Survey on Security Measures of Software Requirement Engineering

R. Saranya
Research Scholar
Madurai Kamaraj University
Madurai-21.

ABSTRACT
Software engineering concerns with wide use of engineering principles to achieve cost-effective software with potentiality to function on real machines. Requirement engineering in software development is more crucial. Everyone agrees that security is difficult. The requirements engineering principles are framed based on an idea that would engage the community overcoming complex problems. Security is about the prevention of several difficulties due to the presence of attackers behaving malicious activities. Software security is incredible due to the intrinsically complex task and the problem happens because of three main reasons such as networks are everywhere, systems are easily extensible and system complexity is rising.

The principle objective of requirement engineering research is not just to point out the fact about security risks keep on rising every day, but rather to defeat attack, just as any other system property. Security should be tackled at the beginning of the software lifecycle. This survey planned to concentrate on the requirements phase as security is a system property.

Keywords: Software engineering, requirement engineering, security, software development,

1. INTRODUCTION
Security in the recent research becomes a basic and established concern for software systems. The earlier stages of research involved huge increase in the amount of attacks but also the relive the attacks are performed on systems. Researchers believe that in order to secure a system from malicious attacks, a more focus is required on the beginning requirement phase. Relevant to other system functions and quality objects, security is more concentrated from beginning, as same as starting with requirements. Security is a nonfunctional requirement (NFR) that is even more dangerous in its significance, identical in its requirements. However, security is still be combined with all other functional and non-functional requirements and traced into efficient architectures, designs, and implementation. The unique characters and needs of security, results in difficulty as well as frequent unsuccessful security aspects, similar to other non-functional requirements using general purpose methods. Therefore, numerous unique and resultant techniques to security requirements engineering are recently proposed.

In the following survey intends in investigating a range of techniques or methods for security engineering more specifically regarding requirements. For the purposes of this survey, expose security requirements engineering into three more convenient phases, such as security requirements elicitation or acquisition along with analysis, security requirements specification and finally security requirements validation. The review is more about evaluation agenda in software development that concentrate on each phase. The evaluation agenda is compiled of a range of issues and response conditions devised in order to investigate working functions of existing approaches in terms of security support in requirements engineering phase.

Requirements engineering is more about the understanding of the goals to be reached by the predicted software system. The requirement processes involve three consistent functions or activities namely requirements elicitation, specification and validation. This study denotes an exhibition of requirements elicitation, specification model with respective concepts and issues targeting the software system.

The objectives of the survey is as follows

i. Spot the insufficiency or lack along with benefits of current security requirements techniques.

ii. Exploit the outcomes from (1) to decide existing desires of security requirements engineering.

iii. Verify whether certain techniques are valuable enough for security requirements and by this mean value further past requirements into architecture, design, and implementation.

iv. If no such technique is found valuable enough, the survey outline the foundation of the development of new security requirements engineering techniques with main objective to maximize the advantages and minimize the insufficiency found in existing techniques.

Along with the goals for survey, a range of primary hope is set to expect survey to be more valuable. The purpose of the survey is to outline the requirement engineering to be the appropriate for application of technical principles and approaches for developing, communicating, and managing requirements. Generally, security is highlighted at various stages in the software lifecycle, more specifically the requirements stage is highly important.

2. SECURITY METHODS IN SOFTWARE REQUIREMENT ENGINEERING
Security and privacy are the less important coverage and needs active research areas in computing for a long time. Security related techniques are early developed to secure information, coding, and more recently communication, from attacks or other intrusions through methods such as access controls and firewalls. But, major techniques are developed for past generations of computing world that is mostly within

a single closed administrative control like distinct project with a definite limit.

The exposed Internet world collectively with new enterprise and administrative practices increase the complication of security and privacy concerns noticeably. In such cases, system successfully communicates and distributes data with a huge set of other systems, regularly on wireless network. Due to the nature of open network, system suffers from malicious attacks during communication in wireless network. Traditional models and techniques for describing and examining security are badly set to tackle the much higher software problems that is hidden in internet-based scenario. Therefore, the security issues are to be solved in the initial requirement stages of the software development.

Requirement engineering is an essential phase of the software development lifecycle. Requirement engineering acts as the source for developing successful software with better understanding of requirements in the primary stage. Requirement engineering holds a variety of tasks for collecting requirements based on the user constraints and needs and also depending on stakeholders of the software product.

