Extended Decision Support Matrix for Selection of SDLC-Models on Traditional and Agile Software Development Projects

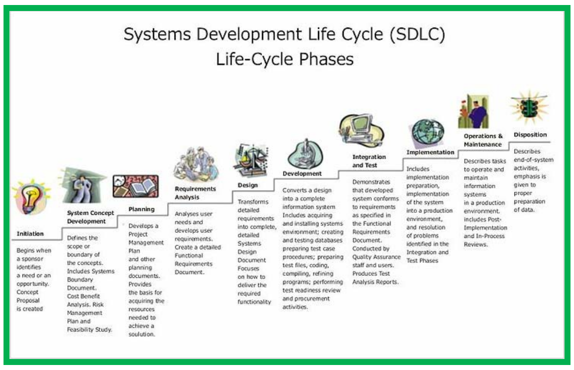
P. M. Khan and M. M. Sufyan Beg
Department of Computer Engineering
Jamia Millia Islamia (A Central University)
New Delhi – 110025, India
pmkhan@hotmail.com, mmsbeg@cs.berkeley.edu

***Abstract*- Software Development Projects can vary considerably in difficulty, size and type. This has led to evolution and development of many associated project management methodologies and standard SDLC-Models. This paper acknowledges the risks associated with wrong selection of SDLC-models on business critical software projects and offers a pragmatic solution by proposing a handy selection matrix for choosing best-fit SDLC models on different types of Software Development Projects, covering both traditional and agile methodologies. This paper is the result of an study carried out to evaluate the methods & practices of *Project Life Cycle Model Selection* actually used and practiced on the projects selected for this study (from businesses and IT-industry in India), with overall objective of proposing better methods and prescriptive guidance for decision making process for right selection of SDLC-Model on business critical software development projects. Right selection of SDLC-Methodology using a decision support tool can and will help successful completion of business critical software development projects and realization of business objectives for which the projects were undertaken.**

Keywords-SDLC, System Development Life Cycle, Software Development Life Cycle

I. INTRODUCTION

Application of structured SDLC methodologies are indispensible for business critical software development projects as they are designed to enforce various degrees of discipline to the software development process with the goal of making the process more efficient and predictable. Software life cycle models describe phases of the software cycle and the order in which those phases are executed. There are tons of models, and many companies adopt their own, but all have very similar patterns. The general, basic model of a typical, traditional SDLC-Model is shown in figure-1. As evident from figure-1, each phase produces deliverables required by the next phase in the life cycle. Requirements are translated into design. Code is produced during Development that is driven by the design. Integration and Test verifies the deliverable of the development phase against requirements.



Initiation: Begins when a sponsor identifies a need or an opportunity. Concept Proposal is created

System Concept Development: Defines the scope or boundary of the concepts. Includes Systems Boundary Document. Cost Benefit Analysis. Risk Management Plan and Feasibility Study.

Planning: Develops a Project Management Plan and other planning documents. Provides the basis for acquiring the resources needed to achieve a soulution.

Requirements Analysis: Analyses user needs and develops user requirements. Create a detailed Functional Requirements Document.

Design: Transforms detailed requirements into complete, detailed Systems Design Document Focuses on how to deliver the required functionality

Development: Converts a design into a complete information system Includes acquiring and installing systems environment; creating and testing databases preparing test case procedures; preparing test files, coding, compiling, refining programs; performing test readiness review and procurement activities.

Integration and Test: Demonstrates that developed system conforms to requirements as specified in the Functional Requirements Document. Conducted by Quality Assurance staff and users. Produces Test Analysis Reports.

Implementation: Includes implementation preparation, implementation of the system into a production environment, and resolution of problems identified in the Integration and Test Phases

Operations & Maintenance: Describes tasks to operate and maintain information systems in a production environment. includes Post-Implementation and In-Process Reviews.

Disposition: Describes end-of-system activities, emphasis is given to proper preparation of data.

