Journal of International Technology and Information Management

Volume 20
Issue 1 *Double Issue 1/2*

Article 1

2011

A Mixed Mode Analysis of the Impact of Requirement Volatility on Software Project Success

Rahul Thakurta
Xavier Institute of Management

A Mixed Mode Analysis of the Impact of Requirement Volatility on Software Project Success

Rahul Thakurta
Xavier Institute of Management
INDIA

ABSTRACT

Requirement volatility has been identified as a significant risk factor behind software project success. This paper describes our findings of a 2-phase study comprising of interviews and surveys on the preparedness of organizations in managing requirement volatility and the resultant effect on project success and failure. Findings illuminate on the current level of awareness and management response to the problem of requirement volatility affecting software projects. The subjective treatment of project success/failure is brought out, and the association with requirement volatility is explored. Results are expected to lead to better governance mechanisms and improve project success rates under requirement volatility.

INTRODUCTION

Requirement volatility which refers to the change to requirements during the software development life cycle surfaces as a frequent and high impact risk in numerous empirical studies performed to identify risk factors, or to understand variables leading to a project's success or failure (Davis et al., 2008; Liu et al., 2008; Schmidt et al., 2001). The problem assumes a greater significance in the present context of increasing technical and business complexity, and hence management of requirements becomes a prerequisite behind successful project outcomes (Chen et al., 2004; Ferreira et al., 2009).

Here we investigate the organizational awareness of requirement volatility, different approaches used for managing and measuring volatility, and its overall association with project success. The study also brings out the multiple perspective regarding software project success and failure, factors instrumental behind success of endangered projects, and prominent reasons to failure. Process model selection factors and characteristics under unsatisfactory outcomes have also been indicated. A process model (synonymously known as systems development life cycle model) referred above relates to a conceptual model used in IT project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application.

This research project was carried out in two phases. An exploratory research design was employed in the 1st phase where requirement volatility and its association to project attributes and management techniques were examined in depth. Senior project managers associated with software development were interviewed for gathering insights on the problem. Content analysis of interview data revealed insights for further investigation. A web based survey utilized in the 2nd phase attempts to validate some of the observations of the 1st phase.

This article identifies the different research questions that emerges out of the findings of phases one and two, and addresses them in light of the available evidences in the survey data. Findings of the interviews have been reported simultaneously to strengthen our claim, and also to report contrasting viewpoints and arguments.

This article is organized as follows. A review of relevant literatures is provided next. The subsequent section lists the different research questions. It is then followed by sections on methodology, study outcome, and a discussion on the results. The concluding section summarizes the key findings.

LITERATURE REVIEW

Requirement volatility has surfaced as a common and frequent risk item in studies on software project risks (Boehm, 1991; Davis et al., 2008; Tiwana & Keil, 2004). Studies on requirement volatility have suggested metrics for measuring volatility, analyze factors behind its causes and effects on project development, and recommend possible ways to manage the problem.

Some of the proposed metrics of volatility include measuring it in terms of quantity of changes (Ambriola & Gervasi, 2000; Costello & Liu, 1995; MIL STD-198, 1994), timing of occurrence of the changes (Javed et al., 2004) and propensity of change based on project characteristics (Costello and Liu, 1995). Causes behind requirement volatility have been traced to external factors like government and market as well as internal factors involving project organization and stakeholders (Ebert & Man, 2005; Kontonya & Sommerville, 2002; Nurmuliani et al., 2004). The effect of requirement volatility on cost, schedule, productivity, defect generation (Ferreira et al., 2009; Javed et al., 2004; Malaiya & Denton, 1998; Zowghi & Nurmuliani, 2002) has also been investigated. The various suggested approaches for managing volatility include adoption of incremental frameworks (Boehm, 1991; Nindel-Edwards & Steinke, 2005), use of joint application design (JAD), prototyping and configuration management (Jones, 1998), baselining requirements (Wiegers, 1999) and formation of change control boards (Vliet, 2008). The importance of use of contextual change management techniques has been highlighted in Ebert and Man (2005), but has not been investigated upon.

Research has analyzed concepts related to software project success and failure, their measurements, and drivers. Software project success has been differentiated from software product success (Westhuizen and Fitzgerald, 2005) and mostly defined in terms of meeting budget, delivery, and business objectives ("The Standish Group", 1994). Kerzner (1995) defines project success as completed within time, within budget, within scope, meeting performance requirements, and obtaining user acceptance. According to Lewis (2001), project success could be defined as meeting performance requirements, cost requirements, time restrictions, and project scope. Three additional dimensions of success in terms of impact to the customer, benefits to the performing organization, and scope of future benefit have been incorporated in Shenhar et al. (2001). Accounting for difference in perspectives among different project stakeholders has been highlighted in Linberg (1999). Measuring project success has been often carried out in dimensions stated above (Emam and Koru, 2008; Globerson and Zwikael, 2002; Procaccino et al., 2005), and also by the quality of project management process and satisfaction of stakeholder expectations (Schwalbe, 2004). There are instances, however, where projects that meet all of these factors are not necessarily viewed as successful (Westhuizen and Fitzgerald, 2005)**,** however the end product could still be a success. On the other hand, there are projects that do not meet above criteria, but that are considered successful nonetheless (Lewis , 2001). The drivers behind software success have also been listed down (Berntsson-Svensson & Aurum, 2006; Curtis et al., 1988; Jiang et al., 1996; Verner et al., 2007). Project failures have been defined as a construct opposite to that of success, and various reasons of failure have been indicated and prioritized (Cerpa & Verner, 2009; Emam & Koru, 2008; Shenhar et al., 2001).

The above studies indicate that software project success and failure are largely multidimensional constructs having wide interpretability depending upon the situation. Studies that have addressed requirement volatility in the context of software success have largely ignored measuring the degree of success, or accounting for its underlying constructs. This research intends to bridge this gap and explore the linkage among these aspects, and how they influence process choices and management procedures.

