EFFECT OF REWORK ON PROJECT SUCCESS

[1] **Faisal Adnan,**
ei.netsolian@gmail.com
[2] **Imran Haider Naqvi,**
drimranhaider@ciitlahore.edu.pk
[1]COMSATS Institute of Information Technology, Lahore.
[2] CIF, COMSATS Institute of Information Technology, Lahore

ABSTRACT: *This paper provides an insight to project success (PS) factors & criteria. Software project teams focus on various factors to achieve the predetermined PS criteria based on the nature of the project. Achieving the project milestones in terms of the triple constraints of iron triangle did not guarantee the PS. PS could never be guaranteed by doing the right work until & unless the stakeholder's perception & vision is not converted into reality. This study contributed in exploring the various rework types & their impact on PS. This study contributed that magnitude of rework is dependent on total project completion duration & rework can be avoided in SDLC. The study analyzed the association between rework & PS. Major causes of unsuccessful projects in view of project team members were explored through a survey conducted in the software industry of Lahore Pakistan. .*

Keywords: Project success criteria, Iron triangle, Success factors, Rework, Project success

1. INTRODUCTION

Project success (PS) criterion varied from one stakeholder to another within software project teams. Various criteria's were used as a standard approach to judge & measure PS. Different factors influenced PS & it was normally viewed in terms of meeting or exceeding predestined objectives of schedule, budget & quality. On the other side stakeholder's have no predefined criteria to measure PS. Project teams primarily focused on schedule & budget constraints. PS was seen differently across the various stages of software development life cycle (SDLC). Turner & Zolin [1] viewed PS in terms of project scope completed within schedule/budget constraints & product delivered in accordance with the agreed software requirements specification (SRS). After the software product delivery PS was perceived in terms of product performance in accordance with the predefined functional /nonfunctional specifications in SRS to judge whether the software product provided intended benefits. At later stages of SDLC, PS was judged in terms of achieving organizational strategic objectives & goals. Shenhar, Levy & Dvir [2] expressed that PS was time-dependent: As time goes by, it matters less whether the project meet the resource constraints; in most cases, after about one year it becomes completely irrelevant. In contrast, after project completion the second dimension of customer's satisfaction becomes more relevant. They also emphasized that project teams should "see the big picture . . ., be aware of the results expected . . . and look for long term benefits" and gave four success factors in terms of project efficiency, product's impact on customers, business success & future measures preparation as critical success factors for PS. Stakeholder's interests must be incorporated as a long term PS factor. Lim & Mohamed [3] differentiated the PS criteria from PS factors & revealed that PS criterion was the set of principles & standards by which judgment was made from. The PS factors were the set of influences contributing to PS. Micro viewpoint of PS was classified in terms of meeting the triple constraints of scope, time & budget while macro viewpoint of PS in terms of stakeholder's satisfaction. Rework emerged as the most frequent burning issue which adversely affected PS. Data showed that almost half of the software projects effort & resource utilization was hidden in fixing/correcting the software defects. The past project performance reported that rework was major cause of project failure in 40% of software projects [StandishGroup, 4]. Rework cost upraised as software headed towards completion. During the initial stages of the project, rework magnitude & the cost associated with fixing bugs, was found comparatively low & manageable as compared to fixing software bugs found during the final stages of SDLC. Rework is a burning issue which adversely affected PS. MicroFocus [5] depicted that the rework was present at all stages of SDLC with maximum intensity during the requirements gathering phase. Major cause of rework in majority of software projects was lack of an integrated approach for communication, collaboration & automated software requirements management (SRM) [Ellis, 6]. Literature does not provided adequate guidelines to avoid/ reduce rework in SDLC. An empirical research for exploring the relationship between rework & PS was hence intended.

1.1 RESEARCH QUESTIONS

Based on literature review the following research questions were proposed for this study.

1).What is the impact of rework on PS?
2).How UseofSRMT impact PS/Rework?
3).What are most common types of rework faced in SDLC?
4).Whether rework is associated with project duration?
5). what type of / how much rework is avoidable in SDLC?
6). what are common reasons of projects failure in view of project team members?

