

Investigation of Adherence Degree of Agile Requirements Engineering Practices in Non-Agile Software Development Organizations

Mennatallah H. Ibrahim

Department of Computers and Information Sciences
Institute of Statistical Studies and Research
Cairo University
Cairo, Egypt

Nagy Ramadan Darwish

Department of Computers and Information Sciences
Institute of Statistical Studies and Research
Cairo University
Cairo, Egypt

Abstract—Requirements are critical for the success of software projects. Requirements are practically difficult to produce, as the hardest stage of building a software system is to decide what the system should do. Moreover, requirements errors are expensive to fix in the later phases of the software development life cycle. The rapidly changing business environment is highly challenging traditional Requirements Engineering (RE) practices. Most of the software development organizations are working in such dynamic environment, as a result, either by or without their awareness agile methodologies are adopted in various phases of their software development cycles. The aim of this paper is to investigate the adherence degree of agile RE practices in various software development organizations that are classifying themselves as adopting traditional (i.e. non-agile) software development methodologies. An approach is proposed for achieving this aim and it is applied on five different projects from four different organizations. The result shows that even the non-agile software development organizations are applying agile RE practices by different adherence degrees.

Keywords—*agile methods; agile requirements engineering practices; requirements engineering*

I. INTRODUCTION

Software development in its own is a very complex process and if requirement is not stable and keep changing from the requirement gathering to the development phase, it becomes very difficult to implement [16]. The rapidly changing business environment in which most organizations operate is challenging traditional RE approach. The traditional RE approach focus on gathering all requirements and preparing requirements specification document early before the beginning of the design phase. Software development organizations mostly deal with requirements that are highly volatile (i.e., requirement that tend to evolve quickly and become useless even before project completion), as a result, the early requirements gathering and specification is not suitable as it leaves no room to accommodate changing requirements later in the development life cycle. Furthermore, many other factors make the traditional RE inappropriate for the dynamic context in which software development organizations operate as [9]: (1) rapid changes in competitive threats; (2) stakeholder preferences; (3) development technology; and (4) time-to-market pressures.

Agile methods seek to address the challenges faced by the software development organizations that operate in such dynamic context. Many agile methods advocate the development of code without waiting for formal requirements analysis and design phases, based on constant feedback from the various stakeholders; requirements emerge throughout the development process [10]. In particular, several agile practices deal with requirements in order to implement them correctly and satisfy the needs of the customer [2]. Agile RE practices focus on the continuous interaction between the software application developers and the stakeholders to address the requirements evolution over time, prioritize the requirements, and deliver the most valuable functionalities firstly.

The aim of this paper is to investigate the adherence degree of agile RE practices in traditional software development organizations. The paper answers two questions which are: (1) Do the traditional software organizations apply any of the agile RE practices? ; (2) To what extent agile RE practices are applied in such organizations? A proposed approach has been followed to investigate the adherence degree of agile RE practices in non-agile software development organizations. Such investigation approach has been applied to investigate five projects developed by four different organizations. The result of the investigation is finally concluded and analyzed.

The paper is structured as follows; it is divided into seven sections. Section II introduces the related work. Section III presents an overview on requirement engineering. Section IV introduces an overview on agile methodologies are. Section V introduces the frequently used agile requirements practices. Section VI introduces the proposed investigation approach. Section VII introduces the application of the proposed approach. Section VIII summarizes the main points discussed in the paper. Section IX introduces the future work.

II. RELATED WORK

Several studies and research works are conducted to address the issue of agile RE practices and their challenges. In [2], it is stated that agile RE differs from traditional RE in that agile RE takes an iterative discovery approach. Case studies in [2] were conducted on two types of organizations: (1) organizations that characterize themselves as involved in agile or high-speed software development but didn't explicitly

follow any specific agile methods and (2) organizations that used XP, Scrum, or both explicitly. It was revealed that agile RE practices are adopted in both types of organizations by various adoption levels. The set of adopted agile RE practices are Face-to-Face Communication, Iterative RE, Extreme Prioritization, Constant Planning, Prototyping, Reviews & Test, and Test-Driven Development.

