*International Journal of Engineering & Technology*, *7 (1.1) (2018) 314-318*

**International Journal of Engineering & Technology**

*Website: www.sciencepubco.com/index.php/IJET*

***Research paper***

# A study on risk assessment techniques in information systems

**D. Nagamalleswari [1], J. Nagalakshmi [2] *, G. Karthik [2], P. Harthita [2]**

[1] *Professor Department of Computer Science and Engineering Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India-522502*
[2] *Student Department of Computer Science and Engineering Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India-522502*
**Corresponding author E-mail: nagalakshmi3045@gmail.com*

**Abstract**

In today's world, IT industry is rushing forward with an advancement of developing the advanced software. The Developers always try to develop the software projects without any errors or failures. Even though the developers take many measures to avoid software project failures, they are facing the failures that are occurred due to Risks that take place in the software projects. We cannot remove risk completely to the 100% extent, but we can try to minimize the risk in the projects by assessing the risks. So, in this paper we are providing a survey which overviews on different risk assessment techniques. This survey provides information about various risk assessment techniques which will be further useful for the software developers to minimize the risk and make the successful project.

***Keywords***: *Assessment; Failures; Risk; RiskManagement; Techniques.*

## 1. Introduction

Software projects will be perfect when they are assessed with risks. Risk based software project or any other information systems may lead to the unsuccessful results. So in order to get the perfect outcome risk should be assessed based on the type of project or an information system. Risk may happen in all type of system like electronics, networks etc. based on the type of action. This paper gives information and role of risk in the information system. It also explains the type of risks that happen in the information system and make the project fail. Many of them proved that, to get a software project risk free, is only possible through assessing the risks. To assess the risk we need to know completely about risk management. So, here we are discussing and surveying about software Risk Management and steps involved in it, how the risk is assessed and what are the risk assessment techniques.

### 1.1. Definition of risk

Risk can be defined as an unexpected interruption taken place which leads to failure of the project. According to ISO Guide 73 ISO 31000 [23], Risk is defined as "Effect of uncertainty on objectives. Note that an effect may be positive, negative, or a deviation from the expected result. Also, risk is often described by an event, a change in circumstances or consequences". Institute of Risk Management (IRM) [23] defined risk as "Risk is the combination of the probability of an event and its consequence. Consequences can range from positive to negative. The Institute of Internal Auditors (IIA) defines risk as the uncertainty of an event occurring that could have an impact on the achievement of objectives [23]. Risk can produce any type of results either positive or negative or may leads to the uncertainty. Guide 73 [23], definitions of risks states that risks are classified into three sections. They are: (i) hazard (or pure) risks (ii) control (or uncertainty) risks. (iii) Opportunity (or speculative) risks.

## 2. Risk management

### 2.1. Risk Management is always a continuous process

Risk Management is well defined as the combination of risk containment and risk mitigation [1].Unless project managers take appropriate measures, Risks may cause adverse effects in the software projects. Risk Management have two main steps, they are Risk control and Risk Assessment, each are alienated into three interior stages [2]. Risk assessment has Risk Identification, Risk Analysis and Risk Prioritization. Risk Control will have Risk Management Planning, Risk Resolution and Risk Monitoring. Risk identification gives project-specific risk items list that they may reduce the successful outcomes in the project. Many more lists of risks which have possibility to occur in software projects have been described [3] [4] [5], and they are classified based on their effects [6]-[10].