Requirement engineering is a primary action of a software development project and probably holds the ability to affect the entire software development actions if improperly implemented. Requirement engineering is a basic foundation and provides solution to the potential achievement of software projects. Requirement engineering is the most crucial and difficult process for the software development due to variety in the gathered requirements and because of fast variations in the requirement. Additionally, requirement engineering is constantly hard to increase accurate requirements that sustain reliable in huge and multifaceted systems.

Requirements are persistent, constantly distressing all development and protection phases of a system's development by offering the main data needed for attack. The necessities form a activate method for the development and protection efforts. Testing, for example, rely on an accurate report of quality and functional requirements to characterize the standard of accuracy against evaluation.

Requirement engineering is a processes used to find, analyze and validate system requirements. The objective of requirement engineering is to describe the principal requirements engineering activities such as elicitation, specification and validation as illustrated in the below figure 1.

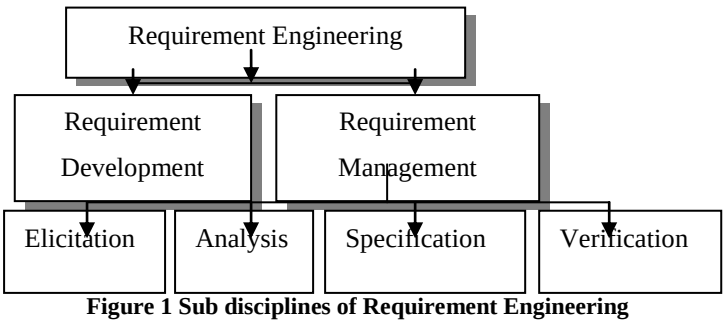

Figure 1 Sub disciplines of Requirement Engineering

Above figure 1 depicts the sub disciplines of requirement engineering. The requirement engineering intends in introducing techniques for requirements elicitation and analysis and to describe requirements validation.

Requirement elicitation is the process of determining, understanding, reporting, and realizing the user requirements and constraints for the system. Requirements analysis is the process of focusing the user's requirements and constraints. Requirements specification is the process of documenting the user's needs and constraints unmistakably and accurately. Requirements verification is the process of promising the system requirements are absolute, accurate, reliable, and understandable.

The security at the requirements stage is most essential concept for understanding not only level of secure software but also a secure way to guarantee user satisfaction with end product. In order to facilitate better understanding, each of elicitation specification and validation phases considers a significant aspect of the requirements engineering stage. Most of the literature investigated pointed out to a variety of activities involved in requirements engineering. The requirement activities ranged from elicitation to verification and maintenance of software requirements. A more concentration specifically in these requirement activities needs a better notice at secure software development.

The longer the system's life span, the more system is out to alterations in the requirements that effect from variations in the needs and aspects of its stakeholders. For instance, the customer needs is able to modify accordingly to original characterize provided by a challenging organization's artifacts.

The organization's enterprise targets and limits vary due to market needs, different rules, or different assurance policy. Various technologies and software tools such as operating systems probably vary the means the system is build. Such variations need fresh methods for proper management as a requirements variation management process.

Software security is a process that facilitates design and implements software that secure the information and tools controlled by that software. Software is a tool and is in need of appropriate security. Since the number of intrusions particularly targeting software is raising, the security of producing software requires assurance.

Security flaws, faults and bugs within the software cause all security susceptibilities found in software. In most situations, these faults are generated by two main reasons such as non-conformance, or a failure to satisfy requirements and an exception or error in the software requirements.

i. Non-accordance or a Failure to Satisfy Requirements

Non-accordance is uncomplicated and most common is a coding error or shortcoming or more difficult i.e., a restrained timing error or input validation error. The significant issue about non- accordance is that verification and validation techniques are intended to identify them. In addition, few security guarantee techniques are designed to prevent non-accordance. An improvement in requirement activities such as elicitation, specification and validation methods, through a software security assurance is able to improve the security of software.

ii. Exception or Error in Software Requirements

The common severe security problems with software based systems occur at the software requirement development resulting in incorrect, inappropriate, or incomplete software. Unfortunately, exceptions or errors in requirements are more complex to detect. For instance, the software probably performs determinately as per users needs, but the requirements are not properly able to tackle some system criteria. When the system faces these problem criteria, unpredicted and unwanted activities result. This type of problem cannot be handled within the software principles. Software principles results from a failure of the system and software engineering processes which developed and allocated the system requirements to the software.