A systematic literature review is conducted in [7] on agile RE practices and their challenges. Such systematic literature review reveals seventeen agile requirements engineering practices which are Face-to-face communication; Customers involvement and interaction; User stories; Iterative requirements; Requirement prioritization; Change management; Cross-functional team; Prototyping; Testing before coding; Requirements modeling; Requirements management; Review meetings and acceptance tests; Code refactoring; Shared conceptualization; Pairing for requirements analysis; Retrospective and continuous planning. Also eight challenges posed by the practice of agile requirements engineering are identified which are: minimal documentation; customer availability; budget and schedule estimation; inappropriate architecture; neglecting non-functional requirements; customer inability; contractual limitations; and requirements change. The research conducted in [6] also reveals the following challenges of agile RE practices: cost and schedule estimation; non-functional requirements; customer access and participation

The research delivered in [1] is concerned with discussing the problem of requirements engineering activities conduction and it suggests some improvements to solve some of the challenges caused by agile requirements engineering practices in large projects. The paper also discusses the requirements traceability problem in agile software development and as well as the relationships between the traceability and refactoring processes and their impact on each other.

The research done in [13] suggests guidelines to improve RE using agile methodologies which are: (1) considering various point of views while eliciting requirements; (2) using various interviewing techniques; (3) considering verification; (4) early consideration of non-functional requirements; (5) adapting requirements management practices; (6) separating environment setup and product development.

III. REQUIREMENTS ENGINEERING (RE)

Requirements are the basis for every software project as requirements. Requirements define what the stakeholders, users, customers, suppliers, developers, and businesses in a potential new system need from the software project and also what the software project must do in order to satisfy all the determined needs [3]. Generally, RE process can be defined as a systematic process of developing requirements through an

iterative co-operative process of analyzing the problem, documenting the resulting observations, and checking the accuracy of the understanding gained [12].

Requirement engineer should work with the following Objectives [16]: (1) Engineer needs to focus on understanding customers and all the stakeholders' desire and their requirement. They should create it and manage it and it will reduce the risk of failure of the software and it should full fill customer's demand; (2) Requirement engineer should give emphasis to know the relevant requirement, remove the conflict and create consensus among the stake holders if any for any requirement. Create unambiguous documentation with given standards and manage requirements systematically.

As shown in figure 1, here are three major activities of RE process which are [12]: (1) Requirements Elicitation; (2) Requirements Documentation/Specification; and (3) Requirements Validation. First, requirements elicitation (also called requirements acquisition) is the activity through which the system's requirements are discovered and elaborated through consultation with stakeholders, from previous documents, and from domain knowledge; the proposed system's boundary is defined during this activity [4].

Second, requirements documentation is the activity that results in producing the output of the RE process which is requirements specification. Generally, there is a wide variety of ways for expressing a requirements specification; such ways are ranging from informal natural language to more formal graphical and mathematical notations [12]. Third, requirements validation is the activity that detects possible problems in the requirements specification before it is being used for software development.

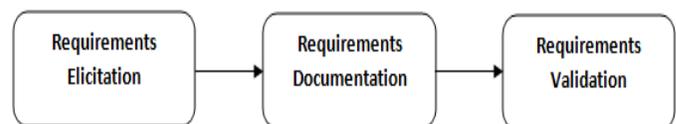


Fig. 1. Major activities of RE process

As mentioned before, requirements are the base of all software projects. However, their elicitation, management, and understanding are common problems for all development methodologies. Particularly, the requirements variability is a major challenge for all of the commercial software projects. According to a study of the Standish Group, five of the eight main factors for software project failure are dealing with requirements which are [2]: incomplete requirements; low customer involvement; unrealistic expectations; changes in the requirements and useless requirements (shown in TABLE I).

TABLE I. MAIN FACTORS OF PROJECT FAILURE [13]

| Software Project Problems | |
|----------------------------|------|
| Problem | % |
| Incomplete requirements | 13.1 |
| Low customer involvement | 12.4 |
| Lack of resources | 10.6 |
| Unrealistic expectation | 9.9 |
| Lack of management support | 9.3 |
| Change in the requirements | 8.7 |
| Lack of planning | 8.1 |
| Useless requirements | 7.5 |

IV. AGILE METHODS

Agile Methods are a family of development techniques which are designed to deliver products on time, within budget, and with high customer satisfaction [2]. This family includes several and very different methods. The common agile methods are Extreme Programming (XP) [11] and Scrum [11]. Agile methods embrace iterations where small teams work together with stakeholders to define quick prototypes, proof of concepts, or other visual means to describe the problem to be solved [15].

Agile Methods generally focus on the value of people to solve problems and share information, not on the process and a massive amount of documentation [2]. However, the people-orientation can represent a main weakness for Agile Methods since skills required to build good agile teams are not common [2]. Team members have to be excellent developers who are able to work in teams and have excellent communication skills since the team is self-organizing and cannot refer to a predefined process to share knowledge and solve problems.

