A Systematic Study of Requirement Volatility during Software Development Process

Prof(Dr.) Harsh Dev[1], Ranjana Awasthi[2]

[1]Prof(Dr.) Harsh Dev
Professor,
Deptt. of CSE,
PSIT, Kanpur

[2]Computer Science & Engineering, Singhania University
Rajasthan, India

Abstract

The amount and complexity of software produced today stagger the imagination. Software development strategies have not kept pace with them. Today's software and system's engineers are facing an increasing number of challenges as they attempt to develop new products that are faster, having high quality and are rich in features. The trend today is defining more requirements, but developers often struggle when the requirements are added or changed during the product life cycle. Despite of advances in software engineering over the past 30 years, most software projects still experience numerous requirements changes during their life cycle, which is brought about by the dynamic nature of development activities [15].

This paper discusses the work done in the area of requirements volatility, and tries to identify the objectives of the research work to be done.

Keyword: Requirements, Requirements Volatility

1. Introduction

Despite of advances in software engineering over the past 30 years, most software projects still experience numerous requirements changes during their life cycle, which is brought about by the dynamic nature of development activities [1]. Requirements volatility, which may be defined as the change in requirements (in terms of the number of additions, deletions, and modifications) during software project development, has been reported as one of the main factors causing a project to experience challenges [30].The literature has pointed out to the causes and effects of requirement volatility, and also outlined some of the approaches in order to deal with the problem.

Requirements are the foundation of the software development providing basis for project estimation and project planning. The success of a software project is also influenced by the quality of the requirements. Although the initial sets of requirements are well documented, changes will occur during the software development lifecycle. Requirement volatility which refers to such changes in requirements during the software development life cycle has been reported as one of the main factors that cause a project to be challenged [31].

2. Motivation and Present State of Research

This paper covers a literature review of prior research relevant to Requirement Volatility. Though, lot of work has been done on Requirement Management, and many requirement management tools have been developed and are being used by the industry, not much has been done to minimize the impact of requirement volatility on software development lifecycle.

In order to find the related work done in the area of requirement volatility, in correlation with requirement management and impact of volatility etc, we explored the databases of digital libraries, such as:

- IEEE,
- ACM,
- Science Direct,
- Springer-Links,
- Google scholar search engine, and
- DBLP Computer Science Bibliography.

Keywords used to search through the databases (we combined them with AND and OR to make more extensive searches):

- Requirement,
- Stability/instability/volatility,
- Metrics,
- Measurement,
- Management, and
- Failure/success.

In total, we found 35 papers that we had to filter in order to find the most relevant ones to our study. To achieve this, we read the abstract, introduction and conclusion parts of the papers. When a paper was found relevant, we read it completely to investigate it more. The overall goal

of the literature study was to increase our knowledge in the area of requirement volatility, searching for related work, finding out whether any research has been done in the area selected by us.

The result of the literature review revealed that although various studies have been done on requirement management area, we found no research specifically focuses on investigating whether impact of requirement volatility can be minimized on software development lifecycle.

In 2002, Didar Zowghi, N Nurmuliani, performed "**A Study of the Impact of Requirements Volatility on Software Project Performance"[3].** This paper describes their findings of an extensive survey based empirical study of requirement volatility (RV) and its impact on software project performance. In particular, findings reveal that requirement volatility has a significant impact on schedule overrun and cost overrun in software projects. There investigation also examined factors that contribute to the extent of requirement volatility and found that variables such as frequent communications between users and developers and usage of a definable methodology in requirements analysis and modeling have impact on the stability of requirements. **The findings indicate that there is a negative relationship between requirements volatility and software project performance, measured by project completing on time and on budget. There is a clear indication that the more unstable requirements become the more likely it is that the project will be completed behind schedule and over budget**. They contend that requirements volatility is not the only factor that would affect the project delays or project cost overruns. We felt that additional investigation was required to examine the influence of other factors. They tested the impact of factors (as control variables) such as organization size (i.e. business turnover) and project size (i.e. project cost, total effort, and number of user) and added these control variables into the logistic model. However, the regression analysis suggested none of the control variables were significant.

