International Journal of Computer Science and Software Engineering (IJCSSE), Volume 4, Issue 1, January 2015
ISSN (Online): 2409-4285 www.IJCSSE.org Page: 22-28

Proposed a Structured Framework for Enhancing Software Projects Quality

Prof. Dr. Adel Darwish[1], Assoc. Prof. Dr. Ayman E. Khedr[2], Dr. Iman Asaad Badr[3], Mr. Ahmed Fouad Omran[4]

[1] Department of Mathematics, Faculty of Science, Helwan University,Cairo, Egypt
[2] Faculty of Computers and Information Technology, Information Systems Department, Future University in Egypt
[3, 4] Department of Computer science, Faculty of Science, Helwan University, Cairo, Egypt

[1]profdarwish@yahoo.com, [2]ayman.khedr@fue.edu.eg, [3]imanb@aucegypt.edu, [4]ahmedfouadomran@gmail.com

ABSTRACT
Although there are large numbers of software quality models are applied, there are still serious problems in the production of the software industry which leads to an increase in cost and time required producing software. Therefore, it is necessary to have a new approach to measure software product quality. The International Organization for Standardization (ISO 9126) that uses to evaluate the software quality model defines the standard of software quality that defines six characters is divided into twenty-one sub-character. This study is attempted to insure that the success of applied integrated model in software product by measuring software product by ISO 9126 software quality model. Accordingly and based on the research findings, the study found out that the integrated model still need some improvement to be measured by the order are privacy accordance with the requirement of ISO standard.

Keywords: *Software Quality, Software Development Projects, ISO 9126 Software Quality Model, Software Evaluation.*

1. INTRODUCTION

Software quality is define as a software must conformance to explicit and implicit requirement the conformance to explicit documented development standard and conformance to implicit characteristic that are expected of all professionally develop software [10].With the development of the software industry and the emergence of many problems in software quality So ask this measure software quality. Meanwhile, Software evaluation is a type of assessment that seeks to determine if software or a combination of software programs is the best possible adequate for the needs of a given client [9]. The clue is to look closely at the resources and tools provided by the software that is either currently in use or is being examined as a possible addition to programs already in use by that client. Based on a prepared list of criteria along with some practical experimentation, a software evaluation makes it possible to determine, either if the products would be helpful to the client, or if some other combination of software products would serve to better advantage [19].

The main aim of the enhanced integrated model is to evaluate the software project that assesses the various software methodologies, which were used throughout the development of the framework, the accuracy of the estimations, and the usefulness of the reviews. The product will be reviewed and evaluated to check; either it accomplishes the ideas presented in the initial overview, or the quality of the product.

2. BACKGROUND AND RELATED WORK

Early studies, dealt with some notions concerning the research proposed topic. In 1984, Jenkins made a survey study to address some problems related to system development project. The study specified that, the developers of 72 information system development projects in 23 major American's corporations were interviewed; the average cost was 36% and the schedule overrun was 22%. Finally, Jenkins proved that the cost and time overrun considered being the most serious problems facing the project development. While, successful project planning mostly relies on decent estimations of cost and closed monitoring of development process [6].

The Standish Group Report (CHAOS) (2012) results showed an increase in projects success rates, with 39% of all projects succeeding, 43% were faced critical challenges and the other 18% are failed. These numbers represent an uptick in the success rates from the previous studies, as well as a decrease in the number of failures. As in 2004, only 29 % of the projects were successful, and in 2012, the results represent a high watermark for success rates in the history of CHAOS research. The following table shows the Project resolution results from CHAOS research from 2004 to 2012 [14, 18].

Table 1: The Resolution Results From CHAOS Research From 2004 to 2012.

projects status	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

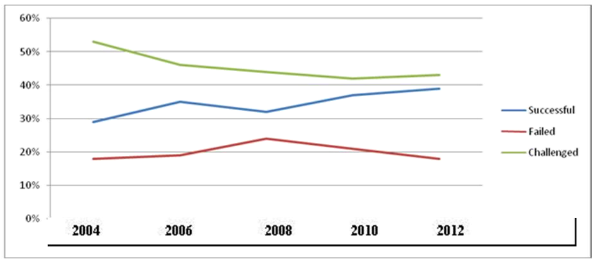


Fig. 1. The changes of the projects successful failed and challenged from CHAOS research (2004 to 2012). Source: The CHAOS

3. RESEARCH PROBLEM

From the previous projects and statistical analysis, there is one fact of software projects fails without other fields of technology. This failure occurs in small, medium-sized companies and international organizations and there are several reasons for the failure of software and we will summarize the reasons.
The failure of software projects was due to the following reasons [7, 20]:

- Underestimation of effort and /resources required: in order to determine required resources, we must estimate the size of an application working progress that must be monitored, thus it will overcome all expected problems to accomplish the project plan.
- Miscommunication among project staff: project staff works in different departments and there is no detected time for meeting with project staff, and this makes everyone works individually outside the general framework of the project.
- Project management: project manager facing some different problems during the project such as, staff problems, budget constraints and time schedule, while the project manager is responsible of controlling the entire problems mentioned and find quick solutions to them.