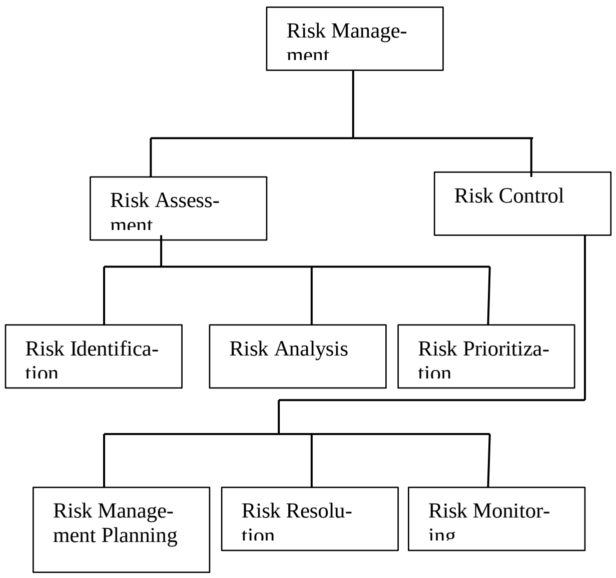

**Fig. 1:** Steps in Risk Management.

Risk management planning defines provides the description about the activities of the risk reduction regards the importance of the process [11], organization [8] [12], and technology [13].The information about the elimination of the risk items are provided by the Risk Resolution. The tracing of tracks development of the project towards risk items resolve is done by Risk Monitoring. By this we may observe that the Risk Management is a process which is continuous. Software development is the most complicated and incalculable action related with the risks causing the fatal error. With increase in many organizations that are highly supporting for the growth of the software because the risk assessment and risk mitigation becomes the vital [14]. The Aihua Yan [14] tries to question the following:

1) In practice the way How to apply risk management?
2) What are the different risk management strategies considered?
3) What is the role of risk management in software development?
4) What is the process of risk management?

## 2.2. Strategies in risk management

From the analysis older researches risk management strategies are classified into four types they are
Risk -action list, risk-strategy analysis, risk-list, risk-strategy model. From the dynamic system theory a continuous process is introduced and also the author introduces a framework for the application of management of risk [14]. The Aihua Yan [14] tries to explain the approaches of Risk Management. Risk Management has the following approaches [14]

1) Risk -action list.
2) Risk-strategy analysis.
3) Risk-list.
4) Risk-strategy model.

### 2.2.1. Risk action list

The record of the prioritized risk with the corresponding resolution actions [25].

### 2.2.2. Risk strategy analysis

Risk-strategy analysis makes a sense that it is a stepwise process. It also provides the in depth details of the risk [25].
This model shows the similarities like the risk-strategy model. This also provides percentage of resolution actions and risk items but, it offers different application of trial-and-error techniques. These two methods differs at the point that there is no model coupling aggregate risk elements to aggregate resolution actions [25]. Customers, Developers, Managers are involved in this process that merges all the risks to actions in order to develop a complete risk strategy.

### 2.2.3. Risk-list

This approach gives the detailed Prioritized Risk items list [25].This list of prioritized risk items might help the project managers only concentrate on the possible risk source. This risk list is most useful for risk assessment [25].

### 2.2.4. Risk-strategy-model

This approach is also called as the contingency model which is used to link the risk events to percentage of resolution actions [25]. Basically, this approach mainly hides the types of risks in it and form a profile for risk. Abstracting this is the first task and then abstracts the action categories and then makes the overall risk strategy [25]. This helps us to classify the project into a category. According to that category, this approach helps us to provide detailed resolution action. The author made a detailed survey to prove that risk management is a continuous process [14]. Here is a continuous process of risk management. Aihua Yan [14] proposed a practical approach based on the Iversen et al [14]. This is modified model for the Iversen Risk assessment model. In order to improve the Software performance Iversen et al developed a risk assessment model by using the action research. The following model is modified by authors [25] [14].
From risk management in software development by Aihua Yan [14] defines the risk in two ways, one is in qualitative and other is in quantitative. The qualitative definition of risk by author is that extent of risk as the project uncertainty and the project failures with potential loss [24].The other one is quantitative, and the risk in the form of quantitative was defined:

$$RE = Prob\ (UO) * Loss\ (UO)$$

Loss (UO) - loss to the parties affected if the result is unsatisfactory
Prob (UO) - probability of an unsatisfactory outcome, and RE refers to risk exposure [14].

## 2.3. Advantages of risk management

There are many advantages by using risk management in the software related projects and includes helping the developers to be focused on linking potential threats to possible actions, emphasizing potential causes of failure, providing the combined interface of the project among its members [25] , and problematic aspects. There are many approaches in risk management that are developed to find, analyze, and challenge risks in portfolio of project, requirement risks, system development risks, and implementation risks [25].

## 3. Literature review

In order to improve the Software performance Iversen et al developed a risk assessment model by using the action research. This model is modified by Iversen, Jakob H [25], Aihua Yan [14]. A lesser acknowledged trying out risk-based technique checking out, this considers the failure rate of the source code and decided by means of complexity [15]. The object-oriented application, a new method was introduced for detecting risk. Risk-based testing is one of the procedures makes sure that it examines and fix solutions for the muddle. Risk can be classified into two factors mainly probability of the failure and intensity of the failure event. Risk calculation can be expressed as:

$$Risk = \sum P\ (Ei) * C\ (Ei),\ i=1, 2, 3 \ldots n.$$

N= no. of events failed that are unique [15].