The research demonstrates that most of the errors in the software development are openly associated to the faults made during time of requirement acquisition and elicitation phases. Survey computes several security concerns of requirement engineering comprising its tasks, tools, various approaches and techniques. The objective of the survey is to locate merits and demerits of requirements engineering processes, tools and techniques with particular reference.

3. ANALYSIS OF RECENT TRENDS IN SECURITY MEASURES OF SOFTWARE ENGINEERING

Surveillance of various research papers for security requirement engineering in software development with requirement activities are detailed below:

3.1 Requirement Elicitation

Requirements elicitation for software is able to gather predictable variations clearly over the anticipated duration of the software. The requirement acquisition means that the group of stakeholders is probably higher than for particular system requirements elicitation. The requirement elicitation comprises domain professionals, market experts, and others. Requirements elicitation concentrates on the scope, clearly gathering the predictable difference, and the adoption of use cases that define the variations that are expected to happen over the duration of the software.

Requirements elicitation is the software engineering actions in which stakeholder requirements are understood. Requirement elicitation involves identifying and prioritizing requirements as a process is complex to balance large software projects with many stakeholders. StakeRare [1] is a method that is able to balance the large software projects with many stakeholders using social networks and collaborative filtering to identify and prioritize requirements. Collaborative filtering [1] [12] is utilized to prioritize the requirements based on the stakeholder ratings.

StakeRare identified stakeholders and request identified stakeholder to suggest other stakeholders and stakeholder actions. Constructs a social network with stakeholders as nodes and their suggestions as connections, and prioritizes stakeholders using a range of social network calculations to decide software project authority. Using the information gathered from surveying and interviewing 87 stakeholders, the report demonstrated that StakeRare forecasts stakeholder requirements accurately.

StakeRare provides a more absolute and correctly prioritized list of requirements using collaborative filtering. But StakeRare involves certain challenges regarding collaborative user requirements. A process [7] is proposed to address the challenges of collaborative user requirements elicitation. The process promotes stakeholder agreement, through exploiting team dynamics to fetch a better understanding of requirements. However, the process needs a further work to evaluate the process within a larger-scale project.
A framework for security requirements elicitation and analysis [2] is build based on constructing a context for the system, denoting security requirements as criteria, and emerging satisfaction metrics for the security requirements. The system context is expressed using a problem-oriented information, then is validated against the security requirements during building of a satisfaction metrics. The accuracy of security requirement is achieved using satisfaction metrics. Furthermore, framework needs to solve risk analysis and understanding of formal arguments.

Software is more usable only if the interaction between the user and system is more proper. A persona is a HCI technique [3] that collects data about users in order to understand their characteristics. Personas provide a better understanding of the user, frequently unnoticed in software engineering developments. The main objective of HCI technique is to modify Personas to voluntarily make the technique into the requirements stage of standard software engineering developments.

Recently, numerous open source area incorporate forum to gather distributed stakeholders' requirements. But the requirements gathering tasks often suffer from the unformatted report and nonspecific consideration. A framework termed ReqForum [4] defined the metamodel of the requirement elicitation called lightweight forum-based requirements elicitation process. The proposed lightweight forum-based requirements elicitation process is feasible and the cost is economic.

A systematic review of recent research [5] in requirements engineering, specifically concerns stakeholder identification methods in requirements elicitation. The survey conveys existing rank of stakeholder identification in software requirement elicitation, the optimal practices suggested for its act, cost of erroneous identification in requirements quality, and, concerns which need to be improved.

A framework [6] is presented to elicit the software requirements and also to prioritize the software requirements. The proposed framework ranked the requirements by the qualified level of threat related with each requirement through AHP. As a further enhancement, AHP is only utilized to only determine the importance weight of the requirement and not to prioritize requirement [10]. Then prioritization is performed to manage the requirements. Prioritization through AHP results using threat level analysis is not more suitable for facing DOS attacks.

In addition, the appreciative inquiry [8] is also a successful method in eliciting software requirements. The appreciative inquiry technique gathers unique and latest requirements. But, appreciative inquiry technique in requirements elicitation process might comprise components for soft issues for different domain. Understanding of accurate method for gathering soft issues data provide better software development process in software engineering community. Even the requirement engineering process in Computer-Supported

Collaboration Working (CSCW) [9] domain faces the soft issues.