2. LITERATURE REVIEW

PS needs to be viewed in broader context & beyond the traditional boundaries of the triple constraints of iron triangle [Collyer & Warren, 7]. Project teams looked for the achievement of short term goals. Stakeholder's satisfaction level was directly linked with PS. Shenhar et al [8] found that PS perceptions varied among project stakeholders & suggested four key dimensions of PS based on project efficiency, customer impact, business success & future adequacy of the project. Davis [9] found lack of agreement on PS perceptions among senior management & project team members. Project managers viewed quality as maintainability,

efficiency/effectiveness of project. While quality perception among other team mates was that how much productive the project was? Or how quickly the product was able to solve the problems? PS rate & criteria differ with projects industry & its complexity [Muller & Turner, 10]. PS criteria & objectives should be determined at the start of the project initiation phase. [Morris & Hough, 11] found that projects which violated, the time & cost estimates were still successful based on the nature of the project. Life critical or flood protection projects like the Thames Barrier (UK, London Flood Protection System) required extra time/cost, but still it was a successful project since stakeholder's prime interests were satisfied.

McLeod et al. [12] developed a framework for the project managers to recognize PS & described PS as an emergent, multidimensional & subjective process based on various perspectives. Savolainen et al. [13] viewed PS from supplier's perspective & identified customer satisfaction, project /customer's short term & long term success as the major criteria for PS. There were numerous examples of software projects where the original objectives were not achieved, but project stakeholders were highly satisfied with the project's outcome and the projects remained successful. There were examples where the project objectives were met, but the clients were quite unhappy with the results and the projects failed. Collyer & Warren [7] give an example of Titanic movie, which was originally late & over budget but the project was eventually successful. Projects meeting the stockholders expectations & interests helped in achieving PS. The Sydney Opera House project was completed with a delay of 15 years & a budget overrun of 14 times the original budget, but still it was a successful project & an engineering masterpiece of Sydney [Fodor's,14]. PS needs to be visualized beyond the triple constraints of the iron triangle, i.e. time, cost & quality. These factors were worthwhile for PS but project teams were unable to certify that whether a project was successful even if the project was completed within scheduled & estimated budgeted parameters.

Agarwal & Rathod [15] defined PS in view of internal stakeholders like software project team members & concluded that the project scope (composed of the functionality and quality of the final product) was the ultimate PS criteria. Quality factor was a phenomenon which changed with the type & nature of the project [Roger,16]. Budget & schedule estimates were the best guesses used when project teams possessed limited information about the project. These estimates were based on the learning's & experiences from previous project's knowledge base library. Project teams were reluctant to visualize the project beyond the cost, time & quality guesses. Project teams looked for smart processes to develop successful products & never looked for smart PS criteria. Project teams tried to meet the two best guesses of time & cost while focusing on the quality phenomenon. This was described as "doing something right to meet the milestones of the project". But it was not the guarantee of PS. PS should be visualized as an art & science of converting stakeholder's vision, perceptions & philosophy into reality [Turner,17]. Stakeholder's interests & satisfaction should be focused as a critical success factor. Projects which fulfilled the iron triangle success criteria were not successful projects until & unless they were productive & meet or exceeded stakeholder demands. Project milestones should not be used as a measure of PS [Williams,18]. Milestones were set for short term measures & the earned value methods were used to monitor the project progress to ensure that project was proceeding on right track. PS criteria depend on type of project. Medical & real time projects were more quality focused. Project team members, managers & top management along with customers were important stakeholder's in determining the PS.

DeLone & McLean [19] visualized six legitimate factors of PS in the context of system quality, information quality, information use, user satisfaction, individual impact & organizational impact & also focused on five performance gaps that occurred between inevitable stakeholders like customers, project teams & project outcomes at different phases of the project. Long term benefits should also be included in the PS criteria. Literature shows that over the past few years, PS rate kept very low up to 16.2%, while 52.7% of the projects were unsuccessful due to schedule or budget overruns.Various factors contributed to the PS but could not be specified as the ultimate PS criteria. Critical success factors which contributed to PS could be seen in Table 1 adopted from [Attarzadeh & Ow, 20].

Table 1: Success factors contributing to PS adopted from [Attarzadeh & Ow, 20]

Project Type	Critical Success Factors	Description
Successful Projects	Customer's satisfaction Clear statement of requirements	A project was successful if it meets the customer's needs. Clearly stated requirements provided a baseline for software project success.
Unsuccessful Projects	Incomplete requirements Changing requirements	Poor management of incomplete & CR caused schedule & budget overruns.
Failed Projects	Unrealistic expectations	Unrealistic expectations from stakeholders & poor RM caused project failures.