The main research problem is some missing important functions: that due to consequence changing customer requirements during the project development and testing Stages, which were not mentioned during the requirement stage, thus lead to loss of time and effectiveness of the project budget.

4. RESEARCH METHODOLOGY

The software development projects consist of a series of phases; each phase has internal main factors which namely are tasks, team members, and reporting, as depicted in

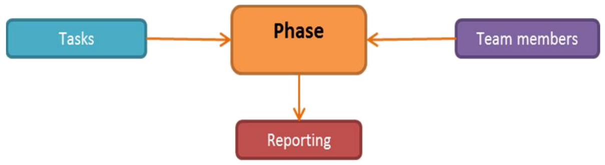


figure 2. The following discussion illustrates these phases

Fig. 2. The Internal phase main factor

These phases are [13]: 1) The project plan phase this phase is the first phase in the software projects It involves creating of a set of plans to help guide your team through the execution and closure phases of the project [17]. 2) The requirement Specification this phase is the second phase in the software development projects, It Gathering and agreeing on requirements is fundamental to a successful project [12]. 3) Design phase the system is designed to satisfy the requirements identified in the previous phases [21]. 4) The Development Phase features a key step in the project, system construction and the previous phases lay the foundation for system development [23]. 5) Testing and integration phase, this phase makes sure that tests such as the system and integration are done before releasing the product[11] . 6) Deployment phase which is the final stage of releasing an application for users (the Installation stage was replaced by the Deployment phase because; it has to be always up-to-date of the project software) [22]. But in this we added new stage to software projects which is the evaluation stage. 7) This phase evaluates each stage of the software project stages independent. The software development project identifies seven stages; each stage contains a set of objectives [5]. Figure 3 shows the seven stages and their objectives.

