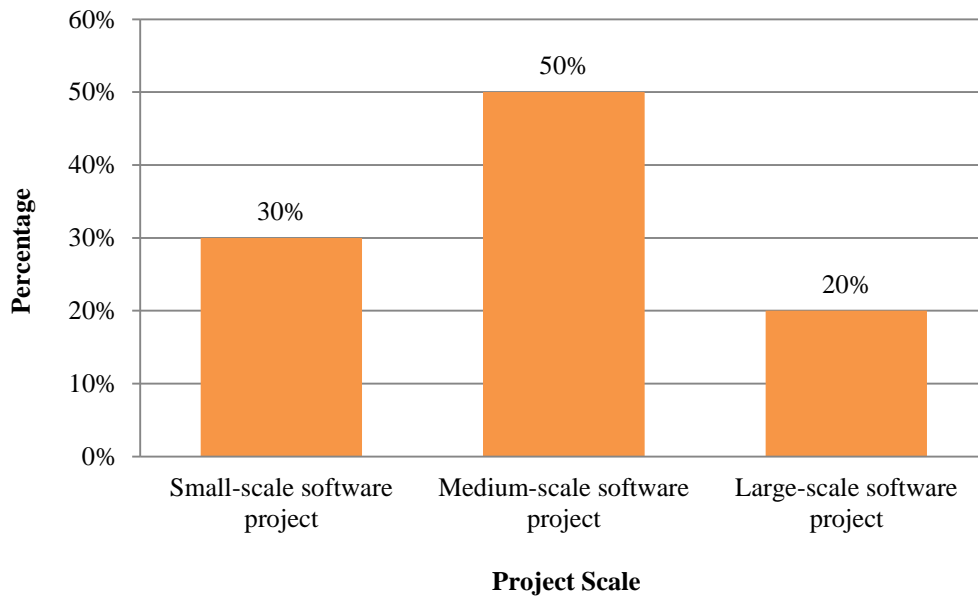


Figure 5 The bar chart of the overall results of the scales of software project

6. Software Type Categories

The total of 80% of the experts involved in a web-based software project, while 10 % were equally involved in a database software type project, and mobile application software project, as shown in Figure 6.

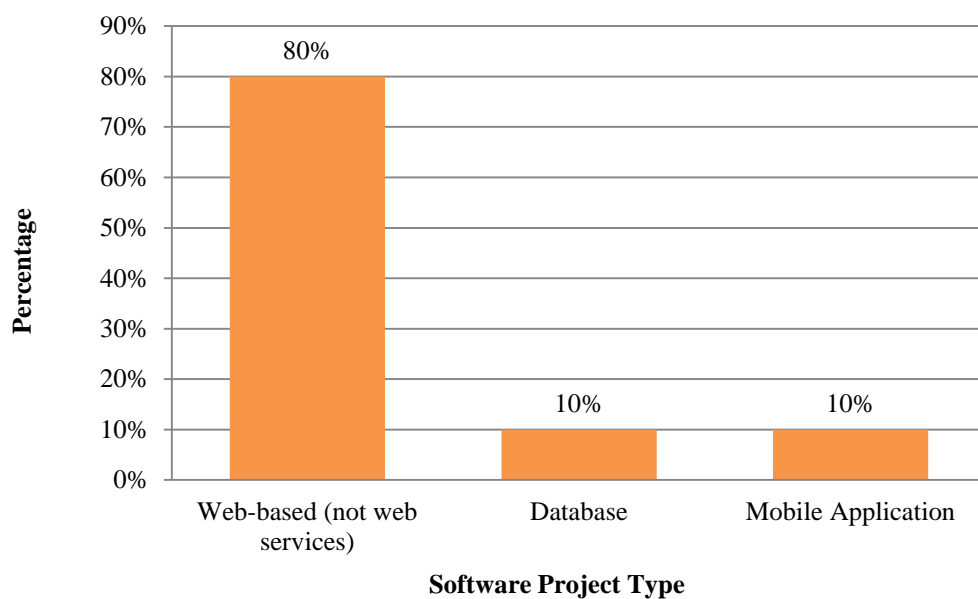


Figure 6 The bar chart of the overall results of software type categories

7. Requirements Engineering Practices

The total of 80 % of the experts involved in a software requirements analysis practice, while 10 % were equally involved in a software requirements specification and software requirements management, as shown in Figure 7.