Requirements managers intends at maintaining their rest of requirements organized, reliable and modern throughout a project's life cycle. A semantic web technology gained more interest in requirements engineering community, with extra focus on requirements analysis. A prototypical implementation of a semantic guidance system [11] is used to support requirements engineers with extracting requirements using a semi-formal representation. Nevertheless, the tool evaluated is not more suitable for larger context like more requirements and a larger ontology. Finally, above all limitation concludes the need for a better requirement elicitation process.

3.2 Requirement Specification

Requirements specification at present comprises reports of a software projects with collection of requirements and artifact-specific requirements. The software requirements contain representative stakeholders that the different artifact-specific requirements reports fill in, develop, or instantiate. Requirement specification perfectly and correctly elaborates each of the vital requirements like roles, performance, design constraints, and quality attributes of the software.

Requirements engineering and software testing are established fields with lot of research. But, the communication between user and developer is lightly investigated beyond the concept of traceability in requirement specification. In order to address disconnectedness, a definition of requirements engineering and software test (REST) alignment taxonomy [13] is characterized by connecting the certain areas and a process to organize alignment but with less data availability.

A computational requirements document using a legal requirements specification language (LRSL) [14] in practice allows valid requirements open and available to customers, business analysts and software developers, alike. An extended Enterprise-Role Based Access Control (ERBAC) approach [15] with the notion of restraint rules enforced authorization processes. The approach also specified and enforced flexible security policies for energetic collaboration.

A systematic review is presented [16] related to the extraction of textual requirements specifications from software engineering models. The advantages of both lists of textual requirements and software engineering models are integrated by two approaches in the specification of system and software requirements reports.

The worst state of requirements techniques and usability is often reported as a concern for software project failure. The techniques [17] intends in improving the quality of the requirement specification. The main complexity is no top-level requirements are defined and fails to define in sufficient detail.

A requirement specification also extends the degree of requirement gatherings resulting in excess over specification. Icarus' predicament [18] is proposed to manage the pathologies of over specification and overdesign. The proposal provides a self-assessment to decide the impact of over specification and overdesign in an organization and resolves the conflicts.

An examination is an illustration of a software product to spot detection and identify software faults, including errors and variations from standards and specifications. The research proposes the plan of controlled testing [19] to investigate the impact of including UML diagrams in software requirement specification. In addition, the rate of defects detected in software requirement specifications and the effectiveness of the examination process in terms of time is also examined.

A complexity metric is proposed [20] using software requirement based specification to identify complexity of software directly at the time of requirement gathering in Software Development Life Cycle (SDLC) process. The requirements reports are unpreserved in organization with the existing models resulting in useless specification. Besides, requirements reports are generated in format that is not changeable so modification is not possible. At times the requirement collections results in overload leading to over specification. In addition, certain attacks also interacts the communications causing security threats during the requirement specification. At last, a better requirement activities more particularly specification with security is in need to enhance the requirement engineering.

3.3 Requirement Validation or Verification

Requirements verification is the process of validating or verifying the requirement elicitation, analysis and specification. Requirement validation comprises a wide critic group with various stages to validate the mistakes or errors found in the software. Initially, the common software requirements are verified. After, as each artifacts entry in the queue of artifact-specific requirements are verified.

A meta-level estimation of requirements [21] offers a useful and practical method that proved valuable to customer, analysts and designers to inform the fulfillment of sufficient requirements analysis. The proposed method achieved four confidence factors and used a goal oriented framework with an effortless ordinal scale to enlarge a method for assessing confidence.

One of the most challenging concerns of requirement engineering is the detection of inconsistencies between requirements. In order, to address this issue, REInDetector [22] a knowledge-based requirements engineering tool is developed to facilitate automatic detection of a range of inconsistencies. REInDetector provides aids to elicit, structure, and control requirements with well-known capabilities for gathering the domain knowledge and the semantics of requirements.