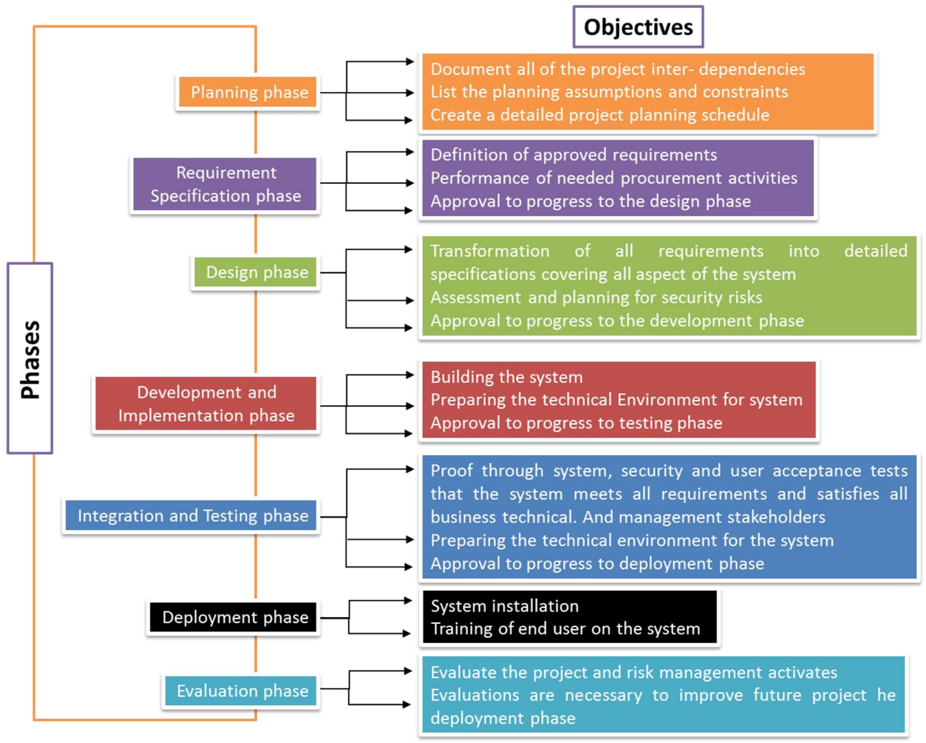

Objectives
Phases
Planning phase
Document all of the project inter- dependencies
List the planning assumptions and constraints
Create a detailed project planning schedule
Requirement Specification phase
Definition of approved requirements
Performance of needed procurement activities
Approval to progress to the design phase
Design phase
Transformation of all requirements into detailed specifications covering all aspect of the system
Assessment and planning for security risks
Approval to progress to the development phase
Development and Implementation phase
Building the system
Preparing the technical Environment for system
Approval to progress to testing phase
Integration and Testing phase
Proof through system, security and user acceptance tests that the system meets all requirements and satisfies all business technical. And management stakeholders
Preparing the technical environment for the system
Approval to progress to deployment phase
Deployment phase
System installation
Training of end user on the system
Evaluation phase
Evaluate the project and risk management activates
Evaluations are necessary to improve future project he deployment phase

Fig. 3. The objectives of software development projects phases

4. ENHANCED INTEGRATED MODEL

The early integrated model referred to detect the defection in each stage of software development project before entering the next stage [4]. This integrated model, as depicted in figure 4, is based on two methods namely, inspection (static test) and testing (dynamic test), where we apply the inspection on the first three stages then apply the testing on the other three stages of the software development projects. The three stages of the first phase of software development projects are project plan (inspect project plan documented and plan schedule) [3], requirement Specification (inspect Software Requirement Documented (SRD) [16], and design (inspect the Architecture Design Documented (ADD) [15], respectively (static test).The static test depends on inspecting and reviewing, after finishing the Process of these stages, the integrated model prepares the report to ensure the end of the first Phase of the model, while the other three stages of the second phase of the model contains the development, integration testing, and installation, respectively (dynamic test). The development stage will check the code review of the project [23]. While the integration testing stage performs the test cases, function testing, and applying the user acceptance test [8]. Finally, the deployment stage which the software will be installed according to the production environment [22]. Therefore, the software inspection (static test) will be used in the first phase of the model and the software testing (dynamic test) will be used in the other phase of the model.

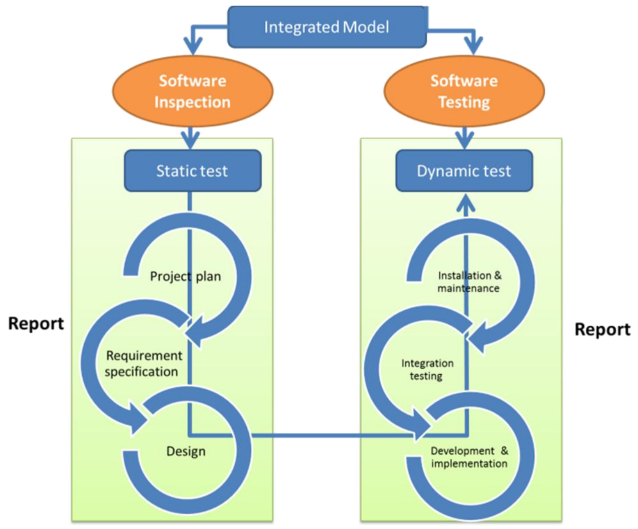


Fig.4: The integrated model for software development projects

Source: Adapted from Proposed an integrated model to detect the defect in software development projects, [4] The enhanced integrated model, improves the previous integrated model by adding a new stage which is the evaluation stage, by which it evaluates the different stages of the integrated model (planning, requirement, design, development, integration testing and deployment stages), as shown in the following figure.

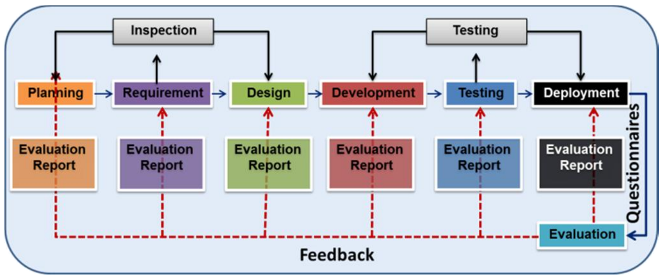


Fig. 5. The research enhanced integrated mode for software development projects

Figure 5 shows the enhanced integrated model according to the evaluation stage which was added to enhance the problems of the previous integrated model. As the integrated model was divided in to two main parts the inspection phase which includes the planning, requirement and the design phases, and the testing phase which includes the development, testing and installation phases, the last phase was changed to be the deployment phase as it expresses the up to date version of software in the production environment. The evaluation stage was added as shown in the previous figure in each stage individually. While, it is recommended to apply the evaluation phase not only by each stage but also for the whole model to evaluate the success of the integrated model after enhancement.