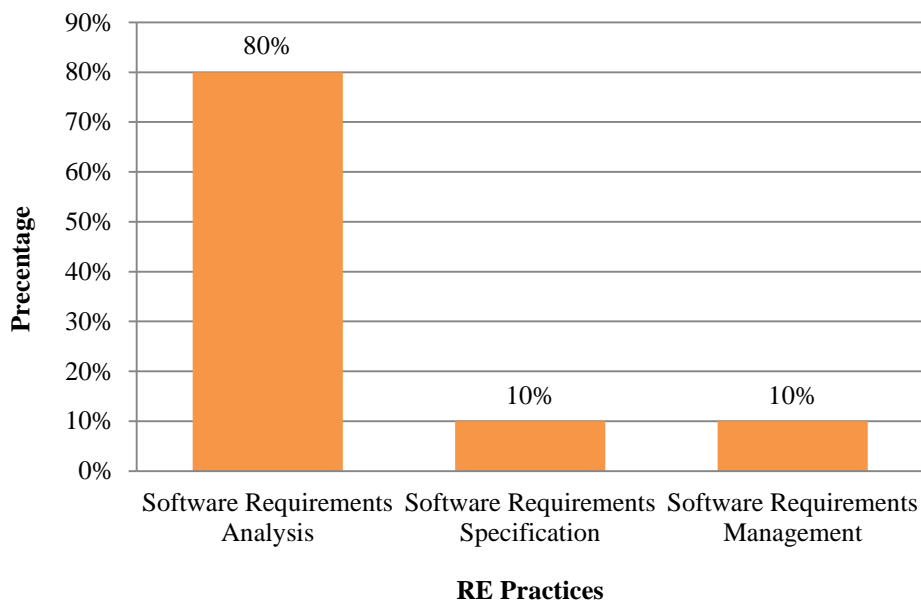


Figure 7 The bar chart of the overall results of requirements engineering practices

8. Software Project Duration

The total of 40% of the experts involved for 1 – 2 years in a software project, while 30% were involved for 6 -12 months in a software project and 30% involved for less than 6 months in a software project, as shown in Figure 8.

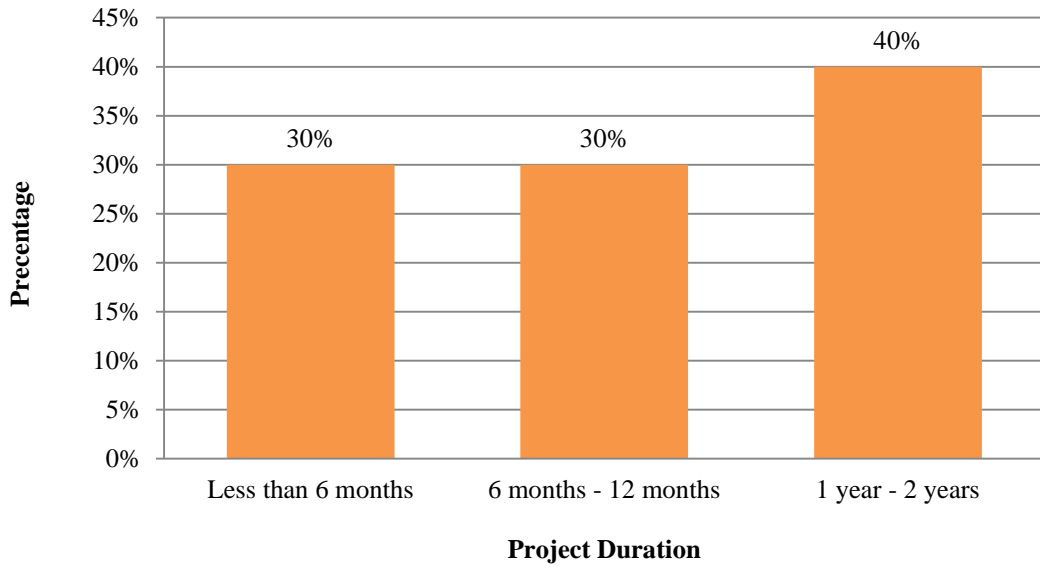


Figure 8 The bar chart of the overall results of software project duration

9. Agile Approaches

The total of 50% of the experts involved in Extreme Programming (XP) approach, while 40.00% of them involved in Scrum approach and 10% in Dynamic system development method, as shown in Figure 9.