Impact of several RE activities and related issues of requirements practice on RV was also examined. The findings indicate that using a definable methodology for requirements analysis and modeling has negative impact on RV. Another activity of RE process that the authors investigated was the impact of performing requirements inspection on RV. The findings indicate that performing requirements inspection reduces the extent of RV. Formal requirements inspection are claimed to be a cost-effective technique to discover requirements defects. However, because inspection needs people from different skills and organizations to get together, it is reported that most project teams do not carry our formal inspection of requirements before design begins. We believe that this is an important finding for motivating practitioners to spend adequate time for validating requirements.

Frequency of communication between developer and customers was yet another issue that we investigated in relation to its impact on RV. The findings suggest that the more frequent developers and customers communicate with each other during RE, the less volatile their requirements will be. A related issue under investigation was to do with the number of user representatives involved in the development team. The findings suggest that the more user representatives involved in the development team the more volatile the requirements were. These two contributing factors seem to be competing and we feel that it requires further investigation. The choice of whether to have more meetings with the stakeholders versus involving a number of users in the development team is not an easy one to make and we leave it for future work to investigate this issue possibly with case studies.

Finally the other two related factors we examined were the impact of using requirements management tools and established traceability. The findings did not suggest any significant impact on requirements volatility.

In 2002, Juha Savolainer et al in their paper "Volatility Analysis Framework for Product Lines"[4], developed volatility analysis framework for product lines. While discussing about volatility in requirements, they mentioned that there are two major aspects of volatility: **the probability of a change in requirement in future, and the vagueness of the requirements.**

In 2002, Dr. Andrew Vickers in their paper "Requirements Engineering: How do you know how good you are?"[5], investigated the use of REVEAL, as a method to improve the way they take Requirement Engineering activities. REVEAL is based on self assessment, and is used to enable the organization to baseline the requirement engineering capabilities for each individual and for the organization as a whole.

In 2004, Nurmuliani N et al worked upon "Analysis of requirements volatility during software development lifecycle"[6]. In this study, case study approach was used, and findings revealed that the main cause of requirement volatility were changes in customer needs(or market demands), developer's increased understanding of products, and changes in the organization policy. It was also discovered that rate of volatility was high at the time of requirement specification completion.

In 2004, Alan M. Davis & Didar Zowghi, published their work on "Good requirements practices are neither necessary nor sufficient"[7]. Their study was to make sure that none of us (as requirements researchers or practitioners) get too overconfident in our quest to develop or apply good requirements practices. They mentioned "Do not discard a good requirements practice just because your project failed" and that "many practitioners adopt some ''good'' requirements practice only to discover that the resulting product failed, and the practice was subsequently abandoned".

Two more important points that came out of this research are:

- Long-term results are more important than short-term results, even though the short-term results are easier to measure. Do not fall victim to goals displacement. Do

not define a successful requirements practice as being one that provides only short-term results (e.g., one that increases the number of requirements agreed to per hour). This could motivate you to permanently adopt a practice that helps in the short-term but causes product failure in the long term.

- Requirements are but a small piece of a large whole. Applying good requirements practices will not guarantee you success, nor will applying bad requirements practices guarantee failure.

''The race is not always to the swift nor the battle to the strong, but that's the way to bet.'' (Damon Runyon)

In 2004, Talha Javed et al, performed a study "A Study to Investigate the Impact of Requirements Instability on Software Defects"[8]. The study indicates that there is a significant relationship between pre/post release change requests initiated by the client and software defects. In addition, study indicates that changes in the design of the system at the later stages of software development i.e., during coding, testing and after release have a significant impact on the high severity defects that affect the major functionality of the system. Also, it was found that insufficient time spent on the design phase and inadequate communication with the client could be some of the causes of requirements changes and consequently software defects.