5. RESEARCH CASE STUDY

The empirical case study was chosen according to the analysis of the problems in the existing software of an Islamic bank in Cairo, Egypt. The researcher is working in the software quality department, in the Information Technology Sector, since 2008. The bank consists of nine sectors and includes about 1600 employees, with thirty branches spreading around Egypt. As the integrated model framework will be applied to a new software project in the bank. The bank staff found that there are many lacks in some functions, causing the appearance of problems for end users. Therefore, the researcher looks at these problems show that there are some problems and challenges that the integrated model cannot deal with it and to overcome these problems, the researcher suggested the addition of a new phase, a phase of evaluation where they are evaluating all stages of the integrated model in addition to the evaluation of the entire project by the questionnaire where they were the work of this questionnaire on the basis of criteria ISO 9126-1 Software Quality Characteristics that identifies six main quality characteristics; namely are: Functionality, Reliability, Usability, Efficiency, Maintainability and Portability, These characteristics are divided into sub characteristics [1] and two general questions as follows:

A. Functionality

The functionality characteristic is the main purpose of any product or service how software could provide the required functions. The Specific question is "Software satisfies functional requirements?":

- Suitability: The capability of the software to provide an adequate set of functions for specified tasks and user objectives.
- Accuracy: The capability of the software to provide the right or agreed-upon results or effects.
- Interoperability: The capability of the software to interact with one or more specified systems.
- Security: The capability of the software to prevent unintended access and resist deliberate attacks intended to gain unauthorized access to confidential information.
- Compliance: Where appropriate certain industry (or government) laws and guidelines need to be complied with, i.e. SOX. This sub-characteristic addresses the compliant capability of software.

B. Reliability

The reliability characteristic is ability of software maintains the level of system performance when used under specified conditions. The specific question is "Software ability to recover, continue functioning and handling the failure?".

- Maturity: The capability of the software to avoid failure as a result of faults in the software.
- Fault Tolerance: The capability of the software to maintain a specified level of performance in case of software faults or of infringement of its specified interface.
- Recoverability: The capability of the software to reestablish its level of performance and recover the data directly affected in the case of a failure.

C. Usability

The usability characteristic refers to the ease of use software could be understood, learned, used and liked by the developer. The specific question is "Software user friendly?".

- Understandability: The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.
- Learnability: The capability of the software product to enable the user to learn its applications.
- Operability: The capability of the software product to enable the user to operate and control it.
- Attractiveness: The capability of the software product to be liked by the user.

D. Efficiency

The efficiency characteristic is concerned with the system resources software provides required performance relative to amount of resources used .The Specific question "Performance of software meets your expectations concerning resources?".

- Time Behavior: The capability of the software to provide appropriate response and processing times and throughput rates when performing its function under stated conditions.
- Resource Behavior: The capability of the software to use appropriate resources in an appropriate time when the software performs its function under stated condition.

E. Maintainability

The maintainability characteristic is identifying and fixing a fault within a software component is what the maintain ability software .The specific question is "Software ability to easy identify the cause of failure and who cause this failure"?.

- Analyzability: The capability of the software product to be diagnosed for deficiencies or causes of failures in the software or for the parts to be modified to be identified.
- Changeability: The capability of the software product to enable a specified modification to be implemented.
- Stability: The capability of the software to minimize unexpected effects from modifications of the software.
- Testability: The capability of the software product to enable modified software to be validated.

F. Portability

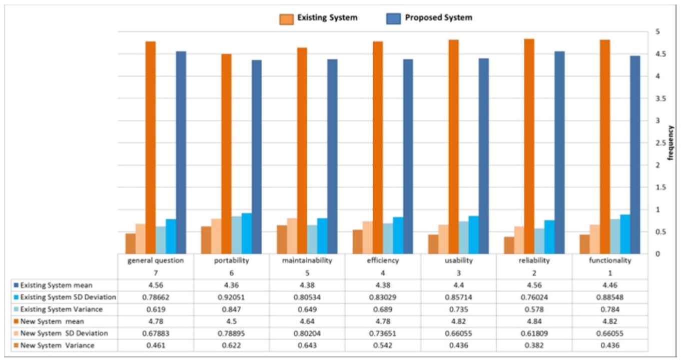
The portability characteristic referring to the software provides required performance relative to amount of resources used and ability to adopt at any environment. The specific question is "Software adaptive to new system changes without any problems and ability exchange components within specified environment?".