The team tasks are to [15]: (1) define the requirements for the iteration; (2) develop the code; (3) create and run integrated test scripts. The users verify the iteration results. Verification occurs early in the development process allowing stakeholders to fine-tune requirements while the requirements still relatively easy to change. Figure 2 shows a generic agile development process features which are an initial planning stage, rapid repeats of the iteration stage, and some form of consolidation before release.

Different agile methods vary in practices and emphasis; but, they follow the same principles behind the agile manifesto which are [1]:

- Working software is delivered frequently (weeks rather than months).

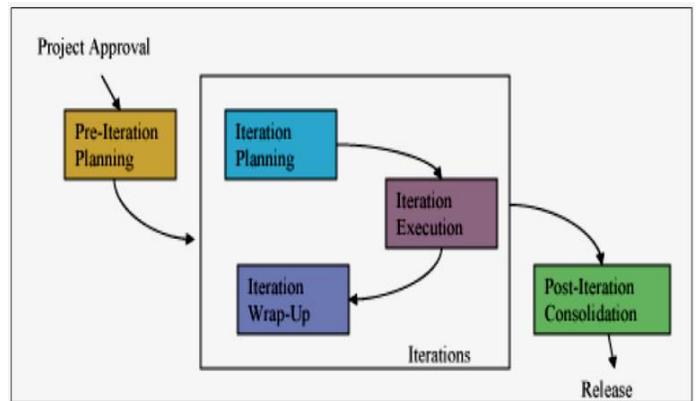


Fig. 2. A Generic Agile Development Process [5]

- Working software is the principal measure of progress.
- Customer satisfaction by rapid, continuous delivery of useful software.
- Even late changes in requirements are welcomed.
- Close daily co-operation between business people and developers.
- Face-to-face conversation is the best form of communication.
- Projects are built around motivated individuals, who should be trusted.
- Continuous attention to technical excellence and good design.
- Simplicity.
- Self-organizing teams.
- Regular adaptation to changing circumstances.

V. AGILE RE PRACTICES

Agile RE processes aren't centralized in one phase before development; they're evenly spread throughout development [10]. Several agile practices deal with requirements in order to implement them correctly and satisfy the needs of the customer [2]. Such practices focus on the continuous interaction between the stakeholders and the development team in order to overcome the problem of requirements volatility, incompleteness and vagueness. The common agile RE practices according to [10] are: Face-to-Face Communication, Iterative RE, Extreme Prioritization, Constant Planning, Prototyping, Reviews & Test, and Test-Driven Development (as shown in Figure 3). Each of such approaches has different aim that helps in performing requirement engineering processes effectively and efficiently.

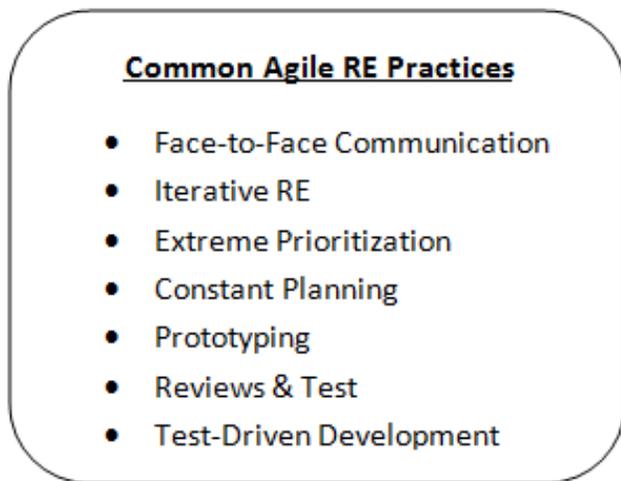


Fig. 3. Common Agile RE Practices

Face-to-Face Communication aims to effectively transfer requirements from the stakeholder to the development team directly without creating extensive documentation. Iterative RE aims to make agile RE continues at each development cycle where at the beginning of each cycle, the stakeholders meet with the development team to provide detailed information on a set of features that must be implemented. During this process, requirements are discussed at a great level of details.