RESEARCH QUESTIONS

Our research looks to investigate the organizational perception of requirement volatility, the different metrics that are used to measure volatility, the impact of volatility on project success and the available competencies in managing the problem. The following questions have been addressed:

- What is the level of understanding among projects managers concerning requirement volatility?
- How much risk is attributed to the problem of requirement volatility affecting software projects?
- How (if at all) have the project managers attempted to measure requirement volatility?
- What fraction of them has tried to proactively manage volatility in their projects?
- Do the approaches differ based on the contract type of the project?
- How is the term “software project success” defined and measured among industry practitioners?
- What factors were instrumental behind success of endangered projects?
- How the term “software project failure” stands out in contrast to the previous definition of software project success?
- What are the prominent reasons for which the software projects fail?
- What kind of impact do various scenarios have on the outcome of projects endangered because of requirement volatility?

METHODOLOGY

Research Approach

The study has been structured as follows:

<u>Interviews</u>

In the 1st phase of the study extending from November, 2007 to March, 2008, in-depth qualitative interviews were carried out with eleven senior software project managers belonging to various organizations within Germany. The interviews sought to capture individual experiences, opinions, perceptions and knowledge regarding preparedness of organization regarding the problem of requirement volatility affecting software projects.

An interview guide approach was used to carry out the interviews. The guide contained questions on the interviewees' demographics, organizational settings, project background information, systems development life cycle methodologies, software project success, awareness and criticality of requirement volatility, requirement change pattern and degree and effect of requirement volatility on project. Greater flexibility was offered to the respondents to express whatever they wanted in any particular order they wished. All the interviews were carried out over telephone and recorded. Information provided by the respondents was guaranteed to be anonymous, and organizational details were kept in confidence. Interviews normally lasted between one and one-half hours.

Several Ph.D. students tested the interview guide for correctness so that it complies with the guidelines prescribed in Maykut (1994). The test provided feedbacks on the overall layout of the questions and helped to clarify and sharpen some questions. The interview guide was finally pre-tested with an IS project manager.

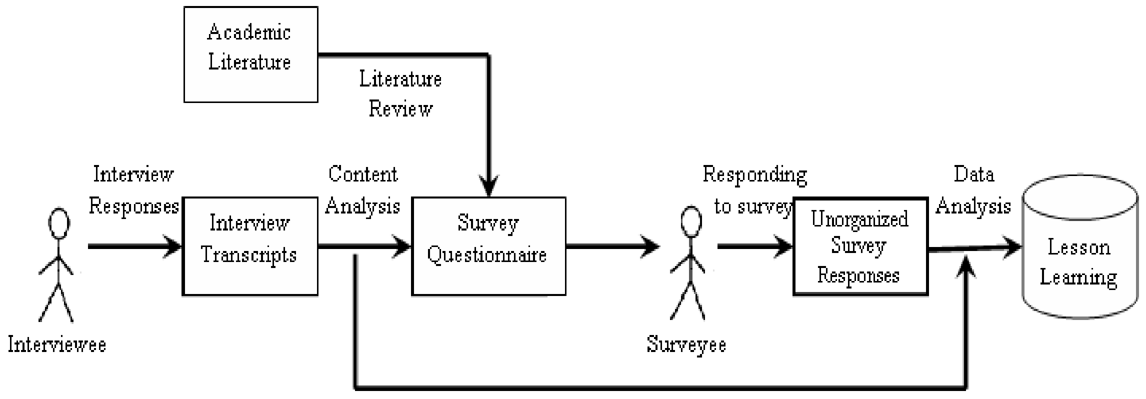
Interview notes were analyzed inductively by the constant comparative method (Glaser and Strauss, 1967). The responses were codified, with each code representing a theme or idea with which each part of the data was associated. New themes were assigned to data that fell outside the possible alternatives. The codes that had common elements were merged to form categories (Patton, 2001), which were subsequently clustered for pattern identification.

Survey

A web-based survey was carried out as a part of phase-2 of the study with an objective to test out the phase-1 observations on a much wider sample. The survey contained five sections, appraising the respondents on the purpose of the study, enquiring about demographic information, their association with software projects and take on requirement volatility. One section was devoted to project specific data corresponding to a recently completed software project that had problems with requirement volatility. This was done to ensure the survey results more reliable than the software practitioners' mere opinions and generalizations.

The questionnaire was pre-tested by several software developers to check for ambiguity. Data analysis was carried out using the SPSS statistical package whenever appropriate. The basic assumptions required for the different tests were checked in advance. Where ever the data did not meet the test criteria, patterns are reported for further analysis.

Figure 1: Research Approach.



Measurement of Constructs

The constructs of this research include awareness and risk of requirement volatility, process maturity, project contract types, project application categories, process models, and software project success and failure.

The first construct attempts to capture the level of awareness among project managers concerning the problem of requirement volatility. The level of awareness has been measured using a 1-5 scale, the two extremes indicating high awareness of the problem and not aware of the problem respectively.

The risk of requirement volatility has been captured in our study using a 5-point risk perception scale with five denoting the highest degree of risk.

The process maturity framework of CMMi (Capability Maturity Model Integrated) was chosen for our study. It defines the following five maturity levels:

Table 1. Process Maturity Level

Maturity Level (L)	Description
L-1 (Initial)	The software process within the organization is ad-hoc and chaotic with ineffective management procedures and project plans.
L-2 (Repeatable)	The organization can successfully repeat projects of the same type. However projects success depends more on individual managers and organization folklore acting as a process description.
L-3 (Defined)	The organization has a defined process with formal procedures to ensure the application to all software projects.
L-4 (Managed)	In addition to the above activity, the organization has a formal program to collect quantitative process and product metrics and analyze and use these for process improvement activities.
L-5 (Optimizing)	In addition to the above activity, the organization demonstrates its commitment to continuous process improvement.

The different categories of project contract were chosen as given in Lewin (2001) as follows:

Table 2: Project contract type.

Contract Type	Description
Fixed Price (or 'Firm Fixed Price')	The owner (business) specifies the work and the contractor (software development organization) gives a price. In this case the contractor assumes almost all of the risk and as a result reaps whatever profit there is.
Fixed Schedule	Here the project development schedule is non-negotiable and decided beforehand. Used for small and mission critical applications.
Time and Materials (T&M)	Simple billing at pre-negotiated rates for labor and materials on a project. Some "Fixed Price" contracts specify this as a method for determining costs of change orders.
Cost Plus Fixed Fee (Also called 'Cost Plus')	This type of contract shifts most of the risk to the owner (business), but also allows the owner a high degree of flexibility. The contractor (software development organization) has profit at risk and will seek to minimize cost/duration to return a higher proportional profit margin.
Cost Plus Percentage of Costs (CPPC)	This is very similar to the 'Cost Plus Fixed Fee' contract except that the contractor (software development organization) bears even less risk. Their fee is calculated based on a percentage of actual costs.