Rework is the work performed again because it was not properly done for the first time. Software teams wasted majority of time & resources in rework. Rework increased as projects progressed in SDLC. The cost associated with rework was very high & it was not limited to extra wastage of time & money on resources. It caused schedule delays, crushed customer's confidence, damaged brand image & affected return on investment. Rework during software development phase caused 200 times more as compared with the rework performed during requirements analysis phase [Boehm &

Papaccio, 21]. Literature indicated three major types of rework. The retrospective rework (RR) was performed in order to achieve the work intentionally left in previous version of the software product. The corrective rework (COR) was performed in order to fix the software bugs. Evolutionary rework (ER) was performed to modify previous versions of software functionality, structure, behavior/quality [Fairley & Willshire, 22]. Various rework categories & their characteristics described in Table 2.

Table.2: Rework categories & characteristics adopted from [Fairley & Willshire, 22]

Evolutionary Rework	Retrospective Rework	Corrective Rework
Caused by unanticipated events like requirements & design changes in the previous version of the product for development of the 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 fixes defects found in the current or previous version of the project.
It added new features to current product & modified current version.	It added features in previous version of product & caused schedule delays.	It added nothing to previous / current version of product & caused schedule delays.
It was found best for the project if it added new features without schedule/ budget overruns.	It was found good if a small amount of effort is required to do the work now.	It was found best for the project when 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 ($\mu \pm 3\sigma$).

Fairley & Willshire suggested that during a specific reporting period in SDLC 10-20% of rework effort were commonly accepted while excessive rework indicated problems in RM process, developer's skills & the technology used in product development. Rework reduction boosted productivity & produced high quality software products. Rework impact raised exponentially as software progressed to later stages of SDLC. UseofSRMT helped in tailoring rework & streamlined the communication gap between project stakeholders [Wang et al, 23]. [Ellis, 6] stated that effective RM during the initial stages of the project life cycle enhanced chances to meet the PS criteria & reduced project overruns by almost 87%. [IBM, 24] study concluded that using an internal website for RM process promoted communication among project stakeholders & facilitated team's cohesiveness.

Hence, the research hypothesis can be stated as:

H1: UseofSRMT is negatively related with rework.
H2: Rework is negatively related with PS.
H3: UseofSRMT is positively related with PS.
H4: Rework is avoidable in SDLC.

3. THEORETICAL FRAMEWORK

MicroFocus [5] showed that 40% or more rework in SDLC was present in requirements phase. Fairley & Willshire [22] indicated three major types of rework i.e. RR, COR, ER. Wang et al. [23] found that UseofSRMT helped in tailoring rework. Similarly [IBM, 24] concluded that effective RM lead to PS. The theoretical framework implied that all the rework types have negative association with UseofSRMT & PS. PS is dependent on effective UseofSRMT & magnitude of rework in SDLC as shown in Figure 1.

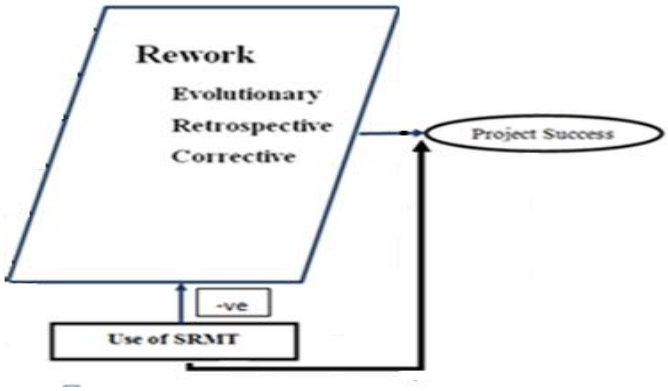

Figure.1: Rework role in project success

4. MATERIALS & METHODS

The research following a pilot study was carried out at the eighteen software houses. Self-administered questionnaire was distributed among randomly selected project team members. It was correlational study. The study design was cross sectional as the data was collected at the same point in time. The research subjects were project team members of both accomplished & near to completion software projects of previous 5 years with documented evidence of rework & UseofSRMT in software projects. The study population included project team members of CMMI Level II & above or software houses with more than 15 project team members. A random sample of 224 project team members working on various software projects was selected from an estimated population of 500 [Sekaran, 25]. The study adopted valid, pretested measurement scales from the existing literature, i.e. [Barry, 26] used for rework & [NAQVI, 27; Simpson, 28] scales were used for PS. The responses were collected on a 5 point Likert scale ranked between 1-5 as depicted in Table 3.