Extreme Prioritization aims to implement the highest priority features at the beginning of the development process so that customers can gain the most business value early. The stakeholders prioritize their feature lists repeatedly during development life cycle as the stakeholder's and the developer's understanding of the project evolves, particularly when requirements are added or modified. Constant planning aims to accommodate requirements changes during the project development so that the system can be tuned easily to better satisfy customer needs. There are commonly two types of requirements changes [10]: (1) adding or dropping features; and (2) changing already implemented features. Generally, changes are easier to implement and cost less in agile development.

Prototyping aims to produces software application in the form of operational prototype, a refinement of the code created for experimentation with required features. This helps the organizations to rush to market as many of these organizations deploy these prototypes rather than wait for robust implementations. The ability to quickly deploy newer versions of the products on the Internet also contributes to this tendency.

Test-driven development is an evolutionary practice in which developers create tests before writing new functional code; such approach treats writing tests as part of a

requirements/design activity in which a test specifies the code's behavior [10]. Such practice aims to help developers team to write an explicit requirements specification. Review & Tests aim to take the advantage of frequent review meetings to validate requirements. At the end of each development cycle, a meeting is held between the developers, the stakeholder, quality assurance personnel and management personnel to validate the specified requirements.

The agile RE practices are usually used in combination with each other within the single organization. Agile RE practices provide benefits of improved understanding of customer needs and the ability to adapt to dynamic environment in which software development organizations operate. However, they pose several challenges to their adopting organizations. Therefore, such organizations should carefully compare the costs and benefits of agile RE practices in their projects.

VI. PROPOSED INVESTIGATION APPROACH

An approach for investigating the adherence degree of agile RE practices in non-agile software development organizations is proposed (shown in Figure 4). Such approach is consisting of seven steps. First step is to identify the list of agile RE practices that their adherence degree will be investigated. Second step is to design a questionnaire by formalizing the agile RE practices in form of questions in order to fit the purpose of our study; the simplicity of the designed questionnaire is considered. The questionnaire is used as a tool for collecting information from various participants (i.e., project managers and development team). We then checked the questionnaire against each of the identified agile RE practice to ensure that all of those practices are addressed within it.

Third step is to select the projects that will be subjected to the study where the selected projects are in different fields and by different organizations. Semi-structured interviews are held with various participants (i.e., project managers and one or more of the development team) in each project as a fourth step. If required, requirement documents of projects are reviewed to get more information about the projects requirements and sometimes to be an evidence of the information gathered from the participants and this is the fifth step.

In the sixth step, the information gathered from all participants in each project is analyzed to identify the adherence degree of each agile RE practice in every single project separately. In the seventh step, all the results (i.e., adherence degree) of each agile RE practices in all project are then aggregated in one table. The mean of all adherence degrees of an agile RE practice in all projects is calculated and then the agile RE practices are ranked according to the calculated mean.

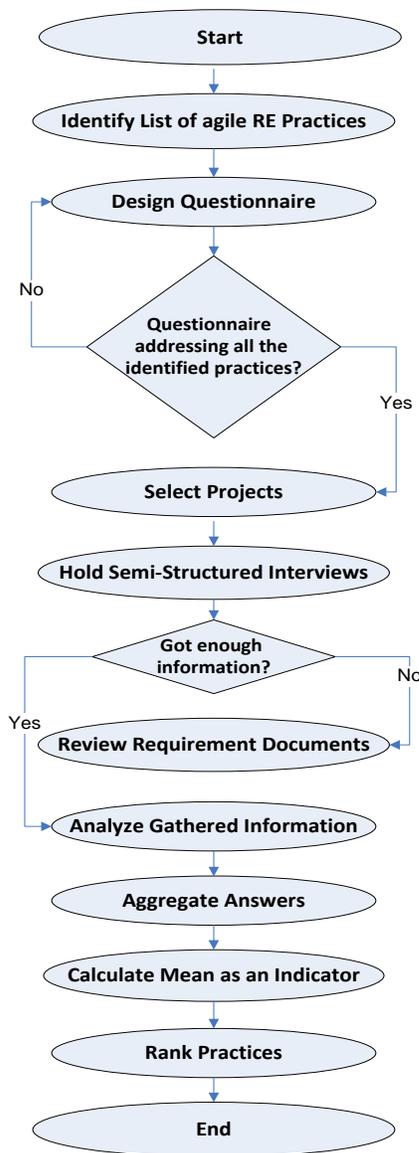


Fig. 4. Proposed Investigation Approach

VII. APPLYING INVESTIGATION APPROACH

The aim of applying the proposed approach is to measure the adherence degree (i.e. applying degree) of a set of common agile RE practices in software applications development. The list of examined agile RE practices are Face-to-Face Communication, Iterative RE, Extreme Prioritization, Constant Planning, Prototyping, Reviews & Test, and Test-Driven Development. Cases studies are applied in four different organizations that employ traditional approaches in developing their software projects (i.e., such companies is not adopting agile methods in neither their working procedures generally nor in their RE processes specifically). However, the results of our investigation show that they are actually applying agile RE practices without their awareness. These organizations are

located in Egypt's capital (i.e., Cairo). The required data is collected from the organizations through semi-structured interviews, questionnaires, in addition to reviewing requirements documents.