The different application categories include commercial, MIS, systems, military, contract or outsourced products, described by Jones (1999) as follows:

Table 3. Application Category.

Application Category	Description
Commercial software	Software that has been designed and developed for sale to the general public.
MIS software	Common business software developed by internal IT departments of organizations for internal use.
Contract or outsourced product (COP)	Software developed by contractor for an organization's internal use.
System software	Code that controls physical devices such as computers or telecommunication systems, and also includes operating systems, databases and middle ware.
Military software	Software written for defense use are written after following

department of standards.

The different process models are classified as given in Table 4.

Table 4: Process model.

Process Model	Definition
Waterfall Model	It is a software development model (with strictly one Iteration/phase) in which development proceeds sequentially through the phases: requirements analysis, design, coding, testing (validation), integration, and maintenance
V-Shaped Model	This is an extension of the waterfall model but instead of moving down in a linear way, the process steps bent upwards after the coding phase in a typical V shape.
Prototyping Model	It is a software development process that begins with requirements collection, followed by prototyping and user evaluation
Incremental-Iterative Model	Here the software project is divided into mini-projects, each of which is an iteration that results in an increment. Each iteration represents a mini-waterfall model.
Spiral Model	This supposes incremental development, using the waterfall model for each step, with more emphasis on managing risk.
Rapid Application Development (RAD)	It is a software development process that allows usable systems to be built in as little as 60-90 days, often with some compromises.
Agile Methodologies	Agile is an evolutionary approach to software development which is performed in a highly collaborative manner by self-organizing teams with the objective of producing high quality software in a cost effective and timely manner. Some of the different Agile Approaches are Extreme Programming (XP), Scrum, Pair Programming, etc.

Our study also looks at the different ways of characterizing software project success and failure, and their associated measures. Since perceptions of success and failure are subjective and situational (Shenhar et al., 2001), we chose to describe them by classifying the responses obtained during the interviews into appropriate categories as identified in the literature. The following section on "results" describes these constructs.

Sample Description

Experienced software professionals having at least five years of expertise or worked as project leads, managers or equivalent were targeted in the survey to minimize guess responses. A simple random sampling strategy was adopted and invitations to participate in the survey were mailed to members of some of the IS mailing lists available online. Follow-up invitations were emailed twice in gaps of two weeks. The survey was made available online for two months. As multiple mailing lists were targeted and these lists are dynamic, it is unknown how many potential respondents actually saw the survey invitation. An access counter on the survey page indicated a total of 176 respondents to visit the survey page, out of which 112 (64%) individuals finally completed it. Of the completed questionnaires, some were out-of-sample responses and problem responses (like multiple response), which had to be discarded. The final usable sample size came out to be 82 (47%). Factors like number of questions (44), depth of information sought, unfamiliarity of the area, amount of time required might have been instrumental behind the low completion rate of the survey. Most of the respondents were male (73%). 51% of the respondents had over ten years of software project experience (Figure 2). 62% of the respondents had been involved in more than ten projects (Figure 3). The breakup of industry sector is provided in Figure 4. Most of the organizations (36%) belonged to the technology sector indicating that the study results to be more relevant for this category. 62% of the organizations represented by the respondents were large ones with more than 1000 employees (Figure 5).

The survey respondents reported being involved in 1470 projects since 2003. More than half of these projects (54.9%) were considered at risk due to requirement volatility. About 68% of these projects were considered to be successful at the end. However the data did not indicate of the projects which failed; how many failed because of requirement volatility happening during project development.

Figure 2: Respondents' Years of Experience

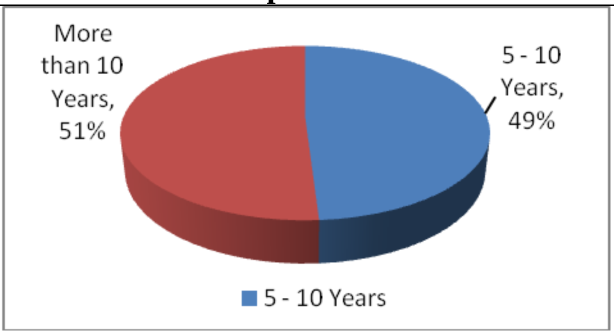


Figure 3: Experience in Number of Projects

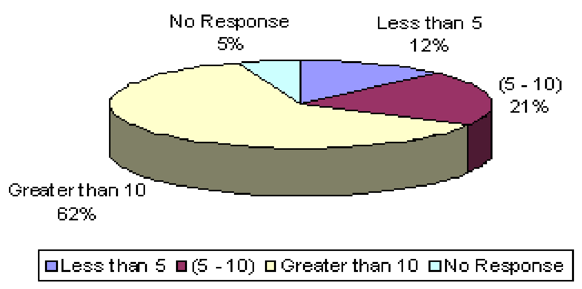


Figure 4. Respondents' Industry Sector

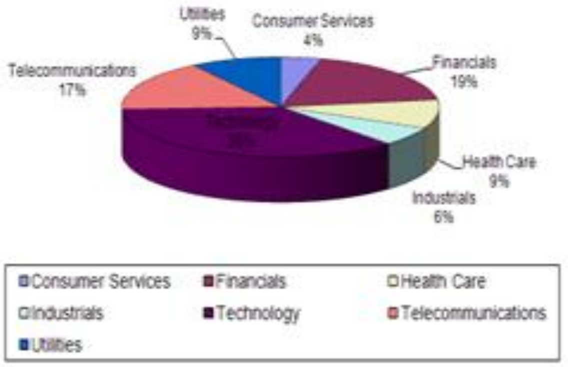
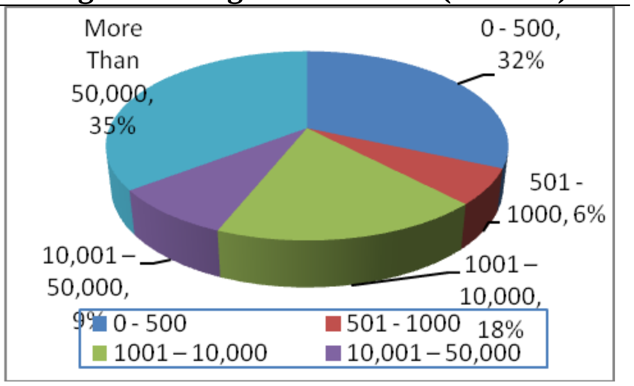