- Adaptability: The capability of the software to be modified for different specified environments without applying actions or means other than those provided for this purpose for the software considered.
- Install ability: The capability of the software to be installed in a specified environment.
- Coexistence: The capability of the software to coexist with other independent software in a common environment sharing common resources.
- Replace ability: The capability of the software to be used in place of other specified software in the environment of that software.

Case study analysis:

Questionnaires used in different forms and used to know opinion about the following: for customer satisfaction, product registration, and new product development, academic research. the researcher in this paper used questionnaire to know opinion the user about the proposed system in bank, This questionnaire consists of 18 questions were distributed to 50 users and we use software SPSS v.19 quantitative techniques to provide statistics analysis for [2]:
Mean: use measures of center to summarize the data and to impart information in a concise way about distributions and very sensitive to changes in the data.
Standard Deviation: measures the amount of variation or dispersion from the average. A low standard deviation indicates that the data points tend to be very close to the mean (also called expected value); a high standard deviation indicates that the data points are spread out over a large range of values
Variance: measures how far a set of numbers is spread out. A variance of zero indicates that all the values are identical. Variance is always non-negative: a small variance indicates that the data points tend to be very close to the mean (expected value) and hence to each other, while a high variance indicates that the data points are very spread out around the mean and from each other.
Chi-Square test: The test is applied when you have two categorical variables from a single population. It is used to determine whether there is a significant association between the two variables.

The findings of questionnaire analysis are shown in figure 6.



	general question	portability	maintainability	efficiency	usability	reliability	functionality
	7	6	5	4	3	2	1
Existing System mean	4.56	4.36	4.38	4.38	4.4	4.56	4.46
Existing System SD Deviation	0.78662	0.92051	0.80534	0.83029	0.85714	0.76024	0.88548
Existing System Variance	0.619	0.847	0.649	0.689	0.735	0.578	0.784
New System mean	4.78	4.5	4.64	4.78	4.82	4.84	4.82
New System SD Deviation	0.67883	0.78895	0.80204	0.73651	0.66055	0.61809	0.66055
New System Variance	0.461	0.622	0.643	0.542	0.436	0.382	0.436

Fig. 6. the comparison findings between the existing system and the proposed system quality in software development projects in Islamic bank.

The CHI-Square test between every two variables, are supposed that:

(H_0), means that no relation between the existing system and the proposed system

(H_1), means that there is a relation between the existing system and the proposed system

Table 2: CHI-SQUARE Test Results

Chi-Square	Value	Df.	Asymp. Sig. (2-sided)
Pearson Chi-Square	185.943[a]	50	.000
Likelihood Ratio	73.018	50	.019
Linear-by-Linear Association	35.336	1	.000
N of Valid Cases	50		

a. 64 cells (97.0%) have expected count less than 5. The minimum expected count is .02

6. FINDINGS ANALYSIS AND DISCUSSION

So, after adopting the enhanced Integrated Model of software development projects, accordingly, the results of the questionnaire showed that,

- The functionality of the proposed system was better than the existing system by 7.2 %.
- The reliability of the proposed system was better than the existing system by 5.6 %.
- The usability of the proposed system was better than the existing system by 8.4 %.
- The efficiency of the proposed system was better than the existing system by 8 %.
- The maintainability of the proposed system was better than the existing system by 5.2 %.
- The portability of the proposed system was better than the existing system by 2.8 %.
- The general question of the proposed system was better than the existing system by 4.4 %.

Accordingly, table 2 showed that the CHI-Square=185.943, and the SIG variable = 0.000 and this value is less than 0.05 which means that there is a statistical relation between the existing system and the proposed system.
Finally, based on the preceding statistical analysis, the project was completed and delivered on schedule and budget detected. The software project was eventually successful. It is not because there were no problems during the software development project phases but because the problems have been found in each phase and overcome before causing serious deviation. The major task of this framework is detecting the defect in each phase of the project. Once phase finished us product report to explain where you are and go throw to next phase. These arrangements of project phases helped to detect the defect early, which lead to the success of the project and reach customer's satisfaction for the requirements.
To conclude, in general the proposed enhanced system is more satisfying and significance for the software project.