In 2005, Lonconsole et al performed an "Industrial case study on Requirement Volatility measures"[9]. Goal of the study was to empirically validate set of measures associated with volatility of use case models(UCM). The goal of the study was twofold: 1) to empirically validate a set of measures associated with the volatility of use case models (UCM); 2) to investigate the correlation between subjective and objective volatility. Data analysis showed a high correlation between the authors measures of size of UCM and total number of changes, indicating that the measures of size of UCMs are good indicators of requirements volatility. No correlations were found between subjective and objective volatility.

In 2005, Donald G. Firesmith published a work on "Are Your Requirements Complete?"[10], in which he suggested some important characteristics in good requirements. According to him good requirements have several useful properties, such as being consistent, necessary, and unambiguous. Another essential characteristic that is almost always listed is that 'requirements should be complete.' But just what does completeness mean, and how should you ensure that your requirements are complete? He addressed these two questions by looking at (1) the importance of requirements completeness, (2) the completeness of requirements models, (3) the completeness of various types of individual requirements, , and (4) the completeness of requirements metadata. In next issue's column, we will continue by addressing (5) the completeness of requirements repositories, (6) the completeness of requirements documents derived from such repositories of requirements, (7) the completeness of sets of requirements documents, (8) the completeness of requirements baselines, and finally (9) determining how complete is complete enough when using an incremental and iterative development cycle.

He deduced that**, evaluating the completeness of requirements models, requirements, their metadata, and the repositories in which they are stored is not a trivial exercise. It is certainly worthy of more than a single checkbox on a simple requirements evaluation checklist.** Anyone responsible for evaluating these requirements work products would be well advised to use a more complete collection of potentially missing items. The contents of this column would be a good starting point for producing such a project or organizational set of information to look for.

Whereas incomplete individual requirements are a major problem, incomplete sets of requirements are an even greater problem. Although this column has provided much that can prevent the specification of incomplete individual requirements, we are not done with this topic. In next issue's column, we continue with our discussion of requirements completeness by addressing the completeness of the requirements repositories that are used to store both requirements and the requirements models from which they are derived, the completeness of requirements documents derived from such repositories of requirements, the completeness of sets of requirements documents, the completeness of requirements baselines, and determining how complete is complete enough when using an incremental and iterative development cycle.

In 2006, N Nurmuliani et al published their work on "**Requirements Volatility and Its Impact on Change Effort: Evidence-based Research in Software Development Projects"[11].** According to the paper, *the impact of RV is often underestimated; particularly the impact of small changes to requirements in the later stages of the development lifecycle. This paper presents evidence-based research on RV and its impact on software development effort. The findings are derived from two software project releases within a multi-site software organization. The paper discusses the usage of a requirements change classification to assist in identifying and assessing the extent of effort required to implement requirements changes. Research findings reveal that the rate of RV decreases toward the end of the project lifecycle. These empirical findings demonstrate that changing requirements, particularly adding new requirements, at the later phases is considered a high risk because it will cost the organization in terms of budget overruns or schedule delays.*

In this paper they have presented evidence-based research on requirements volatility and its associated costs from two software projects within a multi-site software development organization. This study has addressed the following research questions:

RQ1. What are the types of requirements change that require substantial effort?

The analysis suggests that adding new requirements at the later stages of development required substantial

effort. Hence, it is the later, very expensive changes that the organization should pay attention to. In addition to that change types, requirements changes due to *Defect fix*, *Product Strategy*, and *Missing requirements* are considered to be of the expensive change type.

RQ2. Which requirements change attributes contribute most to change effort?

The correlation coefficients indicate that all the change attributes identified in this study, i.e. *total number of requirements change*, *number of documents affected*, *sources of requirements change (internal and external)*, and *change types (addition, deletion, and, modification)*, are statistically significant contributors to the change effort.

The major contributions of this paper are twofold:
1) the usage of a requirements change classification to understand the cost of each requirements change, and
2) the relationship between estimated change effort and requirements change attributes.