P (E) - probability
C (E) – Event's cost.
Risk-based test mostly concentrates in examining product then determining test design stipulated on the regions well on the way to encounter an issue that would have the most noteworthy effect [16] The cost of the failure event c(Ei) rely upon the idea of the application and is controlled by space investigation. In any case, in this paper seriousness evaluation isn't discussed. So in this technique the essential undertaking is to discover the reasons that may make the product down. This has been demonstrated that code which is more complicated that may lead to more occurrence of blunders and issues [17]. Cyclomatic Complexity is the method or an approach used for determining the complexity of the code [18]. Accordingly, constraints are used to forecast project failure by understanding the defects in the module and then sorting them in accordance with the complexity. Finally using the complexity rankings determining the seriousness of the failure event from the domain and also specifies which module should get highest priority. Yet, module multifaceted nature is a single variable measure; furthermore, we can ignore exceptionally code with risk [19].

## 3.1. Metrics for calculating risk

Further, in this paper some system of measurement were discussed which were identified by The Software Assurance Technology Center (SATC) [15] located at NASA and one of the space center has discovers six metrics.
1) Number of Methods (NOM).
2) Weighted Methods per Class (WMC).
3) Coupling between Objects (CBO).
4) Response for Class (RFC).
5) Combination of RFC/NOM.
6) Number of Children (NOC).

As the metrics are determined, we then need to explain guideline.
When we initially started to apply a portion of the customary measurements to protest situated program, we observed esteems by and they are less acclimated with seeing practically composed program. In light of the early boundaries, the OO program is less complicated and significantly more measured than the non-OO inheritance code. However, due to the distinctive way of the OO framework is constructed, the low numbers were usually ambiguous - overlooking the connections among the class present over [15].
The metrics have some boundaries values which noted below [15].
1) Number of Methods (NOM) [15] $\leq 20$ favored, $\leq 40$ adequate per class [15].
2) Weighted Methods per Class (WMC) [15] $\leq 25$ favored $\leq 40$ satisfactory [15].
3) Response for Class (RFC) [15] $\leq 50$. We have seen many classes with RFC more than 50. On off chance that the RFC is higher, it implies the many-sided quality is expanded and understandability is diminished [15].

If there are more numbers of strategies which are appealed from the class in the form of message, there is high intricacy of the class by muddling, debugging and testing. Developing a class with RFC may lead to some problem due to potential for a progressively outstretching influence.
4) RFC/NOM [15] is less than 5 in C++, and is less than or equal to 10 for Java. This is used for shifting the classes that are indeed of testing extensively. The use of classes in java for every use is more, this allows more for the use of this metric [15].
5) Coupling between Objects (CBO) [15] is less than 5. The classes with more CBO indicate that these are very tough to reuse, maintain and understand. If the CBO rate is more, it is very sensitive to change some of the areas like design and it mays leads to difficulty in maintenance [15]. We can notice easily the class due to low coupling which advances the encapsulation and its standards [15].
6) Depth in Tree greater [[15]] than 5, this indicates the measure that the class probability. DIT of 0 means a root. If the percentage of DIT with 2, 3 is more then there is more chance of reuse [15].
7) Number of Children (NOC) the more prominent is the quantity of the youngsters, greater the probability of despicable deliberation of the parent and requirement for extra testing, however the more prominent the quantity of kids, the more the reuse since legacy is a type of reuse [15]. Until there is no "great", "awful" number for NOC, its esteem ends up noticeably vital when a class is found to have high esteems for different measurements [15]. OO software measurements can be utilized as a part of mix to recognize classes that are well on the way to posture issues for an undertaking. The SATC has utilized the information gathered from a large number of question situated classes to decide an arrangement of benchmarks that are successful in recognizing potential issues. At the point when hazardous classes are likewise recognized by area specialists as basic leads an achievement which undertook, testing can be allotted to relieve chance. Hazard-based test may enable engineers to discover and settle imperative programming issues prior in the test stage [15].

