INTERNATIONAL JOURNAL OF SOFT COMPUTING

MEDWELL PUBLICATIONS

Moderating Characteristics of Requirements Management Tools Between Factors of Rework and Project Success

[1]Faisal Adnan and [2]Imran Haider Naqvi
[1]*Department of Information Technology, University of Education, Lahore, Pakistan*
[2]*CIF, COMSATS Institute of Information Technology, Lahore, Pakistan*

Key words: Factors of rework, corrective rework, software requirements management tools, retrospective rework, evolutionary rework, project success

Corresponding Author:
Faisal Adnan
Department of Information Technology, University of Education, Lahore, Pakistan

Page No.: 123-131
Volume: 14, Issue 6, 2019
ISSN: 1816-9503
International Journal of Soft Computing

Abstract: This study quantified the underlying associations and critical moderating role of using software requirements management tools between factors of rework (including project planning, software requirements specifications document quality, software testing, maturity of software development life cycle approach, scope creep) and project success. The study contributed in determining the type of rework which could be avoided and how it could be avoided. The study quantified the magnitude of rework increased/reduced by factors of rework. The study dogged the underlying association of rework with project completion duration along with underlying associations of factors of rework and using software requirements management tools with rework and project success were also quantified. The study focused on using critical features of software requirements management tools to determine their role as an effective methodology for reducing the identified causes of unsuccessful software projects and rework to achieve project success.

INTRODUCTION

Well-structured Requirements Engineering (RE) process improved overall software productivity (Damian *et al*., 2005). While Project Success (PS) is ensured with RE which is a legitimate phase of Software Development Life Cycle (SDLC). RE process consists of Requirement Definition (RD) and Software Requirements Management (SRM) phases (Hennicker and Koch, 2000). RD phase leads to Software Requirements Specification document (SRS). SRM consists of software requirements documentation, Changing Requirements (CR) management and Software Requirements Traceability (SRT) (Gorschek, 2006). While software requirements consistency and completeness is associated with PS (Osmundson *et al*., 2003). Rework is caused due rapidly CR and its impact is measured on the basis of total number of artefacts affected, priority of change, number of resources required and SDLC phase which requires the change (Sharif *et al*., 2012). SRM controls CR based on SRS document which acts as an agreement of understanding between project stakeholders. Post-delivery maintenance work is traced down to poorly described SRS (Lang and Duggan, 2001). SRS contains functional details including what the system will do; system expected performance details, reliability and response time of software. SRS quality is critical for PS (Bokhari and Siddiqui, 2010). Software project teams want SRS to be precisely accurate while users expect SRS to be more flexible to accommodate all CR. SRS grows dynamically throughout SDLC and requires collaborative effort while traceable and modifiable SRS is key to PS

(Hofmann and Lehner, 2001). Moreover, failure to define project scope boundaries and rapidly CR whether small/large outside initially defined scope parameters cause Scope Creep (SC) and it is an important factors in PS (Madhuri *et al*., 2014; Agarwal and Rathod, 2006). Project scope defines the boundaries of what is included/excluded from the project. Further, most critical aspect of SC is project scope definition and verification to avoid rework (Burke, 2009; Schwalbe, 2014). Inadequate Software Testing (ST) increases rework and leads to project failure. The past project performance data shows that rework adversely affects PS and it is a major cause of project failure in 40% of software projects (Standish Group, 1999). During initial stages of SDLC, rework magnitude/cost associated with fixing software bugs is comparatively low and manageable as compared to fixing bugs at final stages of SDLC.