This study is one of a number of longitudinal investigations currently being undertaken. For the first time, this paper provides empirical evidence about the cost of requirements volatility. The findings have provided valuable insights into the dynamic behavior of software requirements from the beginning of the systems development until the end of the project.

Future work will be undertaken to gather more requirements change data including change effort data (estimated and actual effort) to develop a requirements change effort estimation model.

In 2006, Borland published a white paper on "Effective Requirements Definition and Management"[12]. The Borland Solution delivers everything organizations need to successfully implement the five critical requirements definition and management processes into an organization for maximum impact, quickly and with a minimum of risk. The solution can be tailored to reflect an organization's maturity and goals, which is crucial to the ultimate success of any initiative.

The Borland Requirements Definition and Management Solution helps to ensure low-risk, rapid success at the lowest total cost by including "right sized" processes, customizable technology and tailored training packages to enable users and empower customers to proceed with confidence.

In 2007, Donald G. Firesmith, discussed "Common Requirements Problems, Their Negative Consequences, and Industry Best Practices to Help Solve Them"[13]. In this column, he summarized the 12 worst of the most common requirements engineering problems. He had observed these over many years working on and with real projects as a requirements engineer, consultant, trainer, and evaluator. He listed the negative consequences of these problems, and most importantly suggested some industry best practices that can help you avoid these problems, or at least fix them once they have raised their ugly heads. Although there is nothing really new here, these problems are well worth revisiting because they are still far too common, probably because the associated industry best practices are still far from being widely put into practice.

In 2007, a paper was published on "Impact of volatility & size on IT project performance"[14]. The study results were a complete reversal of reports from the Standish Group that approximately 67% of IT projects fail or are challenged, we have found 67% of projects are delivered close to budget, schedule, and scope expectations.

In the survey, Participants were selected from registered readers of Computer Weekly, a popular U.K.-based weekly newspaper for IT professionals. An initial email message and follow-up request to participate in the study were sent to readers registered as project managers on the Computer Weekly site. The survey data was collected using Web-based forms and undertaken between October 2002 and January 2003. They received 412 usable responses from project and program managers from companies of all sizes across all economic sectors. On average, the participants reported high levels of experience with 17 years in the IT industry and nine years as a project manager.

In 2008, Davis A.M. et al, investigated on "Requirements Change: What's the Alternative?"[15]. In this study the authors try to explore the effects of not making requirement changes in response to changes in user needs. The study indicates project incurs as much, if not more, risk when requirement changes are suppressed.

In 2009, Susan Ferreira et al published their work on "Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation"[16]. This paper introduces an executable system dynamics simulation model developed to help project managers comprehend the complex impacts related to requirements volatility on a software development project. The simulator extends previous research and adds research results from an empirical survey, including over 50 new parameters derived from the associated survey data, to a base model. The paper discusses detailed results from two cases that show significant cost, schedule, and quality impacts as a result of requirements volatility. The simulator can be used as an effective tool to demonstrate the complex set of factor relationships and effects related to requirements volatility.

In 2010, Muhammad Akram et al, did a work on "Qualitative And Quantitative Study For Requirement Change Management Model"[17]. According to the authors, r*equirements change management is a crucial activity for requirement engineers in industry during development of product in market driven context. Study shows that this implementation of requirements change is still problematic for requirement engineers. In this paper they have presented a research design that includes both qualitative and quantitative study analysis to find out different factors that can affect the performance of proposed model for requirement change management process. Experiments are used for quantitative study and*

three case studies are conducted in industries for qualitative analysis. It is concluded that the proposed model will be helpful to manage change in requirements during development life cycle of product in market driven environment. This model is anticipated to increase the performance of industry in market.

This report presents two approaches to analyze data, one is qualitative and other is a quantitative approach. They have used these approaches for validation of their proposed model. Main purpose of this model is to help requirement engineers to manage change in SRS. Experiments are based on data analysis. Their proposed model is expected to be helpful to manage change in requirements during development life cycle of product in market driven environment. This experiment will increase knowledge of subjects about quantitative research and change management in SRS. There are certain threats through which performance of experiment could be decreased. Requirement engineers should have some experience to handling requirements so that these threats could be decreased and performance of experiment can increases.