Until here we examined about all the testing methods based on risk. Now we are going to examine about the risk assessment based on source code proposed by Arie van Deursen and Tobias kuipers [21]. This mainly confers about "primary facts" and "secondary facts" for software risk estimation. The facts that are acquired by instinctive analysis of source code of the particular system are primary facts, whereas secondary facts are the facts that are acquired from the people that are going with or going on the system and accessible affirmation. We narrate about both the facts and how they are resolved and how we are connecting the elucidating gap from the unprocessed facts to a brief risk assessment that involves endorsement to reduce the risk. This technique was developed while accomplishing various risk assessments, and is constantly being improvised. These assessments are claimed in a way such that the primary and secondary retrievals are claimed as
Secondary Fact Retrieval: One can examine a system by knowing the data available in the organization. This data is resolved by conducting meetings with the stakeholders, and by analyzing the design documents, affirmation, agreements etc.
Primary Fact Retrieval: Here we examine the program of the system by itself. The source code written in different languages for several different subsystems are incorporated relations with one or other organizations, and resolute definition of the information and code manipulation [21].
The distance between the two reclamations is connected as the outcome of source code analysis is associated to the outcome acquired from the interviews. A software stakeholder have dissimilar perspective of the same system, the outcome that is attained by the analysis of program helps in evaluating the perspectives also recognize if risk is deduced is again a risk or not [21].
Here, Arie van Deursen and Tobias Kuipers [21] gave the basis for the utilization of assessments. Let us discuss some examples that explains the use of assessments.
i) An organization has purchased a benchmark software package. This purchased package doesn't accurately accomplish the needs of the organization. The software producer has been suggested to change or rectify the package in order to accomplish the requirements of the organization. Even the package which is modified is out. The organization has faced some problems to implement that method, and surprised that risk is operating along the structure taking into account that the information of millions of customers is maintained by this software package [21].
ii) Ten years ago the government has a contract out with a huge administrative management system. The price of the contract is too big in the view of the government. To bridge the service cost of this system to other similar systems a standard is maintained. An assessment is done for figuring out the risks in the source code under this standard [21].

## 4. Risk assessments techniques

In this paper, we have many more methods of Risk assessments from 1995. This is collected on the basis of survey which was done by the authors [26]. These models are based on the input characteristics of the problem. So, this may help to get aware of the methods that were existed in the past. This may further help to implement new assessments methods based on these methods. The methods were:

1) The first model is Software metrics data collected propose by Chee at al [26] in 1995, takes the input as the software metrics it is collected in the different stages of the development of the software. The technology used is Probabilistic and Decision analysis which takes influence diagrams and kinds of NN [26]. This method works by using the influence diagrams so that, we can easily regulate the data that is used in problem solving.
2) In 2002, the technique called An Enhanced Neural Network Technique for Software Risk Analysis introduce by Neumann. It takes software metric data as input. It used the analysis of principal component and ANN (artificial neural network).This method works as the method that is used for the categorization of risk. This takes the analysis of principal component for normalization. The risk determination/classification is done by the neural network [26].
3) The approach on Neural Networks for software risk analysis, which is introduced by the Yong et al in 2006. This model takes software risk factors that are collected from interviews as input. This method uses information that is taken from the interviews and created factors for software risk. This method again divided into 4 steps [26].
4) The method called Analyzing Software System Quality Risk Using Bayesian Belief Network introduced by young et al [26] in 2007. This uses factors of Project risk with the help of Delphi method based on early data project in the form of input. The technology used in this method is Bayesian Belief Network, Delphi method. This Technique works by using predicts and BBN, this enhances us to change the software development risks [26].
5) A Risk Assessment Model for software projects by Nogueira et al [26] in 2000. This method uses Complexity metrics and personnel along with Requirements as inputs. This method is the risk assessment that uses different software metrics as technology. This method gives excellent results than other models like Putham and COCOMO [26].

A software risk assessment by source code analysis this is introduced by the W. Eric Wong and Kendra Cooper. This is also a risk assessment technique which is based on source code based. In this method they introduced two models of risk. They are static model and dynamic model [27]. In static model they used the information similar to static structure of code like no.of c-uses and p-uses, decisions, definitions and function calls [27]. The Dynamic model makes use of the code’s dynamic test coverage like p-use, c-use and decision coverage with this they determine the metric value [27]. The metric can also be selected from the either

1) Summation strategy which is the sum of the chosen metrics.
2) Product strategy which takes the product of the metrics that were selected [27]. This equation is given as

$$V * \alpha + F * \beta + D * \gamma + C * \varepsilon + P * \text{P}$$

$\alpha$ , $\beta$ , $\gamma$ , $\varepsilon$, P are weighting factors , V- Definitions, number of function calls- F, number of decisions- D, number of c-uses- C , p-uses- P [27].