Using Software Requirements Management Tools (SRMT) helps in streamlining communication gap between project stakeholders. SRMT helps in effective manipulation of rework (Cass *et al*., 2009). SRMT ensure that project team's do not waste time on verification of those requirements which were not critical for PS while ensuring requirements transparent accessibility to all stakeholders. PS is achieved through effective software requirements management in Scope Management (SM) knowledge area of Project Management (PM). Project managers perform 47 processes including 24 PP processes. Initial PP is the most prominent factor for PS agreed and reported by 744 surveyed project respondents (Besner and Hobbs, 2013). Time spent on PP activities reduces project risk and increases magnitude of PS (Wang and Gibson Jr., 2010). PP at initial stages of project plays an important role in PS. Inadequate PP eventually cause project failure (Thomas *et al*., 2008). PP processes consists of almost 50% of overall PM processes. PM ten knowledge areas (integration, scope, time, cost, quality, human resource, communication, risk, procurement and stakeholder management) and five process groups (initiating, planning, executing, monitoring and controlling along with closing) provide adequate guidelines for PS. PS criteria is based on key influencing factors (clearly defined project objectives, top management support, stakeholder's involvement and project planning), project's characteristics (like nature/type of project, product deliverables, resources constraints) and explicitly defined PS criteria (schedule/budget/scope deadlines) (Marques *et al*., 2013).

Rationale of study: Rework emerged as the most frequent critical issue in SDLC. Literature does not provide adequate guidelines for quantifying moderating role of SRMT between factors of rework and PS.

Problem statement: Rework adversely affects PS and is present at all stages of SDLC. Furthermore, maximum intensity of rework (40-100%) is present during requirements gathering phase (Focus, 2011). An empirical research to quantify the role of SRMT between factors of rework (like PP, SRS document quality (SRSDQ), ST, maturity of SDLC approach (MSDLCA), SC) and PS is required.

Literature review: Requirements analysis is a key process area of SC knowledge area in planning process group (PMI., 2013). An optimum amount of effort and resources are required to be spent for PP. Extensive time spent in PP is associated with poor project performance and cause lack of resources which force project managers to crash projects to meet schedule/budget constraints. Crashing of project cause extensive rework. Scope management knowledge area of PMBOK related to PP is critical for PS (Zwikael, 2009). Software practitioners disagreed that too much PP was always better for PS (Collyer and Warren, 2009). Rework is a sign of insufficient ST of the product and stakeholder's lack of interest in product features (Fairley and Willshire, 2005). Among three major types of rework, the Retrospective Rework (RR) is performed to achieve the work left in previous version of software product while Corrective Rework (COR) is performed to fix critical software bugs. Further Evolutionary Rework (ER) is performed to modify previous version of software's functionality, structure/quality. Extensive rework indicates problems in developer's skills and technology used in product development described in Fig. 1. Rework effort does not depend on size of CR rather it depends on complexity of CR, project size/nature and resource requirements (schedule, budget, technical skills, knowledge and experience) required for rework. SC is another factor for critical software applications relative to non-critical. Scope verification needs to be performed thoroughly to deliver according to customer needs.SC results due to poorly defined SRS with conflicting needs, wrong assumptions from customers, unwillingness to say no to customer, over delivering/gold plating. Further, SC is managed through a formalized Change Control Board (CCB) to grant scope changes, associating relevant cost of scope changes, communicating scope changes impact analysis with customer, ensuring customers to sign off scope change documents. In addition, SC is a critical source of projects cost creep while forcing clients to pay additional cost. Rework is work performed again because it is not properly done for first time. In consequence the cost associated with rework is not only limited to extra wastage of time/money on resources, it extends to schedule delays, crushes customer's confidence, damages brand image and return on investment.

Evolutionary Rework	Retrospective Rework	Corrective Rework
Caused by unanticipated events like requirements and design changes in the previous version of the product for the development of next version.	The project team knows project future needs, but chooses not to include them in previous version due to schedule constraints.	The project team fixed defects found in the current or previous version of the project.
It added new features to current product and modified current version.	It added features in previous version of product and caused time and schedule delays.	It added nothing to previous/ current version of product and caused time and schedule delays.
Evolutionary rework was found best for the project if it added new features without schedule/ budget overruns.	Retrospective rework was found good if a small amount of effort is required to do the work now.	Corrective rework was found best for the project when the total effort was within the project control limits ($\mu \pm 3\sigma$).
Evolutionary rework was bad for a project if it caused schedule/budget overruns.	Retrospective rework was bad if it occurred routinely in SDLC.	Corrective rework was bad when the effort was beyond the project control limits ($\mu \pm 3\sigma$).