At last this study will help the researchers to investigate important aspects of requirement change management in industry. It describes the different aspects of case study and experiment which can help researchers to implement both qualitative and quantitative study in industry.

In 2010, Rahul Thakurta, Frederik Ahlemann, perofmed a study on "**Understanding Requirements Volatility in Software Projects – An Empirical Investigation of Volatility Awareness, Management Approaches and their Applicability"[1].** In this study, the authors present, *Requirements volatility during software project development is known to be the most critical risk, and managing this is paramount to success in software project. The research is based on a combination of interviews and a survey in two phases and aims to investigate the organizational practices in dealing with this risk, and how it is influenced by the adopted project execution strategy with regard to process model selection decisions. The results indicate that thirteen different approaches to managing projects under volatility could be identified. The study findings are expected to assist project managers in their choices related to project administration under requirements volatility.*

Findings are indicative of a shift in project manager's mindset towards requirement volatility with acceptation of the fact that requirement volatility is not an exception. Result advice for a need of caution in decisions regarding the selection of execution strategies in projects. The execution strategy choice was found to be influenced by the business in majority of the projects at risk under requirements volatility. In this regard, suggestions to train project stakeholders were put forward as they may not be aware of the potential hazards associated with volatility, and rather contribute to the cause.

The trend towards adoption of more "proactive" management strategies at higher maturity levels could be investigated further possibility leading to the establishment of a requirements volatility management framework on the dimensions "nature of management approach" and "project characteristics."

In 2010, Mauricio E. Peña, published a paper "**A study of the Causes of Requirements Volatility and its Impact on Systems Engineering Effort"[18].** In this, the author worked on two models: Research Question 1: What causes requirements volatility? and Research Question 2: How does requirements volatility impact systems engineering effort?. He proposed a casual model for further investigation on the research questions. His work is still in progress.

In 2011, Dhirendra Pandey et al, worked on "A Framework for Modelling Software Requirements"[19]. According to them, requirement engineering plays an important role in producing quality software products. In recent past years, some approaches of requirement framework have been designed to provide an end-to-end solution for system development life cycle. In this paper, they presented a requirement modelling framework with the analysis of modern requirements modelling techniques.

Also, they discussed various domains of requirement engineering with the help of modelling elements such as semantic map of business concepts, lifecycles of business objects, business processes, business rules, system context diagram, use cases and their scenarios, constraints, and user interface prototypes. The proposed framework was illustrated with the case study of inventory management system.

The paper discusses implementation of requirement modelling for various requirements analysis purposes and mapping of conventional requirements artifacts into system elements. We have also presented some modeling aspects, which are necessary for ensuring that the requirements elements that are mapped to the same UML element can be differentiated. We can also find critics on using UML as requirements specification language, most of the issues can be solved using UML tool with rich possibilities for modelling environment customization and extensions [18]. On the other hand, there are also suggestions to use more UML for requirements analysis and visualizations [20]. Multiple authors provide numerous papers on more detailed approaches to customizing unified modelling language for specific requirements modelling needs, such as analyzing scenarios, modelling user interface prototypes, refining requirements [21, 22]. Some researchers also suggest that UML can be specialized for early phase requirements gathering process but the proposed framework emphasizes that early phase modelling should focus on same types of artifacts with less detail.

3. Conclusion

The reviewed literature confirms that, it is a widely acknowledged fact that requirements volatility (RV) is inevitable and that it has a great impact on the software development lifecycle. As a major source of risk, the

impact of requirements volatility is often underestimated; particularly the impact of small changes to requirements in the later stages of the development lifecycle.

This paper discusses the work done in the area chosen by the author. The purpose of this paper is to review the related work done so that we can identify the objectives of the work to be done.