The objective of roadmap [23] is to review the state-of the-art and to spot serious challenges for the systematic software engineering of self-adaptive systems. The area of self-adaptive systems is vast and multidisciplinary. Software engineering processes are basically linked with design-time. But, engineering of self-adaptive systems also need run-time processes for facing alteration. The Capability Maturity Model (CMM) [25] becomes an accepted method for enhancing software development processes with the objective of establishing high-quality software within cost and intended cycle time. The main focus is on CMM level 5 projects from several organizations to revise the effects of extremely grown-up processes on effort, quality, and cycle time.

Most software quality research involves attention on identifying faults during requirement validation. A systematic literature review [24] is presented to illustrate taxonomy of

errors like source of faults that exists during the requirements phase of software lifecycle. This taxonomy is projected to facilitate developers during the requirement examination process and to enhance complete software quality. Finally, a requirement validation plays a significant role in providing a better end-result. The limitation of the various above existing methods in terms of their metrics are detailed in the below section.

4. PARAMETRIC EVALUATIONS

A survey of total 12 approaches is elaborated with distinct demerits in order to address security specifically for requirements engineering. Based on the results of the survey, a variety of observations and propose recommendations are provided for improving security requirements engineering. A table below depicts the parametric evaluations of the various approaches.

Techniques	Parameters											
	Integrity	Confidentiality	Availability	Masquerading	DOS Attack	Traceability	Success Rate	Accountability	Attack Tolerability	Fault Localization	Computational Cost	Requirement Validation
StakeRare				Y			Y			Y	Y	
Framework For Security Requirements Elicitation And Analysis		Y	Y									Y
HCI Personas Technique			Y	Y					Y			
Lightweight Forum-Based Requirements Elicitation Process						Y					Y	
Taxonomy for Requirements Engineering and Software Test Alignment (REST)						Y	Y					Y
Legal Requirements			Y			Y						Y
Specification Language (LRSL)												
Enterprise-Role Based Access Control (ERBAC) approach	Y					Y		Y				
On the generation of Requirements Specifications from Software Engineering Models								Y			Y	Y
Meta-level Method for assessing Confidence During Requirements Analysis		Y	Y							Y	Y	
REInDetector, a knowledge-based requirements engineering tool					Y			Y	Y			
Software Engineering for Self-Adaptive Systems: A Research								Y	Y			

Note: Y-yes

In large software projects, requirements elicitation faces three difficulties namely information overload, insufficient stakeholder contributions, and biased prioritization of requirements. Even though StakeRare [1] is hoped to solve information overload, insufficient stakeholder contributions, and biased prioritization of requirements for large software projects, the difficulty arise in stakeholder rankings. Stakeholders find it difficult to point out requirements they actively do not want in rank list. Even the attacker attempts to

act as if stakeholders, to increase the ranking rate casing a security threat. The act of attacker behaving as if like stakeholder tends to masquerading attack.

Vulnerabilities found in framework of [2] are removed through either alteration of the crisis, appending of security functional requirements, or totaling of trust assumptions. Another area for future work is risk analysis and partial satisfaction. However satisfaction metrics solves the requirement issues, the framework still requires an inner metrics to face as well as to indicate risks and satisfaction to security requirements.

Almost all the Personas steps hold a sequence of deficiencies regarding process characterization and artifact report. Modified Personas in HCI techniques [3] enriched software requirement process. However, the requirements activities mostly in requirements elicitation and requirements analysis are affected by incorporating Personas. ReqForum [5] still requires an even more attention on improving applicability and usability of prototypes.

However, in requirements elicitation survey [6], all stakeholder identification for granted do not go ahead of denoting who the stakeholders may be. Even though the truth that success and quality in software products rely to a large scope on requirements specification quality. The taxonomy on requirement engineering and software testing [13] needs a better specification. Taxonomy also predicts suitable characterize alignment in other domains of software engineering, but the information is never been available without a weak association link causing less availability of data.

The legal requirement specification language (LRSL) [14] capabilities still needs to allow better design. Additionally, LRSL needs to avoid or debug specifications, in order to eliminate ambiguity and categorize requirements around vital subjects. The Enterprise-Role Based Access Control (ERBAC) approach [15] suffers from the issue of optimization of restraint rules. The authorization of the method is probably improved through integration of trust and delegation.

The requirements documentation is not maintained in synchronization with the existing models [16] leading to inefficient specification. Moreover, requirements documentation are produced in format that is not modifiable, therefore corrections are not performed. The bidirectional traceability association between models and textual requirements is inappropriate. In addition, the requirements documentation structure model needs to be proper for ease understanding of requirements. Sometimes the requirement gathering probably results in excess collection causing over specification.