Fig. 1: Rework categories adopted from (Fairley and Willshire, 2005)

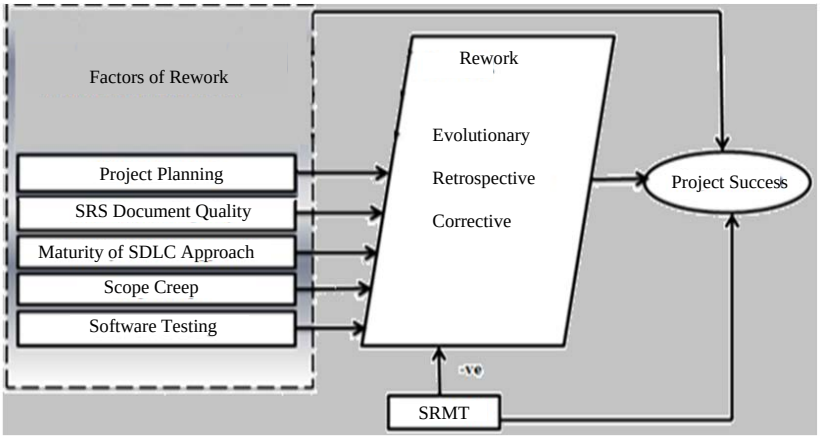

Fig. 2: Theoretical framework

Objectives of study/research questions: Based on literature review the following objectives and research questions are proposed for this study:

- What's role of SRMT between factors of rework and PS?
- How PP is associated with rework and PS?
- What's underlying association of SRS document quality with rework and PS?
- How ST impacts rework and PS?
- How rework is associated with MSDLCA and PS?
- Whether SC is associated with rework and PS?
- What's underlying relation of rework with PS?
- Which type of rework could be avoided and how?

Theoretical framework: Adaptive, functional and corrective CR caused evolutionary, retrospective and corrective rework due to lack of well-defined SRS (Chua and Verner, 2010). More than 40% of rework is present in requirements gathering phase (Focus, 2011). In consequence this study is aimed to quantify underlying associations of factors of rework with rework and PS. Theoretical framework depicted in Fig. 2 shows that SRMT moderates the associations between rework and PS and rework mediated between factors of rework and PS.

MATERIALS AND METHODS

Self-administered questionnaire was distributed among randomly selected 224 project teams from eighteen software houses with an estimated population of 500 (Sekaran, 2000). Study population included project teams from CMMI level II and above software houses. In

Table 1: Coding of data for analysis and interpretation

Variables	Values
Strongly disagree/very little	1
Disagree/little	2
Neither agree nor disagree/neither little nor large	3
Agree/large	4
Strongly agree/very large	5

Table 2: Reliability analysis

Variables	No. of items	Cronbach's alpha
Project planning	9	0.911
SRS document quality	6	0.769
Software testing	4	0.743
Maturity of SDLC approach	2	0.488
Scope creep	8	0.820
Rework	5	0.873
Project success	7	0.910

addition, it was a correlational study and research subjects were project teams using linear software development approach on both already completed and near to completion software projects of previous 5 years having documented evidence of rework and using SRMT. Furthermore, study adopted valid, pretested measurement scales. So, measurement scale for SRS document quality was adopted from (Iqbal *et al*., 2012), ST was adopted from (Kan, 2002), SC was adopted from (Mirza *et al*., 2013), MSDLCA and rework was adopted from (Barry, 2011) while scale for PS was adopted from (Naqvi, 2007). Finally, responses were collected on a 5 point Likert scale which was ranked 1-5 according to Table 1.

Questionnaire reliability was evaluated by Cronbach's alpha and all values for factors of rework and PS were within acceptable range and $p<0.05$ was considered statistically significant as shown in Table 2.