This review helped the authors to understand that though, lot of work has been done on Requirement Management, and many requirement management tools have been developed and are being used by the industry, not much has been done to minimize the impact of requirement volatility on software development lifecycle. Also, it helped in identification of research objectives of the research work.

4. References

[1] Thakurta, R.; F. Ahlemann; "Understanding Requirements Volatility in Software Projects—An Empirical Investigation of Volatility Awareness, Management Approaches and Their Applicability," 43rd Hawaii International Conference on System Sciences (HICSS 2010), USA, 2010.

2] R.Rajnish, "Requirements Volatility: A Risk in Software Development Process" published in the Conference proceedings of International Conference on "Emerging Technologies and Applications in Engineering, Technology and Sciences" (ICETAETS-2008),13th -14th Jan'2008.

[3] D. Zowghi and N. Nurmuliani, "A Study of the Impact of Requirements Volatility on Software Project Performance", *in the proceeding of the 9th Asia-PacificSoftware Engineering Conference*, Gold Coast, Australia, 2002.

[4] Juha Savolainen and Juha Kuusela, "Volatility Analysis Framework for Product Lines", http://delivery.acm.org/10.1145/380000/375277/p133-savolainen.pdf?ip=117.211.90.26&acc=ACTIVE%20SERVICE&CFID=51698261&CFTOKEN=94801133&__acm__=1320140967_3cc93fb7053266d54d1aa2ea74aaa667

[5] Dr. Andrew Vickers et al, "Requirements Engineering: How do you know how good you are?", proceedings of the 6th IEEE International Symposium on Requirements Engineering(RE'02), September 2002.

[6] N. Nurmuliani, D. Zowghi, and S. Fowell, "Analysis of Requirements Volatility during Software Development Life Cycle", ASWEC'04, Australia, 2004.

[7] Alan M. Davis, Didar Zowghi, "Good requirements practices are neither necessary nor sufficient", Springer-Verlag London Limited 2004.

[8] Talha Javed, "A Study to Investigate the Impact of Requirements Instability on Software Defects", ACM Software Engineering Notes, May 2004 Volume 29 Number 4.

[9] Lonconsole et al, "An Industrial Case Study on Requirements Volatility Measures", Software Engineering Conference, 2005. APSEC '05. 12th Asia-Pacific , 15-17 Dec. 2005, 8 pp., Print ISBN: 0-7695-2465-6

[10] Donald G. Firesmith: "Are Your Requirements Complete?", in Journal of Object Technology, vol. 4, no. 1, January-February 2005, pp. 27-43. http://www.jot.fm/issues/issue_2005_01/column3.

[11] N Nurmuliani etc, "Requirements Volatility and Its Impact on Change Effort: Evidence-based Research in Software Development Projects", AWRE 2006 Adelaide, Australia.

[12] "Effective Requirements Definition and Management", White paper, 2006, http://www.borland.com/resources/en/pdf/solutions/rdm_whitepaper.pdf.

[13] Donald G. Firesmith: "Common Requirements Problems, Their Negative Consequences, and Industry Best Practices to Help Solve Them ", in *Journal of Object Technolog y*, vol. 6, no. 1, January-February 2007, pp. 17-33 http://www.jot.fm/issues/issue_2007_01/column2.

[14] Chris Sauer et al, "The impact of size and volatility on IT project performance", Communications of The ACM November 2007/Vol. 50, No. 11.

[15] Alan M. Davis et al,"Requirements Change: What's the Alternative?", l IEEE International Computer Software and Applications Conference(COMPSAC'08), 2008, ISSN: 0730-3157, July 28 2008-Aug. 1 2008, pg 635 – 638.

[16] Susan Ferreira et al, "Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation", Journal of Systems and Software , Volume 82 Issue 10, October, 2009

[17] Muhammad Akram et al, "Qualitative And Quantitative Study For Requirement Change Management Model", www.iqraisb.edu.pk/icbte/Proceeding_ICBTE_**2010**/.../104.pdf

[18] Mauricio E. Peña, "A study of the Causes of Requirements Volatility and its Impact on Systems Engineering Effort", COSYSMO Workshop, March' 2010.