Figure 5. Organization Size (Persons)



The respondents were also asked to report on a recently completed project that was considered at risk due to requirement volatility. Data were obtained on 42 such projects. 32.6% of these projects represented organizations that did not use any of the maturity ratings. Of the others, level-5 organizations were found to be the most represented (32.6%), followed by level-3 ones (11.6%) as shown in Figure 6. 44.2% of these projects used the "Fixed Price" contract to negotiate with their clients. "Time and Materials (T&M)" contract was used in 39.5%, and "Cost Plus" in 11.6% of the cases. Surprisingly use of "Cost Plus Percentage of Costs" contract was not reported in the sample (Figure 7). The respondents reported being mostly associated with MIS applications (41.9%). Commercial applications and systems applications were equally represented in the sample (23.3%) (Figure 8).

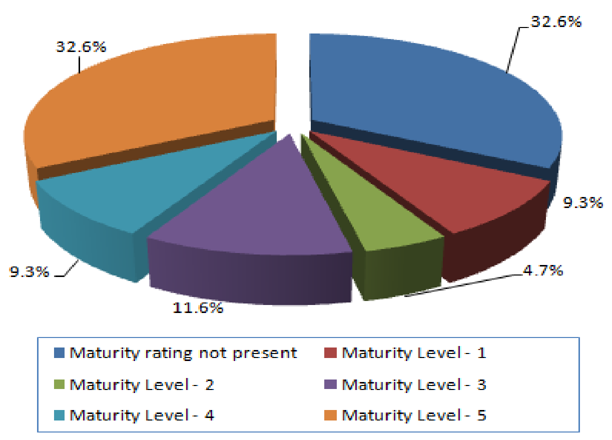


Figure 6. Distribution of Process Maturity Levels

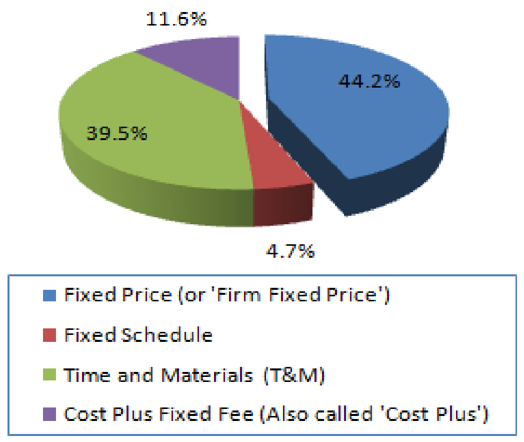


Figure 7. Distribution of Project Contracts

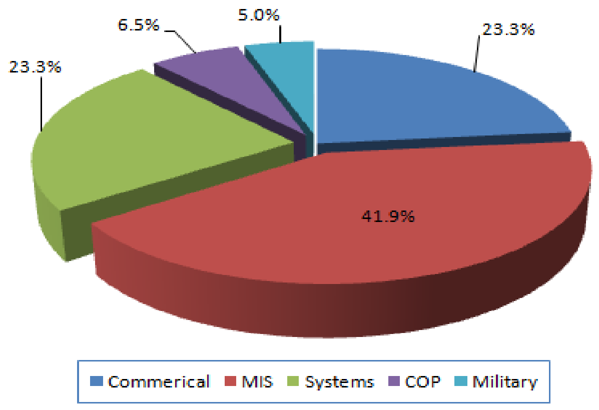


Figure 8. Breakup of Application Categories

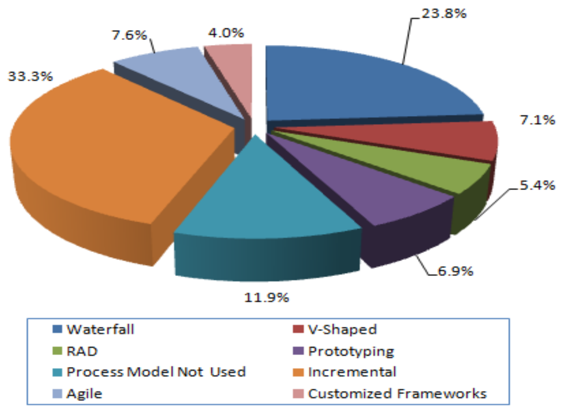


Figure 9. Process Model Usage

11.9% of the 42 endangered projects were found not to use any of the available process models. Of the rest, in contrast to the above data, Iterative-Incremental model was found to be the highest used (33.3%). Waterfall model was utilized in 23.8% of the projects. Other process models like V-Shaped Model, Prototyping Model, Rapid Application Development, and Agile Methodologies were nearly equally represented in the sample (Figure 9). Finally with regard to final outcome, 36.4% of these 42 projects were considered to be successful. 54.5% of projects were regarded as partial failures (defined by the interviewees as a situation where the vendor lost money); the rest (9.1%) failed completely.

Validation of Research Methods

The survey instrument was validated using Straub's (1989) guidelines. Pre-testing was utilized to improve the reliability of our questionnaire. Three of our interviewees also completed the survey questionnaire. Comparison of their data enabled to evaluate the construct validity of the questionnaire. The interview results also helped us to form survey questions and interpret the answers of the respondents.

Our sample represented a broad range of IS project types across a variety of industries and spanning across small to large organizations. The extensive representation of projects and organizations should reduce concerns of bias in the sample. Non response bias which poses a serious threat to the validity of the results was tested by comparing early (those received on first invitation) and late (those received after the 2nd follow up) respondents (Armstrong and Overton, 1977). Results revealed absence of any significant difference along key sample and project characteristics (α=0.05)

However, the results of the research should be interpreted with some caution. There are chances of observation and information bias during the interviews due to the involvement of a single observer. The results may be biased because of recollection error as some respondents reported association and details of projects executed long back. The evaluation of project success and failure were based on participants own perceptions, and were not cross checked with user viewpoints. Project specific responses were also not validated against available data or views of co-project members.