RESULTS AND DISCUSSION

This study found that bug free ST is critical for PS since majority of projects were poorly tested. Consequently, its ripple effect propagates to previous phases of SDLC and ultimately increases number of artefacts and magnitude of rework. The magnitude of software defects removed was low in 44% projects as shown in Table 3.

Accordingly magnitude of software bugs found later by end users was low in 59% projects. In addition, low effort and project resources were used for software testing which caused extensive rework. Firstly, overall high magnitude of rework was present in 66% surveyed projects. Secondly, in 64% of software projects high intensity of COR was present due to poor programming/ logical errors in code. In addition, 56% of projects faced high magnitude of RR, performed due to poor quality of work. Finally, magnitude of ER performed to enhance existing product features remained high as depicted in Table 4. In 45% software projects overall very poor quality of SRS documents was maintained. Since, moderate level of bugs were present in SRS shown in Table 5.

In consequence numbers of bugs removed/modified from SRS were found low in 44% projects. Finally, in 48% projects very low effort was allocated to maintain quality of SRS document which means low effort/project resources were used to ensure quality of SRS documents. Further study found that MSDLCA helped in rework reduction and enhanced chances of PS. Since, in 48% of software projects new defects occurred after little durations while in 52% of projects overall MSDLCA remained low as depicted in Table 6.

Study found that SC is critical and multi factor process. Furthermore, explicitly defined project goals/ objectives contributed little in SC in 72% projects. In addition, change requests formally approved, rejected/ deferred by CCB and documenting changes in scope change log also contributed little in SC. Finally, stakeholder's involvement in defining project scope, incomplete/partial definition of scope documents and restricted sharing of project scope statement contributed little in SC as shown in Table 7.

Various factors (project performance standards, planning for on time product delivery, identifying projects risks, project progress/status reports planning, schedule, scope, cost and stakeholder management planning) contributed in effective PP which reduced rework and leads to PS as depicted in Table 8.

Pearson correlational analysis shows that with PP, SRSDQ, ST, MSDLCA and SRMT, PS was ensured $R = 0.690$, $r = 0.511$, $r = 0.575$, $r = 0.581$, $r = 0.611$, $p<0.01$) as depicted in Table 9. Further, effective PP, SRSDQ, ST, MSDLCA and SRMT ensure up to 48, 26, 33 and 37% of PS, respectively. In addition, usage of SRMT and effective PP were the most critical factors to ensure very high chances of PS. Furthermore, ST ensured high chances of PS while factors like SRSDQ ensured moderate level of PS. While a significant low negative correlation was calculated between SC and PS ($r = -0.145$, $p<0.01$). Study found that SC decreased 2% magnitude of PS. In addition, significant moderate negative correlation was present between PP, SRSDQ, ST, MSDLCA, SRMT, PS and rework $R = -0.495$, $r = -0.284$, $r = -0.419$, $r = -0.560$, $r = -0.345$, $r = -0.485$, $p<0.01$). Accordingly this study found that PP reduced 21% rework, SRS Document Quality reduced 08% rework, ST reduced 18% rework, MSDLCA reduced 15% rework and SRMT reduced 31%rework. Further, significant low positive correlation was calculated between SC and rework ($r = 0.133$, $p<0.01$) which means that SC increased 2% rework magnitude. In addition, rework reduced 24% chances of overall PS. While SRMT and PP were prime factors in significant rework reduction

Table 3: Descriptive statistics of software testing

	Statistics				
Variables	Scales	Age (%)	Means	Mode	SD
No. of defects	Very low	18.8	2.69	3	1.16
	Low	24.6			
Removed during	Neither low nor high	31.7			
SDLC	High	18.8			
	Very high	6.3			
Total number of defects	Very low	23.7	2.34	2	1.04
found later	Low	34.8			
	Neither low nor high	27.7			
	High	11.6			
	Very high	2.2			
No. of soft req. tested	Very low	18.8	2.77	3	1.2
during testing phase	Low	23.2			
	Neither low nor high	29.5			
	High	19.6			
	Very high	8.9			
No. of defects removed	Very low	22.3	2.70	3	1.2
during testing phase	Low	21.4			
	Neither low nor high	27.2			
	High	21.9			
	Very high	7.1			