[19] Dhirendra Pandey et al, "A Framework for Modelling Software Requirements", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1, May 2011, ISSN (Online): 1694-0814

[20] Rahul Thakurta, Subhajit Dasgupta, "Using System Dynamics Modeling to Investigate the Impact of Resource Management Policies on Project Quality Assurance under Requirement Volatility", International Journal of Information Processing and Management Volume 2, Number 3, July 2011.

[21] G. Kotonya and I. Sommerville, Requirements Engineering Process & Techniques: John Wiley & Sons, 2002.

[22] K. Wiegers. Software Requirements. 2nd edition, Microsoft Press, 2005.

[23] I. Sommerville, Software Engineering, 7th ed. Harlow, England, New York: Pearson/Addison Wesley, 2004.

[24] K. E. Wiegers, Software Requirements: practical techniques for gathering and managing requirements throughout the product development lifecycle, 2nd ed. Washington: Microsoft Press, 2003.

[25] N. Nurmuliani, D. Zowghi, and S. Williams, "Characterising Requirements Volatility: An Industrial Case Study", *in the proceeding of International Symposium Empirical Software Engineering*, Noosa - Australia, 2005.

[26] D. Zowghi, R. Offen, and N. Nurmuliani, "The Impact of Requirements Volatility on Software Development Lifecycle", *in the proceeding of the International* Conference on Software, Theory and Practice *(ICS2000)*, Beijing, China, 2000.

[27] D. Pfahl and K. Lebsanft, "Using simulation to analyse the impact of software requirement volatility on project performance", Information and Software Technology, vol. 42, pp. 1001, 2000.

[28] M. Christel and K. Kang, "Issues in Requirements Elicitation", Carnegie Mellon University, Pittsburgh TR.CMU/SEI-92-TR-12, September 1992.

[29] C. Jones, "Strategies for Managing Requirements Creep", in Computer, vol. 29, 1996, pp. 92-94.

[30] M. Sudhakar. Managing the Impact of Requirements Volatility. Master Thesis. Department of Computing Science, Umeå University, Umeå, Sweden. 2005.

[31] A. Tiwana, and M. Keil, "The One Minute Risk Assessment Tool", Communications of the ACM, 2004.

[32] Frank Armour et al "A framework managing requirements volatility using function points as currency".

[33] Detmar W. Straub and Curtis L. Carlson "Validating Instruments in MIS Research", MIS Quarterly/June 1989.

[34] Victor R. Basili, "The Role of Experimentation in Software Engineering: Past, Current, and Future", 0270 - 5257'/96 $5.0001996 IEEE 442 Proceedings of ICSE-18.

[35] "Software process dynamics", by Raymond Joseph Madachy.

Dr Harsh Dev got his M.Sc. Degree in 1995 from Lucknow University & Ph.D. degree in Computer Science in 2005 from Babasaheb Bhimrao Ambedkar University, Lucknow, India. He worked with IISE(affiliated to Gautam Buddha technical University) for more than 9 years, where he was working as a Director for the last three years when left the Institute. He is currently a Professor in Dept. of Computer Science and Engineering, at Prranveer Singh Institute of technology, Kanpur, India. He has 17 years of teaching experience and 10 years of research experience in the field of Computer Graphics, Scientific computing & Software Engineering. He has published more than 15 International and National publications. He is a member of Computer Society of India, International Association of Computer Science and Information Technology (IACSIT) and various other bodies engaged in the field of research

Ranjana Awasthi got her M.C.A in 1994 & her M.Phil. Degree in Computer Science in 2010. She is a Research Scholar at Dept. of Computer Science & Engineering, Singhania University, Rajasthan, India. She has more than 18 years of teaching experience. Currently, she is actively engaged in the research work on the Requirement Volatility. He has produced several outstanding research publications.