Our sample is relatively small and may not be representative of all development projects and organizations. Thus the survey and interview do not provide sufficient coverage of all situations. Some of the findings were also not statistically verified because of insufficient data points. However the focus of our work is more of understanding of the phenomena as experienced in organizations, and the resultant impact on project success. Patterns uncovered in the research are early insights, and is expected to provide basis for further work in this area.

RESULTS

What is the level of understanding among projects managers concerning requirement volatility?

Interviews revealed a high level of awareness of the problem posed by requirement volatility on software projects. 81% of the project managers were found highly aware of the problem, the rest considered themselves moderately/decently aware.

How much risk is attributed to the problem of requirement volatility affecting software projects?

Survey participants were asked if inspite of all the methodological advancements, requirement volatility is still perceived as a significant threat to software projects. We used the descriptor "significant" to imply a rating of four or five on a 5-point scale. 72% of the responses (N: 82) adhered to this group, indicating a heightened perception of risk attributed to requirement volatility among the study respondents.

How (if at all) have the project managers attempted to measure requirement volatility?

We wanted to know whether the project managers have attempted to measure requirement volatility, and the metrics used for the purpose. Just over half (54.5%) of the interviewees were found to have attempted in measuring volatility. With respect to measuring volatility, apart from the metrics suggested in the literature, the followings were also found used across organizations:

- Number of Person-days of change you have to throw away or modify
- Number of alterations of the identified use cases
- Number of times each requirement has changed
- Number of changing requirements identified within the issued change requests
- Measure of realized requirements out of total requirements specified by the business
- Amount of budget the project had to spent on the changing requirements

What fraction of them has tried to proactively manage volatility in their projects?

We tried to figure out if the project managers proactively attempted to manage change in projects threatened because of requirement volatility. About 94% (N: 16) of the project managers who responded confirmed such proactive behavior. This prompts us to believe that most (not all) of the project managers are appreciative of the problem of requirement volatility and resort to proactive measures whenever necessary.

The following 15 approaches (based on 82 responses) were identified as used across organizations for managing projects under volatility:

Table 5: Management approaches under requirement volatility.

Sr No.	List of Approaches
1	Involving business side into the project
2	Using iterative/phased project development approaches
3	Reducing project complexity
4	Project scope negotiation
5	Engaging in requirements management activities
6	Documentation of processes, procedures and activities
7	Adjusting project human resource
8	Using expert knowledge
9	Focusing on communications
10	Rescheduling project deadline
11	Readjusting project effort
12	Variable costing of additional requirements
13	Architecting product to withstand change
14	Training workforce
15	Adopting agile processes

The top three frequently used approaches emerged as "involving business side" (11.3%), "using iterative/phased approach" (10.2%) and "project scope negotiation" (9.8%).

Do the approaches differ based on the contract type of the project?

We wanted to find out if the type of project contract affects the choice of project management approaches under requirement volatility. A classification scheme based on observed frequencies was adopted (Berntsson-Svensson & Aurum, 2006). Logistic Regression techniques which could have been appropriate for this type of analysis was not used as the sample size requirement was not met (Meyers et al., 2005). The comparison was carried out between the two most frequently used contracts i.e. "Fixed Price" and "Time and Material"

Data based on 42 responses however failed to detect any significant difference in pattern among the available requirement volatility management approaches. Apart from the top three listed approaches, "rescheduling project deadline", "engaging in requirements management approaches" and "adjusting human resource" approaches were also observed to be frequent under both the contracts. The practice of "readjusting project effort" was found to be more for "Fixed-Price" contracts. "Agile practices" was used for the "Time and material" projects.

How is the term software project success defined and measured among industry practitioners?

Interviewees were asked on how they wish to define software project success. The responses categorized inductively revealed the different dimensions of software project success as given below (the last column provides an indication of which of these dimensions were more preferred by the respondents).

Table 6: Different dimensions of software project success.

Sr No.	List of Approaches	Preferred Dimensions
1	… has satisfied business objectives	46.3%
2	… has met budget requirements	2.4%
3	… has met schedule requirements	2.4%
4	… has met quality requirements	1.2%
5	… has met functional requirements	7.3%
6	… has met technical requirements	1.2%
7	… has met user's expectations	12.2%
8	… has resulted in stakeholder satisfaction	7.3%
9	… has resulted in a successful product	4.9%
10	… has all the project risks and dependencies under control	3.7%
11	… has successfully managed all the changes	1.2%
12	… creates value to the vendor organization	9.8%

The results indicate the top three choices to be "satisfaction of business objectives", "meeting user expectations", and "creating value to the vendor organization". The interviews also revealed the preferred ways of measuring software project success. The following measures were found to be widely employed:

- Measuring project's stakeholders satisfaction (through the use of questionnaires, review sessions, informal feedbacks)
- The project delivered as per the delivery schedule
- Project's budget conditions fulfilled
- Measuring project quality

What factors were instrumental behind success of endangered projects?

Despite being troubled because of requirements changes, a large percentage (90.9%) of projects was ultimately successful (partial or complete). Support of business was considered prime responsible behind achieving success under the scenario. Other significant factors were strong backing of management, presence of competent project team and skillful resource management (both technical and human).

How the term "software project failure" stands out in contrast to the previous definition of software project success?

We also enquired about the practitioner's viewpoint regarding software project failure. Again the preferred notions were: going out of budget, criteria of time not met, quality target not reached, and acceptance test criteria not met. There were diverse viewpoints regarding the permissible limit to budget overrun (ranging from 3-4% to 100%) beyond which a project can be regarded as a failure. Similar variations were also observed with regard to schedule overrun. While a couple felt that exceeding schedule by 40% is ok, one interviewee remarked *"my project is a failure if it offshoots schedule by more than 10%"* The target of quality was specified as a deviation from the original requirements by maximum 15%

What are the prominent reasons for which the software projects fail?

Reasons to project failure could be identified. Most interviewees' concurred that inability to manage project scope was the prime contributor. Others pointed out to increasing project complexity as leading to failure. In addition, participants mentioned about inadequate process maturity indicating absence of *"intensive working models"* and *"lack of development guidelines"* to be equally responsible. Problems with business, inaccurate upfront estimation, and market competition were also catalytic behind some downfalls.