Figure-1: Example of Life-Cycle Phases in Traditional Staged and Phased SDLC – adopted from [5]

Unfortunately, there is no "one size fits all" approach to applying SDLC methodologies [1]. Each SDLC methodology is only effective under specific conditions. It is important for project teams to understand the strengths and weaknesses of various SDLC-methodologies and make a right choice for decisions related to SDLC model section for delivery of each project. Although at a high-level, SDLC-models can be classified into two categories: traditional SDLC-Models and Light-Weight SDLC-Models, but regardless of which model one chooses, a project manager should review each phase of the cycle before continuing on to the next. The reason for this is that as a project continues, the organization usually commits more money and resources to it. Due to various factors such as compatibility with other organizational goals, potential success, and cost, it may require redirection or even termination. These review sessions by management are referred to as stage-gate review or phase exits or kill points. This is a very important aspect of the life cycle to keep the project on track, or in worst-case scenarios, to cease work on the project. In this research paper, an attempt is made to study three-traditional SDLC-Models and two-agile models as mentioned in following section II and III.

978-0-7695-4941-5/13 $26.00 © 2013 IEEE
DOI 10.1109/ACCT.2013.12

II. EXAMPLES OF TRADITIONAL SDLC-MODELS

There are many traditional SDLC-Models, available – each having it's own strength & weakness. For the purpose of this research, we have considered three traditional models viz. Waterfall Model, V-Process Model & 2I-Process Model., V-Process Model and 2I(Iterative/Incremental)-Process Model.

Waterfall Model:

Waterfall model is depicted in figure-2:

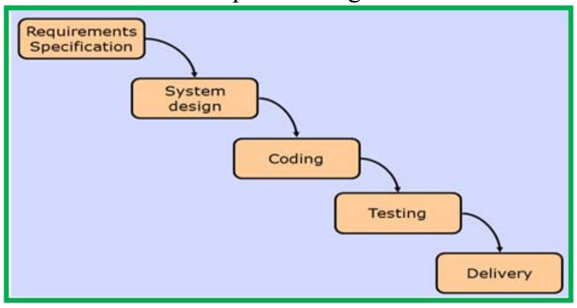


Figure-2: Waterfall Model

V-Process Model:

V-Process Model is depicted in figure-3:

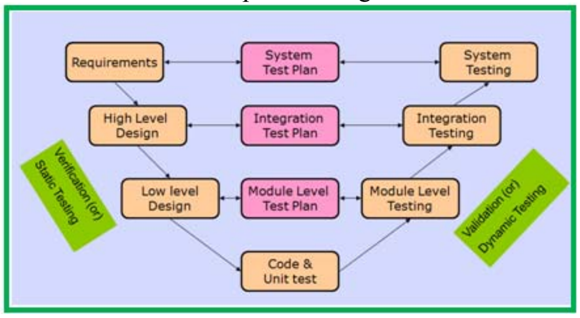


Figure-3: V-Process Model

*2I(Iterative/Incremental)-Process Model***:**

2I-Process Model is depicted in figure-4:

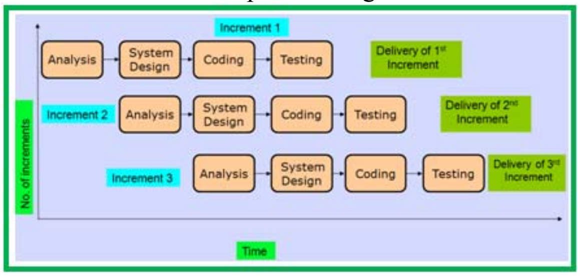


Figure-4: 2I (Iterative/Incremental)- Process Model

III. EXAMPLES OF AGILE SDLC-MODELS

There are many modern SDLC-Models available – each having its own strength & weakness. For the purpose of this research, we have considered two agile process models viz. XP and RUP.

eXtreme Programming(XP) Model:

XP-Process Model is one of the modern (light weight) software development life cycle model referred to as "extreme" because commonsense practices taken to extreme levels. XP is an agile methodology typically suited for small-to-medium-sized teams developing software with vague or rapidly changing requirements. Coding is the key activity throughout a software project. Communication among teammates is done with code and Life cycle and behavior of complex objects defined in test cases – again in code. The concept of eXtreme Programming is based on some simple rules. Fig-5 shows how Extreme Programming's rules work together.