More specification and elicitation of requirement definitely need a validation to justify the requirement gatherings. Meta-model [21] provided better confidentiality during requirements analysis but needs a requirement validation. Even though the method for assessing confidence during requirements analysis achieved better privacy, the requirement validation is unnoticed.

REInDetector [22] is unable to detect the inconsistencies related with the requirements that are not expressible in description logic. The reason for this limitation is restricted support for sequential operators in description logic. Thus, a requirement such as user's availability in the system is not expressed in REIn-Detector. Hence, a further examination of REInDetector is needed for further constructors to permit temporal properties of the system.

The four party namely modeling dimensions, requirements, engineering, and assurances in roadmap [23] outline new challenges that software engineering community suffers in self-adapting software systems. These challenges are due to dynamic character of adaptation. This dynamic feature conveys uncertainty in some standard software engineering principles and techniques.

Based on the evidence collected from the review and the data gathered from initial studies [24] narrates the requirement error taxonomy. The review illustrates the various mistakes and error found in the existing mechanisms. Even with such potential approaches, faults still exist. Therefore, a particular focus on faults does not eventually lead to the removal of all faults.

5. CONCLUSION

A systematic review on security measures of software requirement engineering concludes the pros and cons of the existing approaches. Discussions regarding current trends in software engineering define the demands of security and privacy essential of a system. The parametric evaluation provides overall software requirement engineering performance.

In order for security to be built-in a software system, a good security requirements engineering approach should be elected for efficient process. This survey motivation helps in the election of proper efficient approach. The survey provides the researchers not only with reasons to the responses given to each issue, but also a comparison from different aspects of all techniques surveyed. The facts collected from the review and the data gathered from studies motivated further investigation on usefulness of developing novel security system.

5.1 Future Direction

Survey conveys the uncovered variety of areas in security requirements engineering with lack of support.

i. An extra attention is required more specifically in the parts of requirement elicitation, specification and validation.

ii. A new approach is necessary for requirement elicitation process based on stakeholder suggestion and mutual filtering, to overcome inconvenient and infrequent during interaction of developers and end-users in different organizations or enterprise. In order to address the problems, Stakeholder suggested model is required that comprise Identification of large project, Analysis of requirements, Identify and prioritize stakeholders, Predict requirements, Prioritize requirements.

iii. A definite software specification is in need as a short declaration of the requirements that the software must assure. Through the novel declared requirements, software must support services and capabilities to users, granting them to achieve the specific organizational visions. Good requirements are only guaranteed by the correct grouping of three scopes such as community, administration and expertise.

iv. In addition, above requirement elicitation and specification are to be validated with a novel technique in terms of quality, accuracy and security. Determine challenges in aligning requirements engineering and verification in a

large-scale project. Requirement validation generally needs stakeholders to be straightforwardly involved in reviewing the requirements artifacts. A novel verification technique probably shows requirements consistent with its specification.

Above areas of need become part of future work. On fulfilling the above needs, software requirement engineering is able to achieve better security and privacy for a complete system.

REFERENCES

[1] Soo Ling Lim, and Anthony Finkelstein, "StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation", IEEE Transactions on Software Engineering, Volume: 38 Issue: 3, May-June 2012.

[2] Charles B. Haley, Robin Laney, Jonathan D. Moffett, and Bashar Nuseibeh, "Security Requirements Engineering: A Framework for Representation and Analysis", IEEE Transactions on Software Engineering, vol. 34, no. 1, January/February 2008.

[3] Silvia T. Acuña a, John W. Castro a, Natalia Juristo, "A HCI technique for improving requirements elicitation", Elsevier Science Direct on Information and Software Technology, July 2012.

[4] Han Lai, Rong Peng, Dong Sun, Jia Liu, Chongqing, "A Lightweight Forum-based Distributed Requirement Elicitation Process for Open Source Community", International Journal of Advancements in Computing Technology, Volume4, Number7, April 2012.

[5] Carla Pacheco, Ivan Garcia, "A systematic literature review of stakeholder identification methods in requirements elicitation", Elsevier, Systems and Software, April 2012.

[6] Mohd. Sadiq, Mohd. Shahid and Shabbir Ahmad, "Adding Threat during Software Requirements Elicitation and Prioritization", International Journal of Computer Applications (0975 – 8887), Volume 1 – No. 9, 2010.