7. CONCLUSIONS

This study showed that there are many ways to reduce the failure of projects, development of software and disclosure of types of modern methods have failed, but it turned out after research that the emergence of new problems; the more successful of these methods and techniques to address these problems showed other reasons which lead to the failure of the projects and this in turn, one of the most important reasons that faces the industry.
In the field of research, it was clear that, early detection of the defect is one of the most important reasons for the success of the software projects, as it clearing the defect and in order to evaluate the project phases. It is a must to evaluate the project after each phase so that to be able to maintain the quality of the project.
The Researcher thinks that the integrated model framework can be succeeded in maintaining costs and time of software projects in general. Thus, the Application of the integrated model framework will improve quality of the software development projects.

REFERENCES

[1] Damico S. Nicome, 2010, “THE NEW ERA OF SOFTWARE QUALITY ASSURANCE”, Organizational Design for Measurement, University of New York.

[2] David M. Lane, Mikki Hebl, etl., "Introduction to Statistics", Online Edition, Rice University, University of Houston, Downtown Campus.

[3]Emanuele Della Valle, 2011 – 2012, Planning Phase: Planning and Managing Software Projects. Politecnico di Milano, Italy.

[4] Ayman E. Khedr, Omran Fouad Ahmed, 2012, “Proposed an Integrated Model to Detect the Defect in Software Development Projects", International Journal of Software Engineering (IJSE), Vol. (3), pp. 55-56.

[5]Felipe Ortega, Daniel Izquierdo, Pedro Coca, 2010, “Introduction to software quality evaluation”, University of Rey Juan Carlos.

[6] Ho Leung Tsoi, 1999, “A Management Framework for Software Project Development”, Software Quality Institute, Griffith University, pp. 1-2.

[7] Haughey, D., PMP, 2000-2011, Why software projects fail and how to make them succeed, project smart, pp.1-2.

[8] Mauro Pezzo, Michael Young, 2007, "Software Testing and Analysis", Integration and Component Based Software Testing, Chapter 21, pp. 1-10.

[9] Mike Jackson, Steve Crouch and Rob Baxter, 2011, "Software Sustainability Institute. Software Evaluation: Criteria-Based Assessment".

[10]Roger S. Pressman, 2007, "Software Engineering: A Practitioner’s Approach", Edition 6th Chapter 15, Product Metrics for Software, pp.1-10.

[11]Stuart Anderson, 2011, Integration Testing, school of informatics.

[12]Stuart R. Faulk, 1997, “Software Requirements Engineering”, 2nd R. Thayer. M. Dorfman, Eds., IEEE Computer Society press.

[13]The software development life cycle (SDLC) for database applications, 2000 – 2005, Digital publications, version pp. 1.1c.

[14] Tom Clancy, 2014, The Standish Group CHAOS Report, Project Smart.

[15]ESA Board for Software Standardization and Control (BSSC), “Guide to the software architectural design Phase,”PARIS CEDEX, France, Issue 1 Revision 1 March 1995.

[16]ESA Board for Software Standardization and Control (BSSC), “Guide to the software requirements definition,” PARIS CEDEX, France, Issue 1 Revision 1 March 1995.

[17]Roy Berkeveld, Gijs Direks,etl., “Software Project Management Plan”, 2009, University of technology, Eindhoven, the Netherlands.

[18]The Standish Group Report, CHAOS MANIFESTO, 2013.

[19]Basic Guide to Program Evaluation (Including Outcomes Evaluation), [Online]. Available: http://managementhelp.org/evaluation/program-evaluation-guide.htm, access date 15 March 2014.

[20]Cohen Shwartz Oren, D, 2002, "Why Software Projects tend to fail", Available: http://www.codeproject.com/Articles/20488/Why-Software-Projects-Tend-to-Fail, access date 12 November 2013.

[21]Daniel Davis, 2013, "the Design of Software Engineering", Chapter 3 [Online], Available: http://www.danieldavis.com/thesis-ch3/, access date 12 December 2013.

[22]Robert Dobrzycki, SDLC: Software deployment phase, Available:robertdblog.wordpress.com/sdlc-software-deployment-phase, access date 15 January 2014.

[23]Single Release Custom, PHASE 6: DEVELOPMENT PHASE. [Online]. Available:http://doit.maryland.gov/SDLC/Documents/SDLC%20Phase%2006%20Development%20Phase%20Single%20Custom.pdf, access date 22 N.