The first investigated project is a point of Sale banking application developed by an organization called "SEE Egypt". The investigation's result of this project shows that all of the agile RE practices are applied except Test-Driven development practice. All agile RE practices are either strongly or moderately applied. The strongly applied practices are Iterative RE, Extreme Prioritization, Constant Planning, and Reviews & Tests are, while Face-to-Face Communication and Prototyping are moderately applied practices.

The second investigated project is an application to display and deal with satellite receiver channels developed by an organization called "IG Company". The investigation's result shows that in this project all of the agile RE practices are applied except Test-Driven development and Constant Planning. All the practices are strongly applied.

The third investigated project is a website for faculty of computers and information, Cairo University called "FCI E-Community Website". This project developed by "Centre for the Study of Developing Societies – Cairo University". The investigation's result shows that all the agile RE practices are applied by different degrees. Face-to-Face Communication, Iterative RE, and Extreme Prioritization are applied strongly. Constant Planning, Prototyping and Reviews & Test are applied moderately. Test-Driven Development is weakly applied.

The fourth investigated project is an application to manage and control the employees' data developed by an organization called "Triple L Oil Service". The investigation's result shows that this organization is applying all the agile RE practices by different degrees. Face-to-Face Communication, Iterative RE, Extreme Prioritization, Reviews & Test and Prototyping are strongly applied, while Constant Planning and Test-Driven Development are moderately applied.

The fifth investigated project is an attendance system for Canadian university in Cairo which is developed by "Centre for the Study of Developing Societies – Cairo University" also.

The investigation's result shows that this all the agile RE practices are applied by different degrees. Face-to-Face Communication, Iterative RE, Extreme Prioritization, Constant Planning, Reviews & Test and Prototyping are strongly applied. Test-Driven Development is moderately applied. Prototyping is weakly applied.

By summarizing and analyzing the previous results (as shown in figure 5), it will be clear that both Iterative RE and Extreme Prioritization are the most applied agile RE practices (i.e., strongly applied) in the four organizations, while Test-Driven Development is the least applied practice (i.e., weakly applied). Face-to-Face Communication and Reviews & Test are considered to be strongly applied. Constant Planning and Prototyping are moderately applied. Figure 6 shows the detailed ranking of the applied agile RE practices.

| Practice | Project 1 | Project 2 | Project 3 | Project 4 | Project 5 | Adherence Degree |
|----------------------------|-----------|-----------|-----------|-----------|-----------|------------------|
| Face-to-Face Communication | 2 | 3 | 3 | 3 | 3 | 2.8 |
| Iterative RE | 3 | 3 | 3 | 3 | 3 | 3 |
| Extreme Prioritization | 3 | 3 | 3 | 3 | 3 | 3 |
| Constant Planning | 3 | 0 | 2 | 2 | 3 | 2 |
| Prototyping | 2 | 3 | 2 | 3 | 1 | 2.2 |
| Test-Driven Development | 0 | 0 | 1 | 2 | 2 | 1 |
| Reviews & Tests | 3 | 3 | 2 | 3 | 3 | 2.8 |