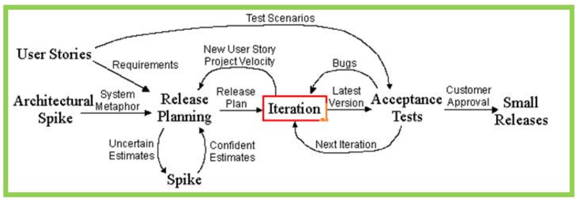


Figure-5: XP Process Model [16]

*Rational Unified Process (RUP)-Model***:**

RUP is also an agile option of SDLC-Model that offers a modern light weight approach for software development methodology. As a matter of fact, RUP is a product of Rational Corporation (acquired by IBM), that offers a configurable software development process platform that delivers practices and a configurable architecture. In its simplest form, RUP consists of some fundamental workflows:

1. *Business Engineering*. Understanding the needs of the business.
2. *Requirements.* Translating business need into the behaviors of an automated system.
3. *Analysis and Design*. Translating requirements into a software architecture.
4. *Implementation:* Creating software that fits within the architecture and has the required behaviors.
5. *Test.* Ensuring that the required behaviors are correct, and that all required behaviors are present.

6. *Configuration and change management.* Keeping track of all the different versions of all the work products
7. *Project Management.* Managing schedule and resources.
8. *Environment.* Setting up and maintaining the development environment.
9. *Deployment.* Everything needed to roll out the project.

These activities are not seperated in time. Rather, they are executed concurrently though-out the lifetime of the project.
As depicted in figure-6, not much code gets written early in the project lifecycle; but the amount is not zero. Late in the project, most of the requirements are known, but some new ones are still identified.

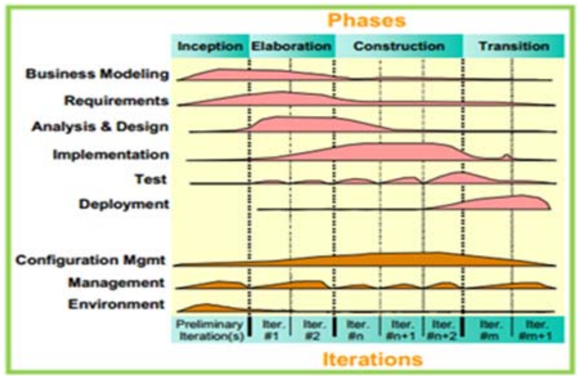


Figure-6: Phases and Iterations of RUP [17]

IV. RESEARCH BACKGROUND AND CONTEXT

Globalization and the internationalization of the markets in IT-industry have increased competitive pressures on business enterprises. Recent global recession has only worsened the problem and impacted the US and Europe markets and indirectly created huge pressures on Indian IT-companies, because most of the customers and revenues for IT-majors were historically linked to US and Europe markets [8]. These pressures have led companies to engage in projects that are not only critical to their performance, but also vital for survival of the enterprise. Majority of the companies continually strive to produce better results by undertaking strategic projects. Distressingly, recent industry trends have found that majority of the projects exceed budget, are completed past scheduled deadlines and do not meet original business objectives [9]. Several project life cycle models have been proposed over the past 52-years in growing body of literature to help improve project delivery performance of organizations managing software development projects, but there is limited evidence that selection of life cycle model for projects are performed objectively. This has been the trigger for this research work. Outcome of this research is envisaged to benefit SMBs and small software development organizations that are not yet having QMS or a matured PMO.

V. OBSERVATIONS ON RELATED WORK

Poor track record of software development projects continues to support the notion that there is a genuine business need for deeper study & further research to get a better understanding of project failures and delivery challenges encountered on real world business critical software development projects from the perspective of key contributors to project challenges and failures. The attempt to gain a more complete understanding of the nuances of business critical software projects has been of great interest to researchers, practitioners as well as to all professionally managed business organizations engaged with software development projects. Several lines of research exist in the growing body of literature dealing with the subject, all of it an attempt to get a solid understanding of various attributes of business critical software development projects [18],[19],[20]. Some researchers have focussed on standard SDLC-methodologies and it's adoption within the context of organizations. Life cycle models have been used in organizational research for many years to explain a wide range of organizational phenomena. Life cycle concept has been applied to the project management process in a variety of settings [10]-[12]. SDLC-Models have also been proposed for improving collaboration and stakeholder communication [6]. Traditional staged and phased methodologies (like Waterfall, V-Process etc.) are often regarded as the proper and disciplined approach to the analysis and design of software applications[2] but there exists another school of thought where traditional methodologies are viewed as too-rigid and/or too-heavy and perceived as falling short of rapid application development needs of business. Such businesses view traditional SDLC methodologies as being cumbersome, bureaucratic, and unsuited to the rapid pace of many software development projects. This opinion is relevant in the context of new e-business software environment, which has lead to evolution of light weight methodologies, in a separate line of research [3]. These lightweight methodologies are in principle a compromise between no process and too much process. These new methods were developed to efficiently manage software projects subjected to short timelines and excessive uncertainty and change.