Table 4: Descriptive statistics of rework

	Statistics				
Variables	Scales	Age (%)	Means	Mode	SD
Rework performed to correct	Very low	2.7	3.79	4.00	1.03
critical errors errors	Low	7.6			
	Neither low nor high	25.9			
	High	35.3			
	Very high	28.6			
Rework performed due to poor	Very low	4.9	3.60	4.00	1.16
quality of work	Low	12.9			
	Neither low nor high	26.3			
	High	29.0			
	Very high	26.8			
Rework performed to enhance	Very low	2.2	3.71	4.00	1.01
existing product features	Low	9.4			
	Neither low nor high	28.6			
	High	35.3			
	Very high	24.6			
Total effects spent in fixing	Very low	0.9	3.79	5.00	1.06
rework	Low	12.1			
	Neither low nor high	26.3			
	High	28.1			
	Very high	32.6			
Overall rework ratio present in	Very low	3.1	3.77	4.00	1.01
the project	Low	6.3			
	Neither low nor high	27.7			
	High	36.2			
	Very high	26.8			

Table 5: Descriptive statistics of SRSDQ

	Statistics				
Variables	Scales	Age (%)	Means	Mode	SD
No. of errors found in SRS	Very low	15.2	2.63	3	1.01
	Low	28.6			
	Neither low nor high	37.5			
	High	16.1			
	Very high	2.7			
No. of errors removed from	Very low	16.1	2.68	3	1.08
SRS	Low	26.3			
	Neither low nor high	35.7			
	High	17.0			
	Very high	4.9			

Table 5: Continue

	Statistics				
Variables	Scale	Age (%)	Means	Mode	SD
No. of errors modified in SRS	Very low	18.3	2.55	3	0.98
	Low	25.0			
	Neither low nor high	41.1			
	High	14.7			
	Very high	0.90			
Total number of req. in SRS document	Very low	19.2			
	Low	22.3			
	Neither low nor high	33.0			
	High	21.4			
	Very high	4.00			
Number of correct req. found in SRS document	Very low	21.9	2.66	3	1.15
	Low	21.4			
	Neither low nor high	28.1			
	High	26.3			
	Very high	2.20			
Effort used to maintain quality of SRS document	Very low	26.8	2.58	1	1.23
	Low	20.1			
	Neither low nor high	26.8			
	High	21.4			
	Very high	4.90			

Table 6: Descriptive statistics of maturity of SDLC

	Statistics				
Variables	Valid	Age (%)	Means	Mode	SD
Duration, since, software products is working efficiently	Very low	24.1	2.63	4	1.18
	Low	21.4			
	Neither low nor high	24.1			
	High	28.6			
	Very high	1.80			
Duration in occurrence of new defects in software product	Very low	16.5	2.55	3	0.99
	Low	30.8			
	Neither low nor high	35.7			
	High	15.2			
	Very high	1.80			

Table 7: Descriptive statistics of scope creep

	Statistics				
Variables	Valid	Age (%)	Means	Mode	SD
Explicitly projects goals/deliverables objectives	Very low	44.2	1.85	1	0.86
	Low	27.2			
	Neither low nor high	27.7			
	High	0.90			
	Very high	0.00			
Change requests formal approval through CCB	Very low	44.6	1.84	1	0.85
	Low	26.8			
	Neither low nor high	28.1			
	High	0.40			
	Very high	0.00			
Change requests rejected/deferred	Very low	48.7	1.78	1	0.86
	Low	25.9			
	Neither low nor high	24.6			
	High	0.40			
	Very high	0.40			
Change requests documented in scope change log	Very low	49.6	1.75	1	0.87
	Low	29.9			
	Neither low nor high	17.9			
	High	1.80			
	Very high	0.90			
Scope change requests impact on project plan	Very low	53.1	1.68	1	0.81
	Low	26.3			
	Neither low nor high	20.1			
	High	0.40			
	Very high	0.00			