0: Not Applied; 1: Weakly Applied; 2: Moderately Applied 3: Strongly Applied

Fig. 5. Results Summarization

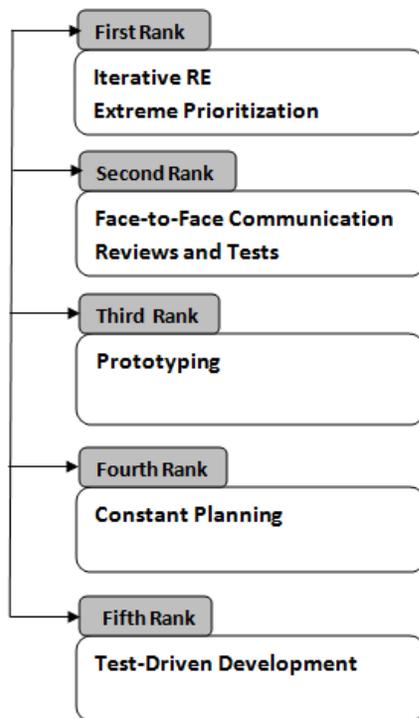


Fig. 6. Detailed Ranking of Applied Agile RE Practices

The results of the conducted case studies show that all agile RE practices are applied in the investigated projects, although, such projects are developed in organizations that are not applying agile methods in their applications development processes. The agile RE practices are applied by different degrees in each project. The most applied approaches are Iterative RE and Extreme prioritization, while, the least applied approach is Test-Driven Development. The results prove that the dynamic working environment and the unstable circumstances in which most of the organizations operate are challenging the traditional RE and enforcing such organizations to apply agile RE practices in their software development lifecycle even without their awareness.

VIII. CONCLUSION

Requirements are critical for the whole development cycle of software systems. Requirements generally define the tasks that the system should perform and the constraints posed on those tasks. Many factors cause the traditional RE to be inappropriate for software development as rapid changes in competitive threats; stakeholder preferences; development technology; and time-to-market pressures. Agile methodologies seek to address the challenges posed by dynamic environment in which most of the software development organizations operate. Agile methodologies focus on the continuous interaction between both the stakeholders and the development team.

The results of applying the proposed investigation approach show that although the investigated organizations consider themselves not applying agile methods in developing their applications, they are actually applying agile RE practices by different degrees without their awareness. The most applied practices are Iterative RE and Extreme prioritization. The least applied practice is Test-Driven Development.

IX. FUTURE WORK

There are many efforts can be done in the field of agile RE practices in the future. Briefly, the following points are expected to be focused:

- Extending the study to cover more software projects in many domains.
- Proposing an approach for evaluating the quality of applying agile RE practices using metrics.
- Evaluating the quality of applying hybrid agile methods to reveal the most used methods in conjunction.
- Using fuzzy logic in the evaluation process.

REFERENCES

- [1] A. D. Lucia, and O. Abdallah, "Requirements engineering in agile software development," *Journal of Emerging Technologies in Web Intelligence*, vol. 3, pp.212-220, 2010.
- [2] A. Sillitti, and G. Succi, "Requirements engineering for agile methods," *Engineering and Managing Software Requirements*, pp. 309-326. Berlin Heidelberg: Springer, 2005.
- [3] E. Hull, K. Jackson, and J. Dick, "Requirements engineering," London: Springer, 2005.
- [4] E. Nasr, J. McDermid and G. Bernat, "Eliciting and Specifying Requirements with Use Cases for Embedded Systems," *International Workshop on Object-Oriented Real-Time Dependable Systems*, IEEE 2002.
- [5] F. Paetsch, E. Armin, and F. Maurer, "Requirements engineering and agile software development," In *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 308-308. IEEE Computer Society, 2003.
- [6] H. Elshandidy, and S. Mazen, "Agile and Traditional Requirements Engineering: A Survey," *International Journal of Scientific & Engineering Research*, vol. 9, 2013.
- [7] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior*, 2014.
- [8] K. Beck, "Extreme programming explained: Embrace change." Addison-Wesley, UK., 1999.
- [9] K. Schwaber, M. Beedle, "Agile software development with scrum," Prentice Hall PTR, Australia, 2001.

- [10] L. Cao, and R. Balasubramaniam, "Agile requirements engineering practices: An empirical study," *Software, IEEE*, vol. 25, pp. 60-67, January 2008.
- [11] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile software development methods: Review and analysis," 2002.
- [12] P. Loucopoulos, and K. Vassilios, "System requirements engineering," McGraw-Hill, Inc., 1995.
- [13] S. Bose, M. Kurhekar, and J. Ghoshal, "Agile Methodology in Requirements Engineering," SETLabs Briefings Online, 2008.
- [14] V. E. Jyothi, and K. N. Rao, "Effective Implementation of Agile Practices - Ingenious and Organized Theoretical Framework", *International Journal of Advanced Computer Science and Applications*, vol. 2, March 2011.
- [15] V. Szalvay, "An introduction to agile software development," *Danube Technologies*, pp. 1-9, 2004.
- [16] V. Tripathi, and A. K. Goyal, "Agile Requirement Engineer: Roles and Responsibilities," *International Journal of Innovative Science, Engineering & Technology*, vol. 3, pp. 213-219, 2014.