In this model they calculates risk index. Risk index calculation is not easy for many more number of lines of code. So they developed a tool called as Risk. Using this tool, they calculated Risk index. This tool gives the count of c-uses, p-uses, definitions, decisions, and function calls of the block of code that is passed into this Risk tool. For an example of the fault the classification is given for both the risk assessments through static model and dynamic model as tabulated below [27].

Table 1 describes the number of tests that were failed and occurrence of each fault of the program. It consists of data for type of fault and subtype along with no.of failed tests and located area of the fault. Table2 describes the Assessment of the risk by static model. This gives us the comparison of two static models named ms1 and ms2. This contains the data of percentage of the high risk functions which is more than faulty function and percentage high risk blocks. Table3 gives us the data regarding risk assessment which are based on the dynamic models. Here, two dynamic models based on the number of test cases that are successful, with the data from percentage of the functions that are contained with high risk comparatively higher than functions that are faulty and percentage of functions that are with high risk which are higher than faulty block.

**Table 1:** Classification of Number of Failed Tests and the Location for Each Fault [27]

| | Fault Classification<br>Fault type | subtype | No.of Failed Tests | Location of Fault |
|---|---|---|---|---|
| F01 | Logic neglected or not correct | Condition test was missed. Missing condition test | 26 | sgramp2n |
| F02 | Logic neglected or not correct | Missing condition test | 16 | sgramp2n |
| F03 | Computational problems | Equation insufficient or incorrect | 36 | mkshex |
| F04 | Logic omitted or incorrect | Forgotten cases or steps | 35 | fixselem |
| F05 | Computational problems | Equation insufficient or incorrect | 32 | seqrothg |
| F06 | Computational problems | Equation insufficient or incorrect | 32 | seqrothg |

**Table 2:** Risk Assessments Based on Static Model [27]

| | Ms1<br>% of functions with higher risk than the faulty function | % of blocks with higher risk than the faulty block | Ms2<br>% of functions with higher risk than the faulty function | % of blocks with higher risk than the faulty block |
|---|---|---|---|---|
| F01 | 3.70 | 1.61 | 6.67 | 1.50 |
| F02 | 4.44 | 1.61 | 7.41 | 1.50 |
| F03 | 8.15 | 4.90 | 4.44 | 3.85 |
| F04 | 11.85 | 48.36 | 14.07 | 33.92 |
| F05 | 14.07 | 1.92 | 14.07 | 2.34 |

**Table 3:** Risk Assessment Based on the Dynamic Model

a) Risk assessment with respect to F02

| Expr No. | No. of successful test cases | MD1 % of functions with higher risk than the faulty function | MD1 % of blocks with higher risk than the faulty block | MD2 % of functions with higher risk than the faulty function | MD2 % of blocks with higher risk than the faulty block |
|---|---|---|---|---|---|
| 1 | 58 | 1.48 | 0.70 | 2.96 | 1.08 |
| 2 | 0 | 4.44 | 1.61 | 7.41 | 1.56 |
| 3 | 11 | 2.22 | 0.77 | 4.44 | 0.94 |
| 4 | 83 | 2.96 | 0.45 | 2.22 | 0.63 |
| 5 | 16 | 2.22 | 0.87 | 3.70 | 1.29 |

b) Risk assessment with respect to F03

| No. of successful test cases | MD1 % of functions with higher risk than the faulty function | MD1 % of blocks with higher risk than the faulty block | MD2 % of functions with higher risk than the faulty function | MD2 % of blocks with higher risk than the faulty block |
|---|---|---|---|---|
| 0 | 8.15 | 4.90 | 4.44 | 3.085 |
| 3 | 2.96 | 2.97 | 2.22 | 2.45 |
| 76 | 1.48 | 1.50 | 2.22 | 1.4 |
| 7 | 64.44 | 51.75 | 57.78 | 45.45 |

AVG 37.3 2.07.
0.77 3.48 1.0.

## 5. Conclusion

This paper gives the survey of different types of techniques which are explicitly used for the assessment of the software risk that are present in the software projects. The techniques which are mentioned in this project may help to eradicate and minimize the risk. Not only a correct result gives immense effect to project but also risk free projects do. Either following these techniques will help developers or based on these assessments techniques we can find ways and other refinements to get many more assessment techniques.