[7] Aida Azadegan, K. Nadia Papamichail, Pedro Sampaio, "Applying collaborative process design to user requirements elicitation: A case study", Elsevier, Science Direct on Computers in Industry, May 2013.

[8] Omar Isam Al Mrayat, Norita Md Norwawi, and Nurlida Basir, "Appreciative Inquiry in Eliciting Software Requirements", International Journal of Computer Science and Electronics Engineering (IJCSEE) Volume 1, Issue 3, 2013.

[9] Rahman A. N. and Sahibuddin S., "Extracting Soft Issues during Requirements Elicitation: A Preliminary Study", International Journal of Information and Electronics Engineering, Vol. 1, No. 2, September 2011.

[10] Mohd. Sadiq, Javed Ahmad, Abdul Rahman, R. Suman and Shweta Khandelwal, "More on Adding Threat during Software Requirements Elicitation and Prioritization", International Journal of Engineering and Technology, Vol.2, No.3, June 2010.

[11] Stefan Farfeleder, Thomas Moser, Andreas Krall, Tor St˚alhane, Inah Omoronyia, and Herbert Zojer, "Ontology-Driven Guidance for Requirements Elicitation", Springer / ACM, The semantic web: research and applications, 2011.

[12] P.Deepika1, P.S.Smitha, "Requirement elicitation based Collborative filtering using social Networks", International Journal of Emerging Technology and Advanced Engineering, Volume 3, Special Issue 1, January 2013.

[13] M. Unterkalmsteiner, R. Feldt, T. Gorschek, "A Taxonomy for Requirements Engineering and Software Test Alignment", ACM Transactions on Software Engineering and Methodology, January 2013.

[14] Travis D. Breaux, David G. Gordon, "Regulatory Requirements Traceability and Analysis Using Semi-Formal Specifications", Springer /ACM, Requirements Engineering: Foundation for Software Quality, 2013.

[15] Yuqing Sun, Bin Gong, Xiangxu Meng, Zongkai Lin, Elisa Bertino, "Specification and enforcement of flexible security policy for active cooperation", Elsevier Science Direct on Information Sciences, 2009.

[16] Joaquín Nicolás, Ambrosio Toval, "On the generation of requirements specifications from software engineering models: A systematic literature review" Elsevier science Direct on Information and Software Technology, 2009.

[17] Tom Gilb, "What's Wrong with Requirements Specification? An Analysis of the Fundamental Failings of Conventional Thinking about Software Requirements, and Some Suggestions for Getting it Right", Journal of Software Engineering & Applications, 2010.

[18] Alex Coman, Boaz Ronen, "Icarus' predicament: Managing the pathologies of overspecification and overdesign", Elsevier science Direct on Project Management, 2010.

[19] Balsam A. Mustafa,Hamayoon Ghafory, "Investigating the Inspection Effectiveness of Software Requirements Specification with UML Diagrams: A Concept Paper", Journal of Emerging Trends in Computing and Information Sciences, Vol. 4, No. 11 November 2013.

[20] Olabiyisi S.O., Adetunji A.B, Olusi T.R, "Using Software Requirement Specification as Complexity Metric for Multi-Paradigm Programming Languages", International Journal of Emerging Technology and Advanced Engineering, Volume 3, Issue 3, March 2013.

[21] Kenneth Boness, Anthony Finkelstein, Rachel Harrison, "A method for assessing confidence in requirements analysis", Elsevier Science Direct on Information and Software Technology, May 2011.

[22] Tuong Huan Nguyen, Bao Quoc Vo, Markus Lumpe, and John Grundy, "REInDetector: A Framework for Knowledge-based Requirements Engineering", IEEE/ ACM, Automated Software Engineering, 2012.

[23] Betty H.C. Cheng, Rog´erio de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, "Software Engineering for Self-Adaptive Systems: A Research Roadmap", Springer, Self-Adaptive Systems, 2009.

[24] Gursimran Singh Walia, Jeffrey C. Carver, "A systematic literature review to identify and classify software requirement errors", Elsevier Science Direct on Information and Software Technology, 2009.

[25] Manish Agrawal and Kaushal Chari, "Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects", IEEE Transactions on Software Engineering, vol. 33, no. 3, March 2007.