Some of the most commonly referred lightweight SDLCs are Adaptive Software Development (ASD), Agile Software Process (ASP), Crystal, Dynamic System Development Method (DSDM), Extreme Programming (XP), Feature Driven Development (FDD), Rational Unified Process (RUP), SCRUM, and Whitewater Interactive System Development with Object Models (Wisdom) [4]. Ground realities suggest that operational teams within business organizations and having very little bandwidth available to engage in research and improve the level of understanding and optimization of project lifecycle processes. This has resulted into another line of research and creation PMOs within the organizations to add value propositions in software development projects [15] and improve the understanding of software project life cycle processes. Thorough understanding of life cycle models enables organizations to evolve their own tailored organizational methodology for delivery of software projects and best practices for organization[7],[13]. In [21], an attempt has been made to propose a selection matrix for SDLC-Models but this is limited to traditional SDLC-Models and does not address the problem of selection pertaining to modern Agile Methodologies.

As noted above, little work has been devoted to assessing the impact of choosing wrong SDLC-Methodology on the outcome of business critical software projects. This research paper is an attempt to address this issue, by making an assessment of possible impacts of wrong selection of SDLC-Methodology on the project itself and proposing an extended selection matrix for objective selection of SDLC-models encompassing 3-traditional models (waterfall, v-process and iterative/incremental-process model) and 2-agile models (XP and RUP).

VI. RESEARCH METHODOLOGY

The methodology adopted in this study was a combination of exploratory research as well and grounded theory research methods. There were two reasons for adopting such a method for this research:

(i) First, the research was aimed at the extension of existing theory. Grounded theory is generally deemed appropriate for such efforts as it allows theory to emerge from interview transcripts and organization artifacts.
(ii) Second, the methodology is reputed to help separate researcher biases from interpretation of the data.

To begin with, a survey of real world project managers and practitioners from IT-Industry covering multiple organizations and different geographies was also conducted in parallel, over multiple professional meets and technical forums to acquire real life experience of senior project managers on impacts of wrong SDLC-selection due to human factors. Questions posed to the participants of the survey are listed below in Table-1. A total of 59-project managers and practitioners participated in the survey and their objective responses are tabulated in Table-I.

TABLE-I: QUESTIONS USED IN SURVEY OF PRACTITIONERS

Wrong selection of SDLC-Methodology on Business Critical Software Projects (*due to inclination towards a specific methodology that project manager was experienced with and other political factors of the organization*) **will result into**:	
Option-1	No impact on project, as all methodologies are equally good
Option-2	No significant impact on project outcome, as methodology is immaterial
Option-3	Severely risk the project success, as selection of right life cycle methodology is vital for project outcome
Option-4	Major Performance Bottlenecks in the project
Option-5	Project Failure

Consequent upon survey, a further study of 11 real-life, global software development/implementation projects from a proper blend of multiple organization (covering matured organizations, SMBs and start-ups) was carried out to perform an assessment of life cycle methodology selection process and impacts of possible wrong selections. Projects chosen for this study were those which were using either traditional or modern light weight SDLC-methodologies, but found to be challenged with *schedule slippages* and *cost-overruns* beyond 50% were selected for detailed analysis and study from the perspective of life cycle related challenges and risks.