## Acknowledgement

This study is done with the help of the paper “Source Code-Based Software Risk Assessing” from ACM Symposium on Applied Computing by W. Eric Wong†, Yu Qi, and Kendra Cooper Department of Computer Science University of Texas at Dallas Richardson, TX 75083.

## References

[1] Kutay, C. and Babar, M. A. 2005. Teaching three quality assurance techniques in tandem-lessons learned. In Fifth International Conference on Quality Software. QSIC’05. IEEE, 307–312 https://doi.org/10.1109/QSIC.2005.62.

[2] B. W. Boehm, “Software Risk management: Principles and Practices,” IEEE Software, vol. 8, no. 1, pp. 32-41, Jan.1991. https://doi.org/10.1109/52.62930.

[3] B. W. Boehm, Software Risk Management, Tutorial, IEEE CS Press, 1989.

[4] H. Barki, S.Rivard, and J. Talbot, “Toward an Assessment of Software Development Risk,” J. Management Information Technology, vol. 22, no. 2, pp. 359-371, Dec. 1993. https://doi.org/10.1080/07421222.1993.11518006.

[5] M. Carr, S. Kondra, I. Monarch, F. Ulrich, and C. Walker, “Taxonomy-Based Risk Identification,” Technical Report SEI-93-TR-006, SEI, Pittsburgh, USA, 1993.

[6] S. A. Sherer, “The Three dimensions of Software Risk: Technical, Organizational, and Environmental,” Proc. 28th Hawaii International Conference on System Sciences, pp. 369-378, 1995. https://doi.org/10.1109/HICSS.1995.375618.

[7] C. G. Chittister and Y. Y. Haimes, “System Integration via Software Risk Management,” IEEE Trans Systems, Man, and Cybernetics, vol. 26, no. 5, pp. 521-532, Sep. 1996. https://doi.org/10.1109/3468.531900.

[8] J. Ropponen and K. Lyytinen, “Components of Software Development Risk: How to Address Them? A Project Manager Survey,” IEEE Trans. Software Engineering, vol. 26, no. 2, pp. 98-112, February 2000. https://doi.org/10.1109/32.841112.

[9] M. Keil, P.E. Cule, K. Lyytinen, and R.C. Schmidt, “A Framework for Identifying Software Project Risks,” Communications of the ACM, vol. 4, no. 11, pp. 76-83, Nov. 1998. https://doi.org/10.1145/287831.287843.

[10] L. Wallace and M. Keil, “Software Project Risks and Their Effect on Outcomes,” Communications of the ACM, vol. 47, no. 4, pp. 68-73, April 2004. https://doi.org/10.1145/975817.975819.

[11] B. W. Boehm, “A Spiral Model of Software Development and Enhancement,” IEEE Computer, vol. 21, no. 5, pp. 61-72, May 1988. https://doi.org/10.1109/2.59.

[12] A. Gemmer, “Risk Management: Moving Beyond Process,” IEEE Computer, vol. 30, no. 5, pp. 33-41, May, 1997. https://doi.org/10.1109/2.589908.

[13] H. Hecht, Systems Reliability and Failure Prevention. Artech House Publishers, 2003.

[14] Aihua Yan,”Risk Management in Software Development: A Continuous Process” IS 6840 Term Paper, fall 2008; Submitted to Dr. Vicki Sauter, November 21, 2008.

[15] Greenbelt, MD 20771 Greenbelt, MD 20771 Greenbelt, MD 20771301-286-0087 301-286-0101 301-286-8012.

[16] McMahon, Keith, "Risk Based Testing", ST Labs, WA, 1998.

[17] Pfleeger, S.L. and Palmer, J.D., “Software Estimation for Object Oriented Systems,”Int’l. Function Point Users Group Fall conference, San Antonio TX, 1990.

[18] Jingyue LI, Reidar CONRADI, Odd Petter N. SLYNGSTAD, Marco TORCHIANO, Maurizio MORISIO, Christian BUNSE.

[19] Fundamentals of Risk Management Understanding, evaluating and implementing effective risk management by Paul Hopkin.

[20] Barki, Henri, Suzanne Rivard, and Jean Talbot. "Toward an Assessment of Software Development Risk." Journal of Management Information Systems, 1993: 203-225. https://doi.org/10.1080/07421222.1993.11518006.

[21] Iversen, Jakob H., Lars Mathiassen, and Peter Axel Nielsen. "Managing Risk in Software Process Improvement: An Action Research Approach." MIS Quarterly, 2004: 395-433. https://doi.org/10.2307/25148645.