What kind of impact do various scenarios have on the outcome of projects endangered because of requirement volatility?

We attempted to find out if usage of any specific project contract has led to more successful endeavors. The classification was carried out based on observed frequencies across the different project success categories (Berntsson-Svensson and Aurum, 2006). Even though with both "Fixed Price" and "Time and Material" contracts there were similar success rates, all of the projects that failed used the former contract (N: 42) "Fixed Price" contract was observed to be *"inappropriate"* in situations of volatile requirements.

We also investigated existence of relationship between non-compliance of development process and project final outcome. We measured non-compliance as follows:

- Not using any of the available process models
- Not adhering to the steps as laid down in the process guidelines

Project's final outcome was measured using ordinal scale as success, partial failure and complete failure as described earlier. We used the logistic regression as the minimum sample size requirements, and assumptions related to multi-co linearity were satisfied (Meyers et al., 2005). Results based on 42 cases are tabulated in Table 7. The likelihood ratio test indicates the contribution of each variable to the model. No significant association at alpha = .05 could be observed between use of process model (variable: *SDLCused*) and the degree of project success (p-value: 0.291 > alpha). However non-adherence to process guidelines (variable: *SDLCcompliance*) affected project outcome (p-value: .002 < alpha). A test of spearman correlation among these two was also observed to be significant (coefficient value: 0.492).

The first half of Table 8 indicates projects adhering to the process guidelines (SDLCcompliance=1) compared to those not-adhering (SDLCcompliance=2) were less likely to be failures (Project was S/F/I=2), Odds Ratio (OR) = 0.126 (95% CI 0.023 to 0.702), p=0.018. That is we can say that projects not following the process model guidelines have more chances of being failures (OR =7.94, reciprocal of 0.126).

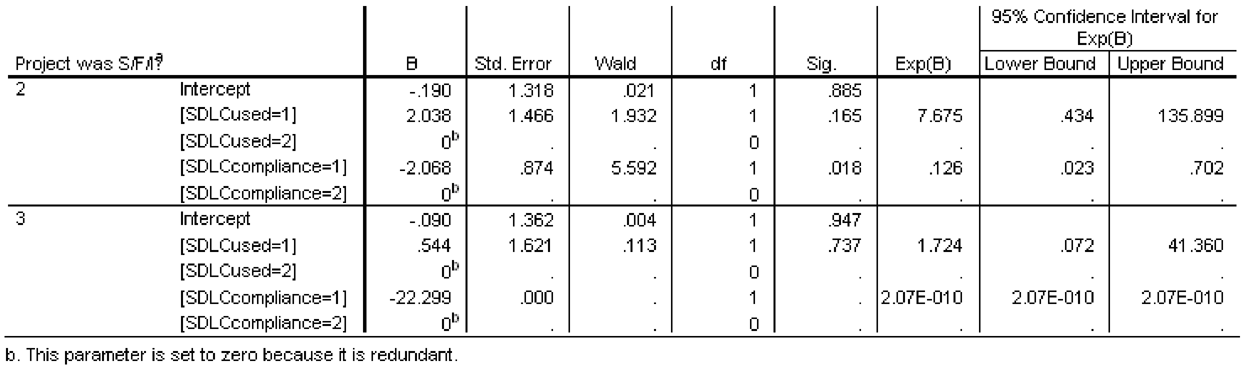
31.6% of the successful projects used the iterative-incremental process model, followed by the waterfall process model (26.3%). Use of agile methodologies was reported in 10.5% of projects.

Table 7: Likelihood ratio tests.

Effect	**Model Fitting Criteria**	**Likelihood Ratio Tests**		
	-2 Log Likelihood of Reduced Model	**Chi-Square**	**df**	**Sig.**
Intercept	11.956a	0.000	0	-
SDLCused	14.428	2.472	2	0.291
SDLCCompliance	24.663	12.707	2	0.002

The chi-square statistic is the difference in -2 log-likelihoods between the final model and a reduced model. The reduced model is formed by omitting an effect from the final model. The null hypothesis is that all parameters of that effect are 0.
This reduced model is equivalent to the final model because omitting the effect does not increase the degrees of freedom.

Table 8: Parameter estimates.



Project was S/F/I?[a]		B	Std. Error	Wald	df	Sig.	Exp(B)	95% Confidence Interval for Exp(B)	
								Lower Bound	Upper Bound
2	Intercept	-.190	1.318	.021	1	.885			
	[SDLCused=1]	2.038	1.466	1.932	1	.165	7.675	.434	135.899
	[SDLCused=2]	0[b]	.	.	0	.	.	.	.
	[SDLCcompliance=1]	-2.068	.874	5.592	1	.018	.126	.023	.702
	[SDLCcompliance=2]	0[b]	.	.	0	.	.	.	.
3	Intercept	-.090	1.362	.004	1	.947			
	[SDLCused=1]	.544	1.621	.113	1	.737	1.724	.072	41.360
	[SDLCused=2]	0[b]	.	.	0	.	.	.	.
	[SDLCcompliance=1]	-22.299	.000	.	1	.	2.07E-010	2.07E-010	2.07E-010
	[SDLCcompliance=2]	0[b]	.	.	0	.	.	.	.

b. This parameter is set to zero because it is redundant.

We asked the respondents if they felt the choice of process models they used in their respective projects were appropriate. Under successful outcomes, more than 93% considered the process model to be appropriate for the project. In contrast under situations were the project was a failure to some extent, only about 41% regarded the choice of the model to be appropriate. Probing further revealed the factors responsible behind selection of process models in relation to unsatisfactory project outcome (i.e. partial or complete failure). In situation where the respondents were not convinced (i.e. were indecisive or felt that the choice of the model was incorrect) about the appropriateness of the process models, the following 3 factors were found influential: management preferences (29.2%), influence of business/customers (20.8%), maturity of the overall development process (20.8%).