Project documents of all the projects selected for this detailed study - including Project Charted, Scope Document/SRS, Non-transactional Project-Plan, Transactional Project Plan, monthly PMRs, PDMRs, monthly presentations made to PMOs (wherever applicable) and monthly reports formally communicated to project sponsor and key stakeholder(s) were thoroughly examined. In the event of any ambiguities and lack of organizational processes, standard best practices of PMBOK [14] were referenced. Necessary clarifications were also sought from project manager/project team/PMO during the course of project document reviews. Authenticity of reports was reconciled with PMIS-data and a few inconsistencies observed (during reconciliation process) were challenged with reference to the project life cycle documentation - as and when required during the course of study. One of the key objectives of study was to understand how the life-cycle model selection was made in each of the projects under study and validate if the selection was done objectively or it was influenced by individual's bias. Impact of poor selection and lack of clarity on selection criteria were also given due consideration in this study, along-with relating it project challenges encountered. Each project was also examined on how the project life cycle selection was made and seeking documented evidence of alternative evaluation of possible options based on theoretical considerations of strengths and weaknesses of the life-cycle models.

A total of 32 persons, including project managers and senior team members of project team and PMO-staff (related to the projects selected for detailed study) were interviewed over a period of 6-months. The project managers chosen for the interview were generally very experienced in their position with average experience of more than 9-years and had previous experience of managing project budgets of multi-million-dollars. One of the project managers interviewed was also ex-employee, having left the organization within the preceding 3-months of the study.

The interviews were facilitated via open-ended questions intended to weed-out project life cycle related issues, challenges and suggestions. The resulting pages of interview transcripts were analysed to identify recurrent themes and concepts associated with good and bad times in the business. Those themes and concepts were then tested against archived data including survey results, with an eye to find evidence reinforcing or challenging the themes and concepts. Preliminary conclusions were then tested via follow-up interviews with a subset of original 32-individuals. This iterative process (interviews, concepts identification, concepts aggregation, theme analysis, testing against organizational artifacts, review with interviewees, adaption, repeat) led to the identification of key concepts and vital lessons learnt for Project Life Cycle Selection. Results of practitioner survey and subsequent detailed study, analysis and have led to the development of a simple and easy to use decision support matrix for choosing Life Cycle Model on software development projects, and discussed in the following sections VII and VIII.

VII. DISCUSSION ON OUR RESEARCH FINDINGS

Summary of responses captured in survey of practitioners and project team members is tabulated in Table-II.

TABLE-II: SUMMARY OF RESPONSES CAPTURED IN SURVEY OF PRACTITIONERS

Options articulated	No. of Responses	% of Responses
Option-1	01	02 %
Option-2	03	05 %
Option-3	48	81 %
Option-4	05	08 %
Option-5	02	03 %

Total responses → 59

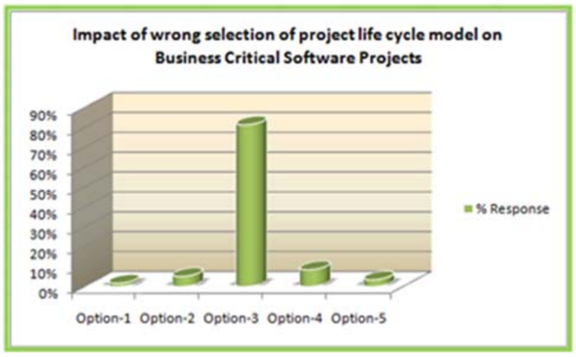


Figure-7: Graphical representation of survey results

A. From the practitioner's survey results, it is crystal clear that wrong selection of project's life cycle model can be detrimental to project's outcome. This calls for having a defined process of right selection of life cycle model for business critical software projects, so that chances of wrong selection of SDLC-Model on projects are minimized.
B. During the course of subsequent detailed study of 11-real-life, global software development / implementation projects, existing methodology of life cycle selection was thoroughly examined with a view of understanding the possibility of human error. The results of this study are as follows:

1. *Results of detailed study from startup IT-companies*:
    a. On 5-Projects studied from startups, SDLC-selection decision on all 5-projects was found to have been made based on specific individuals (expert technical person) choice.
    b. There was no validation of the experts' opinion and the selection was based purely on analogy of the expert available in the organization, at that point in time – when the project was initiated.
    c. In a volatile market of IT-business (where star performers may leave anytime for a better job), this poses a huge risk for the organization.
    d. No documentation of process for SDLC-decision was claimed.
    e. There is a high possibility of human error on SDLC-selection decision.