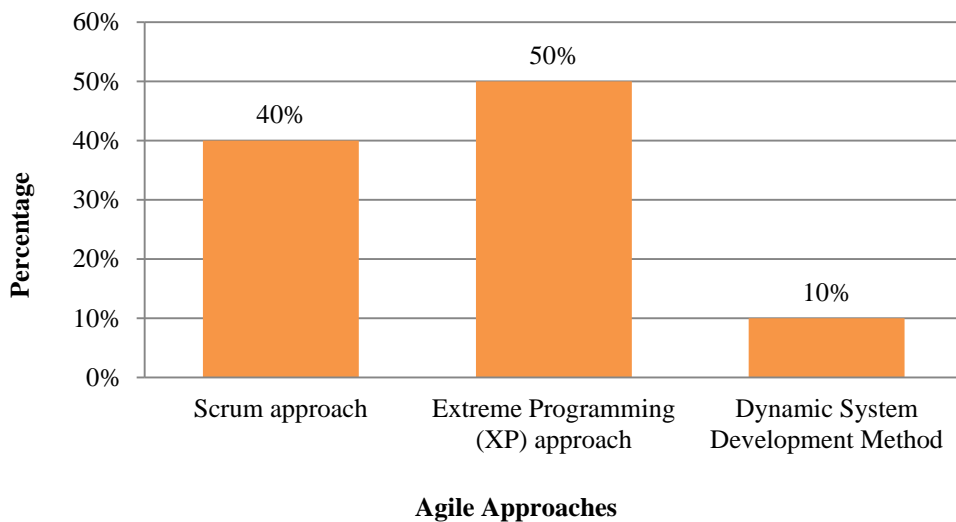


Figure 9 The bar chart of the overall results of agile methodology approaches

10. Agile Requirements Engineering Experience

The total of 50% had more than 5 years' experience in agile requirements engineering, while 40% had experience between 4 to 5 years, and 10% had experience from 1 year to 2 years, as shown in Figure 10.

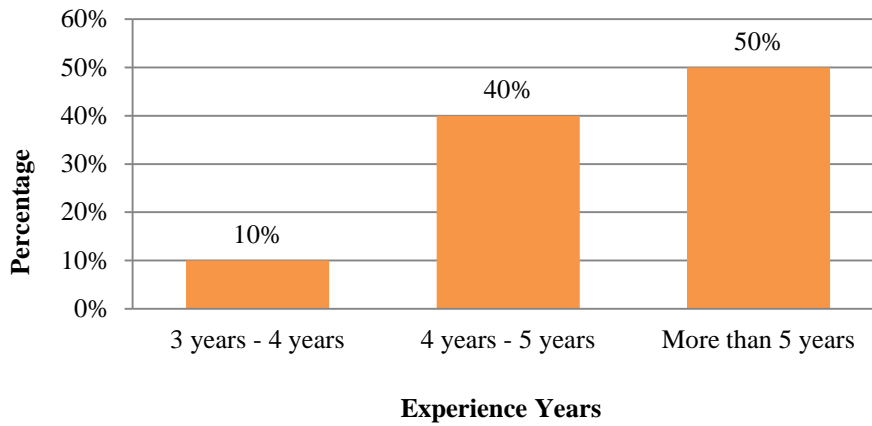


Figure 10 The bar chart of the overall results of experts' experience in agile RE

11. Agile Requirements Engineering Documentation Experience

The overall participants' experience in agile RE documentation was 50%, while 40% was for the rest of experience duration category, and 10% was for 3-4 years' experience as described Figure 11.

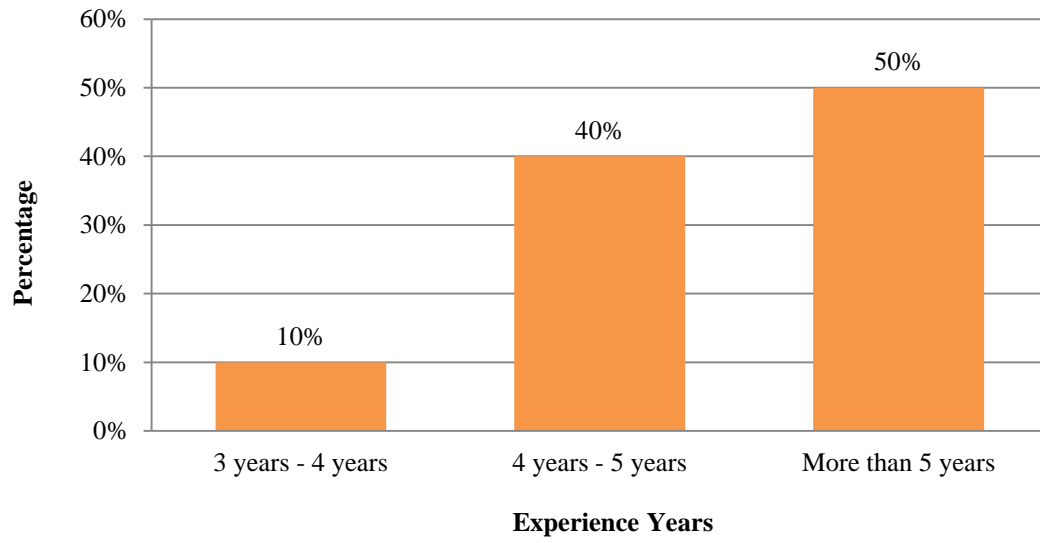


Figure 11 The bar chart of overall results of agile RE documentation experience

APPENDIX J

LIST OF PUBLICATIONS

Indexed Journal

1. **Elghariani, K.**, Kama, M. N., Firdous, N., Abubakar, N. A. 2018. Implicit Thinking Knowledge Injection Framework for Agile Requirements Engineering. International Journal of Advanced Computer Science and Applications. **(Indexed by Scopus)**

Indexed Conference Proceeding

2. **Elghariani, K.**, and Kama, M. N., 2016. Review on Agile requirements engineering challenges, in 3rd International Conference on Computer and Information Sciences, Kuala Lumpur, Malaysia, pp. 507-512. **(Indexed by Scopus)**