Table 7: Continue

	Statistics				
Variables	Valid	Age (%)	Means	Mode	SD
Stakeholder's involvement in defining project scope	Very low	51.8	1.17	1	0.84
	Low	28.1			
	Neither low nor high	18.8			
	High	0.40			
	Very high	0.90			
Unclear/incomplete or partial defination of scope documents	Very low	47.3	1.74	1	0.84
	Low	34.8			
	Neither low nor high	15.2			
	High	1.80			
	Very high	0.90			
Restricted sharing of projects scope statement	Very low	46.4	1.79	1	0.87
	Low	30.8			
	Neither low nor high	20.1			
	High	2.20			
	Very high	0.40			

Table 8: Descriptive statistics of project planning

	Statistics				
Variables	Valid	Age (%)	Means	Mode	SD
Project performance standard	Very low	15.18	2.83	3.00	1.10
	Low	20.54			
	Neither low now high	33.93			
	High	26.79			
	Very high	3.570			
Planning to ensure project outputs are delivered on time	Very low	17.41	2.92	4.00	1.24
	Low	19.64			
	Neither low nor high	25.89			
	High	28.13			
	Very high	8.930			
Identify project risks problems issues	Very low	17.86	2.74	4.00	1.17
	Low	26.79			
	Neither low nor high	22.32			
	High	29.46			
	Very high	3.570			
Effective communications within project stakeholders	Very low	22.77	2.79	4.00	1.28
	Low	18.30			
	Neither low nor high	23.21			
	High	28.57			
	Very high	7.140			
Planning project progress and status reports	Very low	13.84	2.88	3.00	1.14
	Low	23.66			
	Neither low nor high	29.02			
	High	27.68			
	Very high	5.800			
Project scheduled management planning	Very low	16.52	2.90	4.00	1.17
	Low	19.64			
	Neither low nor high	25.89			
	High	33.48			
	Very high	4.460			
Project scope management planning	Very low	17.86	2.82	4.00	1.18
	Low	22.32			
	Neither low nor high	24.11			
	High	31.25			
	very high	4.460			
Project budget management planning	Very low	18.30	2.85	3.00	1.20
	Low	18.75			
	Neither low nor high	29.02			
	High	27.23			
	Very high	6.700			
Project stakeholders management planning	Very low	19.20	2.79	4.00	1.20
	Low	21.88			
	Neither low nor high	25.00			
	High	28.57			
	Very high	5.36			

Table 9: Correlations

Correlations	Project success	Rework
Project planning	0.690	-0.495
SRS document quality	0.511	-0.248
Scope creep	-0.145	0.133
Software testing	0.575	-0.419
Maturity of SDLC approach	0.581	-0.385
SRMT	0.611	-0.560
Project success	1	-0.485

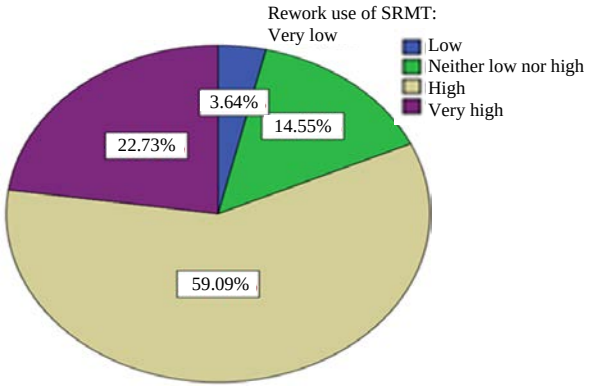

Fig. 3: High rework and very low SRMT

and ensuring high level of PS. Finally, SRMT, PP and rework statistically significantly predicted PS, thus, model equation is:

$$\text{Project success} = 0.779 + 0.046\,(\text{software testing}) + 0.087\,(\text{SRS document quality}) + 0.106\,(\text{maturity of SDLC approach}) - 0.065\,(\text{scope creep}) + 0.403\,(\text{project planning}) + 0.355\,(\text{SRMT}) - 0.119\,(\text{Rework})$$