5. EMPIRICAL FINDING:

5.1 RELIABILITY ANALYSIS:

Reliability of questionnaire was checked through cronbach's alpha. Correlation & regression analysis were used to test the research hypothesis. Table 4 showed that cronbach's alpha values for rework & PS were within acceptable range.

Table.3: Coding of data for analysis & interpretation

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.4: RELIABILITY ANALYSIS

Constructs	No. of Items	Cronbach's Alpha
Project Success	7	0.910
Rework	5	0.873

5.2 RESULTS & DISCUSSION:

This study found that overall a high magnitude of rework was present in 66% surveyed projects (Mean: 3.77±1.01) which required a high effort & resources to fix rework (3.79±1.05) as depicted in Figure2. In 64% of software projects high intensity of COR was present (Mean: 3.79± 1.02). This study found that high intensity of COR was due to poor programming/logical errors in code & lack of effective RM process. This study contributed that COR was avoidable in SDLC by ensuring that programming standards were strictly followed while writing lines of code & minimizing logical errors during software development. Survey statistics of this study showed that almost 56% of software projects faced high magnitude of RR, performed due to poor quality of work (Mean: 3.60±1.15). This study concluded that high intensity of RR was avoidable in SDLC by ensuring that small amount of effort was required in fulfilling future needs of project during software development. Magnitude of ER performed to enhance existing product features remained high (3.71±1.01). This study found that high intensity of ER was due to weakly defined requirements characteristics, poor communication between user/software developer & lack of effective RM process. ER was unavoidable in SDLC if the addition of new module/features did not cause extensive schedule/ budget overruns. This study found that rework was dependent on the total project completion duration. Small, medium or long term software projects faced varying amount of rework in SDLC. Survey results showed that majority of small to medium term software projects (0-2 year's duration) faced high magnitude of rework while long term software projects (2-3 or more than 3 year's duration) faced very high magnitude of rework.

Figure 2: Descriptive statistics of rework in SDLC

Statistics	Valid	Mean	Mode	Std. Deviation
Rework Performed to Correct Critical Errors	224	3.79	4	1.02
Rework Performed due to Poor Quality of Work	224	3.60	4	1.15
Rework Performed to Enhance Existing Product Features	224	3.71	4	1.01
Total Effort Spent In Fixing Rework	224	3.79	5	1.05
Overall Rework Ratio Present in the Project	224	3.77	4	1.01

This study complemented the findings of [Collyer & Warren, 7] that PS should be visualized in the broader context while stakeholder requirement satisfaction, meeting quality standard & revenue generated from project were ultimate PS criteria. This study found that in view of 38% project team members staying within budget constraints was not critical for PS (Mean: 2.88 ±1.18). In view of 43% of project team members, project's scheduled limits were found not critical for the PS (Mean: 2.78±1.26). This survey showed that 41% of the project team members did not supported that staying within the pre-defined scope limits was the ultimate criteria for PS (Mean: 2.83±1.18). This survey also showed that 43% project team members agreed that stakeholder's requirement satisfaction was most critical factor for PS (Mean: 3.09±1.20). Statistics cleared that 41% of project team members agreed that revenue generated from the project was critical for PS (Mean: 3.00±1.26). PS statistics depicted that 39% project team members did not preferred that speed to market versus competition was linked with PS (Mean: 2.92±1.24). Also 44% project team members did not favor that meeting quality standard was the key to PS (Mean: 2.99±1.37). This study is in agreement with [Mark, 29] & further added that misunderstanding customer requirements & poor UseofSRMT were prominent factors for unsuccessful projects in view of project managers, team leads, functional leads & senior software engineers as depicted in Figure 3.