2. *Results of detailed study from SMBs:*
    a. On 4-Projects studied from SMBs, SDLC-selection decision was claimed to have been made based using documented process.
    b. Documented evidence of SDLC-selection process was unavailable on all of the 4 projects.
    c. Different team members had differing opinion and understanding of SDLC-selection process.
    d. There is a possibility of human error on SDLC-selection decision

3. *Results of detailed study from Established IT-companies:*
    a. On 2-Projects studied from startups, SDLC-selection decision was claimed to have been made based on defined and documented process.
    b. The evidence of documented process shared was quite subjective and liable to have multiple interpretations. and chances of errors.
    c. There is still a possibility of human error on SDLC-selection decision.

4. There is a need to have a simple and easy to use decision support tool to help with selection of right SDLC-model for business critical software development projects.

C. A simple and easy to use decision support tool, proposed by authors, for Extended SDLC-selection decisions covering 3-traditional SDLC-Models (Waterfall, V-process and Incremental/Iterative) and 2-agile SDLC-Models (XP and RUP) is articulated below in Table-III:

TABLE-III: DECISION SUPPORT MATRIX FOR SDLC MODEL-SELECTION

Sr. No.	**Evaluation Criteria**	**Traditional SDLC-Models**			**Agile SDLC-Models**	
		V-Process	***2-I (Incremental / Iterative)***	***Water-fall***	***XP***	***RUP***
1.	Business Criticality	High	Medium	Low	Low	High
2.	Customer Involvement through Life Cycle	Low	Medium	Low	High	Low
3.	Requirements Clarity	High	Medium	High	Low	Medium
4.	Requirements Volatility	Low	High	Low	High	Medium
5.	Availability of business users	Medium	High (& tapers to Medium)	Low	Medium	High (& tapers to Medium)

Sr. No.	Evaluation Criteria	Traditional SDLC-Models			Agile SDLC-Models	
		V-Process	*2-I (Incremental / Iterative)*	*Water-fall*	*XP*	*RUP*
6.	Project Size	Large	Large	Small	Medium	Large
7.	Complexity (Business, Technical)	Low, Medium	High, High	Low, Low	Medium, High	High, Medium

VIII. CONCLUSION

(i) An important finding of this study lies in identification of possibility of wrong selection of life cycle models on software projects, its consequent impacts on project's success.

(ii) Another important finding of this study is unearthing the need to have an easy to use decision support tool to help with objectively select right life cycle model for business critical software projects.

(iii) This study has also resulted into a valuable contribution and creation of prescriptive guidance for business critical software projects, by extending a previously proposed decision support matrix for making a right choice of SDLC-models to include 2-Agile Models (XP and RUP) also, over and above the existing 3-traditional models (viz. Waterfall Model, V-Process Model and Incremental/Iterative Model).

IX. IMPLICATIONS FOR FURTHER RESEARCH

This research is a step forward towards further understanding of the impact of SDLC-selection decisions on project objectives of business critical software projects. This research has focused on focused on detailed assessment of SDLC-selection methods used on a number of challenged projects to gain an understanding of existing practices and recommended appropriate selection matrix for traditional and agile methodologies. Planned extensions of work, in this regard could be extending it further to include other agile and traditional models, and industry case studies on development of organization project delivery frameworks using a blend of traditional and/or light weight SDLC-methodologies. Another direction of work could be related to challenges involved in migrating from traditional to light-weight methodologies.

ACKNOWLEDGEMENT

Authors would like to thank all the project managers and practitioners from Global IT-Industry who participated in the academic survey conducted for this study. Also, special thanks to senior project managers and team members of the projects selected for detailed study for their cooperation, support and sparing their time, providing their valuable inputs as and when required during the course of detailed study, including semi-structured interviews that were conducted and iterated to weed-out the results. Many thanks to all the peers, colleagues and organizations who supported this research by sharing their valuable time and feedback on experiences in project life cycle models.

REFERENCES

[1] Lindvall, M., & Rus, I. "*Process diversity in software development*", IEEE Software, *17*(4), 14-18, July/August 2000.