Firstly, study quantified that 1% increase in magnitude of SC and Rework decreased PS magnitude by 6 and 12%, respectively. In addition, 1% increase in magnitude of ST, SRSDQ, MSDLCA, SRMT and PP increased the magnitude of PS by 4, 8, 10, 35 and 40%, respectively. Further, study revealed that in view of 38% project teams staying within budget constraints was not critical for PS. While in view of 43% of project teams, scheduled limits were also not critical for PS. Software projects in which usage of SRMT remained either very low/low; the intensity of rework remained high. While rework magnitude in 60% of projects remained high and very high in 25% projects in which usage of SRMT was reported very low as depicted in Fig. 3. Furthermore, rework is dependent on total project completion duration while small/medium/long term software projects faced varying amount of rework. In addition, majority of small to medium term software projects (0-2 year's duration) faced high magnitude of rework while in long term software projects (2-3 or more than 3 year's duration) magnitude of rework was reported very high. Furthermore, usage of SRMT remained very low in more than 75% of software projects. Consequently, projects which used SRMT faced low magnitude of rework as compared to projects which did not used SRMT. Finally, study model was found significant without introducing product term of factors of rework and SRMT and model was also found significant after introducing the product term of factors of rework and SRMT and interaction between factors of rework and SRMT accounted for more variance.

CONCLUSION

Firstly, usage of SRMT is prime factor in rework reduction to ensure high level of PS and it critical moderating role in controlling the relationship between factors of rework and PS. Furthermore, addition/update related CR caused extensive rework while deletion related CR required less rework effort in SDLC. In addition, COR could be avoided by ensuring that programming standards were strictly followed while writing lines of code and minimizing logical errors. Consequently, high intensity of RR could be avoided by ensuring that small amount of effort is required in fulfilling future project needs. Further, high intensity of ER was due to poorly defined SRS/poor communication between end-users and software developer. In addition, ER is unavoidable and good if addition of new module or features did not cause extensive schedule/budget overruns. Finally stakeholder's requirement satisfaction is critical factor for PS since projects which used SRMT faced relatively low magnitude of rework as compared to those which did not used SRMT. Furthermore, rework is dependent on total project completion duration and rework magnitude increases with project completion duration while multiple factors including initially defined project goals, change requests formal approval through CCB, stakeholder's involvement in defining project scope and restricted sharing of project scope statement contribute in SC.

RECOMMENDATION

This research determined critical moderating role of SRMT. Future, research could see effect of SRMT among individual SDLC phases to find role of various types of rework among individual phases of SDLC.

REFERENCES

Agarwal, N. and U. Rathod, 2006. Defining success for software projects: An exploratory revelation. Int. J. Project Manage., 24: 358-370.

Barry, R.M., 2011. Cert ware workbench metrics: A workbench for safety case production and analysis. Proceedings of the IEEE Aerospace Conference, March 5-12, 2011, Big Sky, USA., pp: 1-10.

Besner, C. and B. Hobbs, 2013. Contextualized project management practice: A cluster analysis of practices and best practices. Project Manage. J., 44: 17-34.

Bokhari, M.U. and S.T. Siddiqui, 2010. A comparative study of software requirements tools for secure software development. BVICAM's Int. J. IT (BIJIT.), 2: 207-216.

Burke, R., 2009. Fundamentals of Project Management: Tools and Techniques. 2nd Edn.,/Vol. 1, Burke Publishing, London, UK., ISBN: 9780958273367, Pages: 379.