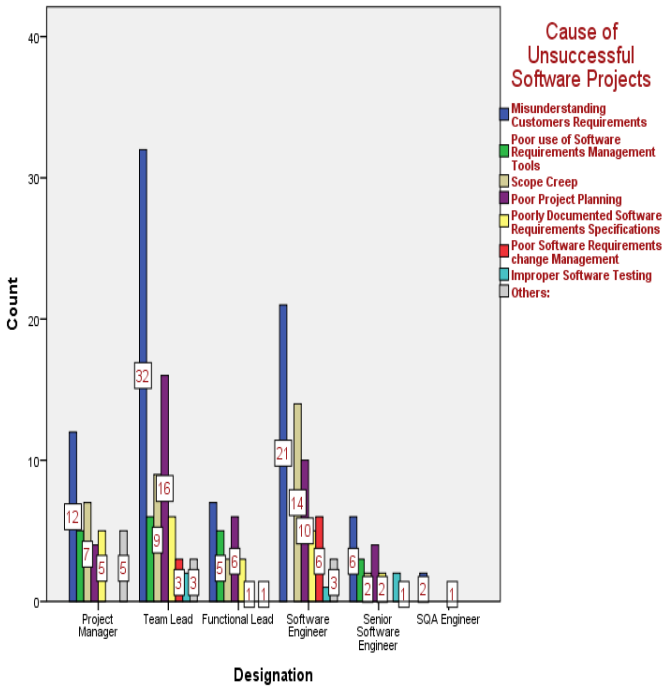

Figure.3: Cause of unsuccessful software project

The Pearson correlation coefficient data analysis between PS & UseofSRMT showed moderate positive correlation with a value of ($r=+0.478$, $p < 0.01$). A significant moderate negative correlation was found between PS & rework with a value of ($r=-0.485$, $p < 0.01$).Survey statistics of this study showed that in more than 75% of software projects, the UseofSRMT during SDLC remained low. Projects which used automated software requirements management tools (SRMT) faced relatively low

magnitude of rework in SDLC as compared to the projects which did not used SRMT as depicted in Figure 4.

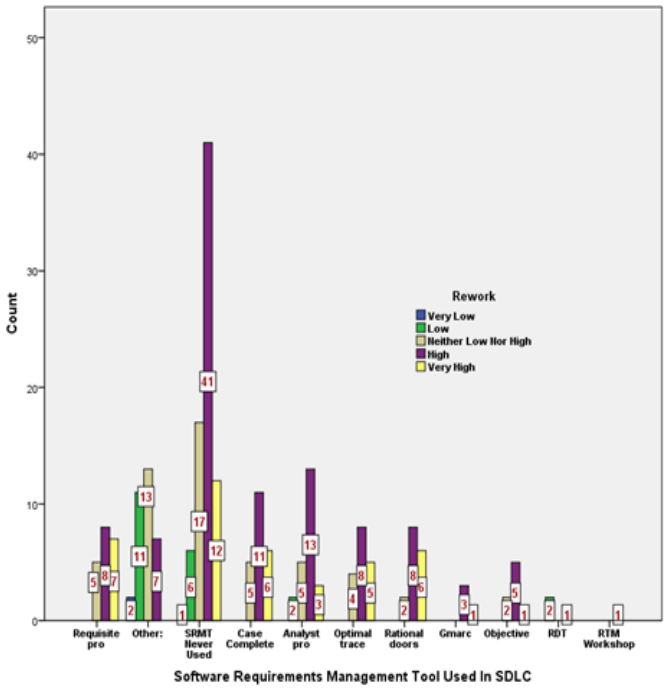

Figure 4: Rework & UseofSRMT in SDLC

6. CONCLUSIONS:

This study concluded that rework was dependent on the total project completion duration & found that small to medium term software projects faced high magnitude of rework while long term software projects faced very high magnitude of rework. This study supplemented findings of [Mark, 29] that misunderstanding customer requirements & poor UseofSRMT were prominent factors for unsuccessful projects in view of project managers, team leads, functional leads & senior software engineers. This study found that UseofSRMT was negatively related with rework & positively related with PS. This study explored that majority of software projects never used SRMT & projects which used SRMT faced relatively low magnitude of rework as compared to projects which did not used SRMT.

This study is in agreement with [Fairley & Willshire, 22] & further supplemented that majority of software projects faced high intensity of COR, RR & ER in SDLC which could be avoided. This study contributed that high intensity of COR was avoidable in SDLC by ensuring that programming standards were strictly followed while writing lines of code & minimizing logical errors during software development. This study further contributed that the high intensity of RR was also avoidable in SDLC by ensuring that a small amount of effort was required to fulfilling future needs of project. This study found that high intensity of ER was unavoidable & good in SDLC if addition of new module/features did not cause extensive schedule/ budget overruns. This study is in agreement with [Collyer & Warren, 7] & added that PS should be visualized in broader context while stakeholder's satisfaction, meeting quality standard & revenue generated from project were ultimate PS criteria.

7. FUTURE RECOMMENDATIONS:

Based on the critical literature review & the conclusion of this research paper the following recommendations are suggested for the future research,

1. The research focused on analyzing the overall rework role in PS. The future research could be more focused in exploring the role of various rework types in individual stages of SDLC.
2. Future research could help in quantifying the exact amount of rework present at various stages of the SDLC.