DISCUSSION

The following contributions can be drawn out of the study:

Some interesting viewpoints on requirement volatility were obtained. A couple of interviewees referred it in terms of the degree of requirements variance. Some preferred to define it in terms of requirements understandability (i.e. what the customer wants and what the vendor understands)

The perception of the threat posed requirement volatility could be captured in the study. Even though it was rated as a significant threat affecting over half of the projects surveyed, not all viewed it as a problem as one enquired *"Isn't it time we accepted requirement change as a part of our daily life and adapt accordingly?"* Other risk factors over which concerns were raised related to improper communication, attrition, project complexity and expectation management. Communication issues received the highest priority while attrition was viewed as a growing concern as one remarked "*retention of qualified personnel is becoming an alarming problem in today's IT"*

Some slackness in measuring requirement volatility was noticed with just over half of the respondents trying to measure it. Interestingly, even though a cited metric of requirement volatility is to measure it in terms of "number of additions, deletions and modifications of requirements", one interviewee strongly opined against it quoting *"I won't use this in my projects"*

Even though measuring volatility was not a common phenomenon, almost all the project managers were proactively engaged in managing volatility. Such a scenario could lead to solutions not adequately suited for the problem as one remarked "*We have tried several approaches under crunch situations hoping that some would click and steer our project to safety".* This in turn could also increase chances of project failure.

Different approaches for managing projects under requirement volatility could be identified. Even though project contract didn't significantly affect usage of approaches, the practice of "readjusting project effort" was more noticed for "Fixed-Price" contract. Agile practices which emerged to reduce development overheads and facilitate communication (Holcombe and Holcombe, 2008) was however found not suitable for this contract.

The study revealed the preferred notions of software project success among industry practitioners. Three of the top five factors were found to be related to business side, stressing the importance of business involvement in projects as amply highlighted in the literature (Ives and Olson, 1984). Some other aspects referred in relation to project success definition included *"high adaptability to change"*, and "*doing right things at right place at the time"*. The subjective connotation of "success" was brought out as one commented *"It depends on what 'success' means. They all resulted in software that is in use, adding value. But they all exceeded budget and cost estimates. Some of them brutalized the developers as the project approached delivery. Some were humane."*

Different measures of project success included measuring stakeholder satisfaction in terms of if the stakeholders are "*happy"* with the project. Here the importance of "sales team satisfaction" was also stressed. Some interviewees indicated delivery schedule adherence to be "number one priority for customer driven projects". Project quality was referred in terms of "number of bugs produced", and "delivering at least 80% of business specified requirements". Other adopted measures were in terms of "return on investment", "comparison of target with results", and "extent of adherence to acceptance tests"

Most participants viewed project failure to be a construct which is just opposite to their viewpoint of project success. This was reflected in their preferred notions listed above and in other choices like "customer losing money", "stakeholder dissatisfied and unhappy" and "stakeholder expectations not met". Descriptors like "*no roll out of the system*" were also used to qualify project failure

A little less than fifty percent of the projects that faced problems with changing requirements were found to be not adhering to the steps laid down in process model guidelines. This again correlated with unfavorable project outcomes, a finding also supported in the literature (Stepanek, 2005). Even though based on the total sample, waterfall model emerged as most used, the successful projects under this situation more resorted to the iterative-incremental model. When the project was viewed as a failure to some extent, in some instances though, the process model was considered fitting with "*management failures*" held responsible for the debacle. Mostly though, the respondents expressed their dis-satisfaction with the choice of process models under unfavourable project outcomes. In this scenario, the influence of business side emerged as a significant driver behind process model selection as interviewees observed *"It is not us that decide the life cycle model for our project. Our client has a specific process model and we have to follow the model even if it's not appropriate for the project"*. As the clients may not be aware of the problem posed by requirement volatility, these projects faced difficulties because of the changing requirements.

CONCLUSION

Software projects continue to be troubled because of requirement volatility. This paper presents the organizational awareness of requirement volatility, different approaches used for managing and measuring volatility, and its overall association with project success. The study also brings out the multiple perspective regarding software project success and failure, factors instrumental behind success of endangered projects, and prominent reasons to failure. The contribution of process models under unsatisfactory outcomes has also been indicated.

A high level of awareness of the problem of requirement volatility was noted. Even though volatility emerged as a significant risk, only about half of the respondents attempted to measure it. Different approaches to managing projects under volatility could be identified, among which "involving business side" and "Using Iterative project development approaches" emerged to be the most frequent. The top three preferred notions of software project success were found to be "satisfaction of business objectives", "meeting user expectations", and "creating value to the vendor organization". Measuring success through stakeholder's satisfaction was found to be widely employed. Respondents preferred to measure project failure in opposite terms to that of success. The common reasons of project failure were identified as improper management of project scope, increasing project complexity and inadequate process maturity. Unsuccessful project outcomes were found to correlate with non-adherence to model guidelines. In such scenarios, the process model choice was largely regarded as inappropriate.

The importance of our study was echoed by one respondent as *"Due to the environment I work in financial services and banking services, scope creep is an ever increasing problem that if not addressed immediately and appropriately can derail a project. Its one of the big challenges of a software engineering project has over other typical traditionalist engineering practices".* Follow up work could look to statistically validate the usage patterns of the different management approaches, establish statistical significance of patterns related to viewpoint, measure and factors of project success and failure, and investigate the influence of cultural factors on the overall context. A well rounded perspective of requirement volatility considering both the software project organization and the business users is also expected to leverage our overall understanding of the problem.

REFERENCES

Ambriola, V., & Gervasi, V. (2000). Process Metrics for Requirement Analysis. 7th European Workshop on Software Process Technology, Kaprun, Austria.

Armstrong J. S., & Overton, T. S. (1977). Estimating Nonresponse Bias in Mail Surveys. *Journal of Marketing Research 14(3).*

Berntsson-Svensson, R., & Aurum, A. (2006). Successful Software Project and Products: An Empirical Investigation. *Proceedings of the ISESE'06*, Rio de Janeiro, Brazil.

Boehm, B. (1991). Software Risk Management: Principles and Practices. *IEEE Software*.

Cerpa, N., & Verner, J.M. (2009). Why did your Project Fail? *Communications of the ACM 52(12)*, 130 – 134.

Chen, H.-G., Jiang, J. J., Chen, J.-C., Shim, J. T. (2004). The Impacts of Conflicts on Requirements Uncertainty and Project Performance. *Journal of International Technology and Information Management 13(3).*

Costello, R. J., & Liu, D. (1995). Metrics for Requirements Engineering. *Journal of Systems and Software*.