Cass, A.G., L.J. Osterweil and A. Wise, 2009. A pattern for modeling rework in software development processes. Proceedings of the International Conference on Software Process, May 16-17, 2009, Springer, Berlin, Germany, pp: 305-316.

Chua, B.B. and J. Verner, 2010. Examining requirements change rework effort: A study. Int. J. Software Eng. Appl., 1: 48-64.

Collyer, S. and C.M. Warren, 2009. Project management approaches for dynamic environments. Int. J. Project Manage., 27: 355-364.

Damian, D., J. Chisan, L. Vaidyanathasamy and Y. Pal, 2005. Requirements engineering and downstream software development: Findings from a case study. Empirical Software Eng., 10: 255-283.

Fairley, R.E. and M.J. Willshire, 2005. Iterative rework: The good, the bad and the ugly. Comput., 38: 34-41.

Focus, M., 2011. Successful projects start with high quality requirements. Micro Focus Inc., Rockville, Maryland.

Gorschek, T., 2006. Requirements engineering supporting technical product management. Ph.D. Thesis, Blekinge Institute of Technology, Karlskrona, Sweden.

Hennicker, R. and N. Koch, 2000. A UML-based methodology for hypermedia design. Proceedings of the 3rd International Conference Unified Modeling Language, (UML'00), Springer Verlag, New York, pp: 410-424.

Hofmann, H.F. and F. Lehner, 2001. Requirements engineering as a success factor in software projects. IEEE. Software, 18: 58-66.

Iqbal, S., M. Naeem and A. Khan, 2012. Yet another set of requirement metrics for software projects. Int. J. Software Eng. Appl., 6: 19-28.

Kan, S.H., 2002. Metrics and models in software quality engineering. 2nd Edn., Addison-Wesley, New York, ISBN: 0201729156.

Lang, M. and J. Duggan, 2001. A tool to support collaborative software requirements management. Requirements Eng., 6: 161-172.

Madhuri K.L., J.R. Jawahar and V. Suma, 2014. Effect of scope creep in software projects: Its bearing on critical success factors. Int. J. Comput. Applic., 106: 9-13.

Marques, A., J. Varajao, J.J. Sousa and E.P. Correia, 2013. Project management success ICE model-a work in progress. Procedia Technol., 9: 910-914.

Mirza, M.N., Z. Pourzolfaghar and M. Shahnazari, 2013. Significance of scope in project success. Procedia Technol., 9: 722-729.

Naqvi, I.H., 2007. Developing a framework for effective IT project management and best HR practices. Ph.D., Thesis, National University of Modern Languages Islamabad, Islamabad.

Osmundson, J.S., J.B. Michael, M.J. Machniak and M.A. Grossman, 2003. Quality management metrics for software development. Inf. Manage., 40: 799-812.

PMI., 2013. A Guide to the Project Management Body of Knowledge (PMBOK Guide). 5th Edn., Project Management Institute, Newton Square, PA., USA.

Schwalbe, K., 2014. Information Technology Project Management. 7th Edn., Cengage Learning, Boston, Massachusetts,.

Sekaran, U., 2000. Research Methods for Business: A Skill Building Approach. 3rd Edn., John Wiley, New York, USA.

Sharif, B., S.A. Khan and M.W. Bhatti, 2012. Measuring the impact of changing requirements on software project cost: An empirical investigation. Int. J. Comput. Sci. Issues (IJCSI.), 9: 170-174.

Standish Group, 1999. Chaos manifesto: A recipe for success. Standish Group International, Boston, Massachusetts.

Thomas, M., P.H. Jacques, J.R. Adams and J. Kihneman-Wooten, 2008. Developing an effective project: Planning and team building combined. Project Manage. J., 39: 105-113.

Wang, Y.R. and G.E. Gibson Jr., 2010. A study of preproject planning and project success using ANNs and regression models. Automation Constr., 19: 341-346.

Zwikael, O., 2009. The relative importance of the PMBOK® guide's nine knowledge areas during project planning. Project Manage. J., 40: 94-103.