[2] Rothi, J., & Yen, D. "*System Analysis and Design in End User Developed Applications*", Journal of Information Systems Education, 1989. viewed on-line: http://www.gise.org/JISE/Vol1-5/SYSTEMAN.htm

[3] Yourdon, E. "*The Emergence of "Light" Development Methodologies*", Software Productivity Center, October 2000. viewed on-line: http://www.spc.ca/resources/essentials/oct1800.htm#3

[4] Ronald G. Wolak, "*DISS 725 – System Development: Research Paper 1 SDLC on a Diet*" School of Computer and Information Sciences, Nova Southeastern University, April 2001

[5] DOJ, The Department of Justice Systems Development Life Cycle Guidance Document, viewed on http://www.usdoj.gov/ jmd/irm/lifecycle/table.htm, January 2003.

[6] S. Cohen, D. Dori, U. de Haan, "*A Software System Development Life Cycle Model for Improved Stakeholders' Communication and Collaboration*", Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844, Vol. 5, No. 1, pp. 20-41, 2010.

[7] NIH System Development Life Cycle (SDLC) IT Security Activities Matrix, viewed on-line: http://irm.cit.nih.gov/ security/nih-sdlc.html, December 2006.

[8] Deloitte report 2009–"*Global economic slowdown and its impact on the Indian IT industry,* viewed on-line at

http://www.deloittemeet.com/files/IT%20Recession.pdf , July, 2010.

[9] Standish report - *CHAOS Summary 2009*, viewed on-line: http://www1.standishgroup.com/newsroom/chaos_2009.php , July 2010.

[10] H.J. Thamhain and D. I. Wilemon, "*Conflict Management in Project Life Cycles*" Sloan Management Review, vol. 17, pp. 31-50, 1975.

[11] J.K. Pinto and J.E. Prescott, "*Variations In Critical Success Factors Over The Stages In The Project Life Cycle*" Journal of Management, vol. 14, pp. 5-18, 1988.

[12] J.K. Pinto and S.J. Mantel, "*The Causes of Project Failure*" IEEE Transactions of Engineering Management, vol. 37, No. 4, pp. 269-276, 1990.

[13] The Stanford University IT-Services Process documentation Website, viewed on-line: http://www.stanford.edu/dept/its/projects/PMO/files/our_process.html

[14] A guide to the Project Management Body of Knowledge, Fourth Edition, from PMI – World's Leading Professional Association for Project Management Profession (http://www.pmi.org).

[15] P.M. Khan and M.M.S. Beg, "*Project Management Office(PMO): An Organizational Asset for Value Propositions in Software Development Projects* " Proc. International Conference on Emerging Trends in Engineering and Technology, College of Engineering, T. M. University, Moradabad, India, April 6-7, 2012.

[16] The Rules of Extreme Programming documentation, viewed on-line: http://www.extremeprogramming.org/rules.html

[17] The Rational Unified Process Documentation, viewed on-line from Object Mentor web-site: http://www.objectmentor.com/resources/articles/RUPvsXP.pdf , May 2012.

[18] P. M. Khan, M. M. S. Beg, "*Kicking Off Risk Management for Software Projects*" International Journal of Software Engineering and Computing (IJSEC), vol. 3, no. 2, July-December 2011, International Science Press, ISSN: 2229-7413, pp. 95 – 99.

[19] P.M. Khan, M.M.S. Beg, "*Measuring Cost of Quality (CoQ) on SDLC Projects is Indispensible for Effective Software Quality Assurances*" International Journal of Soft Computing and Software Engineering (JSCSE), vol. 2, no. 9, 2012, e-ISSN: 2251-7545.

[20] P.M. Khan, M.M.S. Beg, "*Application of Sizing Estimation Techniques for Business Critical Software Project Management*" International Journal of Soft Computing and Software Engineering [JSCSE], ISSN: 2251-7545, Accepted for publication in 2013, (Article in Press).

[21] P.M. Khan, M.M.S. Beg, "*Decision Support Matrix for Selection of Life Cycle Model on SDLC-Projects*" Proc. 6th International Conference on Advanced Computing & Communication Technologies, ICACCT 2012, ISBN: 9789382062677, vol-1, pp. 81-87, Nov 2012.