Curtis, B., Krasner, H., & Iscoe, N. (1988). A Field Study of the Software Design Process for Large Systems. *Communications of the ACM 31(11)*, 1268 – 1287.

Davis, A. M., Nurmuliani, N., Park, S., & Zowghi, D. (2008). Requirements Change: What's the Alternative? *Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference*, 635 – 638.

Ebert, C., & Man, J. (2005). Requirements Uncertainty: Influencing Factors and Concrete Improvements. *Proceedings of the ICSE.*

Emam, K. E., Koru, A., & Günes. (2008). A Replicated Survey of IT Software Project Failures. *IEEE Software 25(5)*, 84 – 90.

Ferreira, S., Collofello, J., Shunk, D., & Mackulak, G. (2009). Understanding the Effects of Requirements Volatility in Software Engineering by Using Analytical Modeling and Software Process Simulation. *Journal of Systems and Software 82(10)*, 1568 – 1577.

Glaser, B., & Strauss, A. (1967). The Discovery of Grounded Theory. Chicago: Aldine.

Globerson, S., & Zwikael, O. (2002). The Impact of the Project Manager on Project Management Planning Processes. *Project Management Journal 33(3)*, 58 – 64.

Holcombe, M., & Holcombe, W. M. L. (2008). Running an agile software development project. John Wiley and Sons.

Ives, B., & Olson, M. H. (1984). User lnvolvement and MIS Success: A Review of Research. *Management Science 30(5).*

Javed, T., Maqsood, M., & Durrani, Q. (2004). A Study to Investigate the Impact of Requirements Instability on Software Defects. *ACM SIGSOFT Software Engineering Notes Archive 29(3).*

Jiang, J. J., Klein, G., & Balloun, J. (1996). Ranking of System Implementation Success Factors. *Project Management Journal*, 49 – 53.

Jones, C. (1998). Estimating Software Costs. New York, NY: McGraw Hill.

Jones, C. (1999). Software Sizing. *IEEE Review,* 45(4), 165 – 167.

Kerzner, H. (1995). Project Management: A Systems Approach to Planning, Scheduling, and Controlling. Van Nostrand Reinhold, United States of America

Kontonya, G., & Sommerville, I. (2002). Requirements Engineering Process and Techniques. Wiley Publications Reprinted.

Lewin, M.D. (2001). Better Software Project Management: A Primer for Success. Wiley-1 Edition.

Lewis, J.P. (2001). Project Planning, Scheduling, and Control: A Hands-On Guide to Bringing Projects in On Time and On Budget. New York, NY: McGraw Hill.

Linberg, K.R. (1999). Software Developer Perceptions about Software Project Failure: A Case Study. *Journal of Systems and Software 49(2-3),* 177 – 192.

Liu, D., Wang, Q., Xiao, J., Li, J., & Li, H. (2008). RVSim: a Simulation Approach to Predict the Impact of Requirements Volatility on Software Project Plans. *Proceedings of the International Conference on Software Process,* Berlin, Heidelberg, 307 – 319.

Malaiya, Y., & Denton, J. (1998). Requirements Volatility and Defect Density. *Proceedings of the 10th International Symposium on Software Reliability Engineering*, Fort Collins.

Maykut, P. (1994). Beginning Qualitative Research: A Philosophical and Practical Guide. Routledge.

Meyers, L. S., Gamst, G. C., & Guarino, A. J. (2005). Applied Multivariate Research: Design and Interpretation. Sage Publications.

MIL STD-198. (1994). Software Development and Documentation. U.S. Department of Defense.

Nindel-Edwards, J., Steinke, G. (2005). The Development of a Thorough Test Plan in the Analysis Phase Leading to more Successful Software Development Projects. *Communications of the IIMA 5(1).*

Nurmuliani, N., Zowghi, D., & Fowell, S. (2004). Analysis of Requirements Volatility during Software Development Life Cycle. *Proceedings of the 2004 Australian Software Engineering Conference (ASWEC'04).*

Patton, M. Q. (2001). Qualitative Research & Evaluation Methods. Sage Publications, Third

Edition.

Procaccino, J.D., Verner, J.M., Shelfer, K.M., & Gefen, D. (2005). What do Software Practitioners really Think about Project Success: an Exploratory Study. *Journal of Systems and Software 78(2),* 194 – 203.

Schmidt, R., Lyytinen, K., Keil, M., & Culle, P. (2001). Identifying Software Project Risks: An International Delphi Study. *Journal of Management Information Systems*.

Schwalbe, K. (2004). Information Technology Project Management (Third Edition). Course Technology, Boston.

Shenhar, A.J., Dvir, D., Levy, O., & Maltz, A.C. (2001). Project Success: A Multidimensional Strategic Concept. *Journal of Long Range Planning 34.*

Stepanek, G. (2005). Software Project Secrets: Why Software Projects Fail (Expert's Voice). *Apress*.

Straub, D.W. (1989). Validating Instruments in MIS Research. *MIS Quarterly, 13(2)*.

The Standish Group (1994). CHAOS. Retrieved June 14, 2007, http://www.standishgroup.com/sample_research/chaos_1994_1.php

Thomsett, R. (2003). Project Pathology: Causes, Patterns and Symptoms of Project Failure. Retrieved November 1, 2007, from http://www.thomsett.com.au/main/articles/path/toc.htm

Tiwana, A., & Keil, M. (2004). The One Minute Risk Assessment Tool. *Communications of the ACM*.

Verner, J. M., Evanco, W.M., & Cerpa, N. (2007). State of the Practice: an Exploratory Analysis of Schedule Estimation and Software Project Success Prediction. *Information and Software Technology, 49(2),* 181 – 193.

Vliet, H. V. (2008). Software Engineering: Principles and Practice. Wiley.

Westhuizen, V. D., & Fitzgerald, E.P. (2005). Defining and Measuring Project Success. *Proceedings of the European Conference on IS Management, Leadership and Governance (ECMLG05),* Reading, UK.

Wiegers, K. (1999). Software Requirements. Washington: Microsoft Press.

Zowghi, D., & Nurmuliani, N. (2002). A Study on the Impact of Requirements Volatility on Software Project Performance. *Proceedings of Ninth Asia-Pacific SE Conference (APSEC' 2002)*.