REFERENCES:

[1] Turner, R., & Zolin, R. (2012). Forecasting Success on Large Projects: Developing Reliable Scales to Predict Multiple Perspectives by Multiple Stakeholders Over Multiple Time Frames, *Project Management Journal*, **43**, 87-99.

[2] Shenhar, A., J., Levy, O., & Dvir, D. (1997). Mapping the dimensions of project success. *Project Management Journal*, **28**, 5-9.

[3] Lim, C., S. & Mohamed, Z. (1999). Criteria of project success: An exploratory re-examination. *International journal of project management*, **17**(4), 243-248.

[4] StandishGroup. (1999). Chaos Manifesto*: A Recipe for Success. Standish group international*.

[5] MicroFocus. (2010). *Successful Projects start with high quality requirements*, 1-10.

[6] Ellis, K. (2009). The path to success. IAG Consulting. *Business Analysis Benchmark.*

[7] Collyer, S., & Warren, C., M. (2009). Project management approaches for dynamic environments. *International Journal of Project Management*, **27**, 35-364.

[8] Shenhar, A., J., Levy, O., & Dvir, D. (2001). Project Success: *A Multidimensional Strategic Concept. Long Range Planning*, **34**(6), 699-725.

[9] Davis, K. (2013). Different stakeholder groups and their perceptions of project success. *International Journal of Project Management.*

[10] Muller, R. & Turner, J., R. (2007). Matching the project manager's leadership style to project type. *International Journal of Project Management*, **25**, 21-32.

[11] Morris, P., & Hough, G. (1987). The Anatomy of Major Projects, *A Study of the Reality of Project Management John Wiley, UK.*

[12] McLeod, L, B., Doolin, & MacDonell, S, G. (2012). A Perspective-Based Understanding of Project Success. *Project Management Journal.* **43**(5), 68-86.

[13] Savolainen, P., Ahonen, J., J., & Richardson, I. (2012). Software development project success and failure from the supplier's perspective: A systematic literature review. *International Journal of Project Management*, **30**(4), 458-469.

[14] Fodor's (1997). *Australia. Fodor's Travel Publications Inc.*

[15] Agarwal, N., & Rathod, U. (2006). Defining success for software projects: An exploratory revelation. *International Journal of Project Management*, **24**(4), 358-370.

[16] Roger, A. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, it's time to accept other success criteria. *International Journal of Project Management* ,**17**(6), 337-342

[17] Turner, J., R. (1996). Editorial: International Project Management Association global qualification, certification and accreditation. *International Journal of Project Management*, **14**(1), 1-6.

[18] Williams, B., R. (1995). Why do software projects fail?

GEC Journal of Research, **12**(1), 13-16.

[19] DeLone, W., H. & McLean, E., R. (1992). Information systems success: the quest for the dependent variable. *Information Systems Research*, **3**(1), 60-95.

[20] Attarzadeh, I., & Ow, S., H. (2008). Project management practices: The criteria for success or failure. *Communications of the IBIMA.*

[21] Boehm, B., W. & Papaccio, P., N. (1988). Understanding & Controlling Software Costs. *IEEE Transactions on Software Engineering*, **14**(10)**,** 1462–1477.

[22] Fairley, R., E. & Willshire, M., J. (2005). Iterative rework: the good, the bad, and the ugly. *IEEE computer society.*

[23] Wang, Q., Garousi, V., Madachy, R., & Pfahl, D. (2009). Trustworthy software development processes. Cass, A. J. Osterweil, L. J., & Wise, A. (Eds.), *A pattern for modeling rework in software development processes.*

[24] IBM .(2009). Reducing rework through effective requirements management, 1-6.

[25] Sekaran, U. (2003). Research Methods for Business, A Skill Building Approach (4th Edition). *New York: John Wiley and Sons.*

[26] Barry, R., M. (2011). Cert Ware Workbench Metrics: A workbench for safety case production and analysis. *Proceedings of the IEEE Aerospace Conference 2011.*

[27] NAQVI, I., H. (2009). *Developing a Framework for effective IT Project Management & Best HR Practices*. PhD thesis, National University of Modern Languages, Islamabad.

[28] Simpson, J. (2009). The State of Requirements Management: *The results of a recent industry survey shed light on the latest trends, challenges & solutions in software product development*, 1-19.

[29] Mark, A., L. (2014). PMI's Pulse of the Profession: The High Cost of Low Performance. A Core Competency for Project and Program Success. Project Management Institute