Software Risk Assessment and Estimation Model

Daya Gupta
Department of Computer Engineering
Delhi College of Engineering, Bawana Road, Delhi-110042
Faculty of Technology, University of Delhi-110007 (India)
E- Mail: Daya_gupta2005@yahoo.co.in

Mohd. Sadiq
Computer Engineering Section, University Polytechnic
Faculty of Engineering and Technology
Jamia Millia Islamia (A Central University), New Delhi-110025 (India)
[Research Scholar (Ph.D. in Computer Engineering), DCE, University of Delhi-110007]
E-Mail: msq_delhi@yahoo.co.in

Abstract

Just like any development projects, there is inherent risk in software development projects. Most of the software's fails due to over budget, delay in the delivery of the software's and so on. In this paper we have proposed a software risk assessment and estimation model (SRAEM). Using this model it is possible to predict the possible results of software projects with good accuracy. Proposed model not only assess the risk but it also estimate the risk. In this paper the risk is estimated using risk exposure and software metrics of risk management and this metric is based on Mission Critical Requirements Stability Risk Metrics (MCRSRM). This software metrics is used when there are changes in requirements such as addition, subtraction, or deletion. The proposed model gives the incremental risk for every phase and also the total cumulative risk as the software progress from phase to phase.

Keywords: Software risk, Risk exposure, Risk management.

1. Introduction: The management of risks is a central issue in the planning and management of any venture. In the field of software, Risk Management is a critical discipline. The process of risk management embodies the identification, analysis, planning, tracking, controlling, and communication of risk. It gives us a structured mechanism to provide visibility into threats to projects success. Risk management is a discipline for living with the possibility that future events may cause adverse effects. Risk management partly means reducing uncertainty **[1].** Reducing uncertainty has a cost associated with it. We need to balance such costs we could incur if the risk is not addressed. It may not be cost effective to reduce uncertainty too much. Risk management standard is the result of work by a team drawn from the major risk management organization. It is a central part of any organizations strategic management. It is the process whereby organizations methodically address the risk attaching to their activities with the goal of achieving sustained benefit within each activity and across the portfolio of all the activities [5].There are three dimension of software risk. (i) Technical risk (ii) Organizational risk (iii) Environmental risk. The technical dimension results from uncertainty in the task and procedure. The organizational dimension results from poor communication and organizational structure. The environmental dimension results from rapidly changing environment and problems with external relationship with software developers and/or users. As first technical, then organizational, and more recently, environmental risk, have hindered development, use, and maintenance of information systems, procedures have been developed to manage the various risk components. Not all organization uses these procedures and, even if they do, not all organizations use them successfully [10**].** The paper is organized as follows. In section-2 we present the background and the related work of risk management and risk assessment model. In section -3 we introduce the proposed software Risk assessment. Section -4 gives you the detailed description of the proposed model and its advantages with suitable examples. Finally we conclude the paper in section-5.

978-0-7695-3308-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ICCSIT.2008.184

2. Background and Related Work:

Software Risk Evaluation (SRE) is a process for identifying, analyzing, and developing mitigation strategies for risks in a software-intensive system while it is in development. The SRE process has been in evolutionary development at the SEI since 1992 and has been used on over 50 Department of Defense (DoD) and civil (Federal and state) contractors and program offices. There are certain questions that are related to Risk Management. The most important question is why we manage risk? The answer of this question is that all projects have some level of risk associated with them. Even if the product under development is simply another version of an existing system or product, risk may appear in areas such as: Change in the development personal (and resulting experience levels with the product) and changing market conditions and customer expectations. We would like to say only "the more you understand the risks, the better equipped you are to manage them". Risk management is a process that is systematically and continuous and it can be best described by the SEI risk management paradigm. There are six paradigm of risk management: (i) Identify (ii) Analyze (iii) Plan (iv) Track (v) Control and (vi) Communication. The detailed description of this paradigm is given in [7].There are a few published model that evaluate the risk of software projects. In [16], a method to evaluate risk using drivers is described. Risk drivers method is conceived following US Air force's guidelines for software risk identification and abatement. The US Air force defined the major risk components as performance risk, cost risk, support risk and schedule risk. This model does not have questions that bring out process related risks and is more suited for acquisition than development of software. Another model named Software Engineering Risk Model (SERIM) focuses on three risk elements: (i) technical risk, (ii) cost risk, and (iii) schedule risk. This model does not take into account of the software complexity issues, which plays an important role in determining the risk for the software projects. It also does not account for issues related to requirements. The proposed models addresses this problem and estimate the risk at each phase of software as it progress from phase to phase. In the series of risk assessment models there is another model called Software Risk Assessment Model (SRAM). This model considers the nine critical risk elements (i) complexity of the software; (ii) staff involved in the projects (iii) targeted reliability (iv) product requirement (v) method of estimation (vi) method of monitoring (vii) development process adopted (viii) Usability of software (ix) tools [16]. The above existing risk assessment models does not include the sources of estimate uncertainty, i.e. measurement error, model error and assumption errors. Existing methods have considered the prioritization as a single step of risk assessment but does not specify how prioritization would be done. In this paper we have considered two different approaches to estimate and rank the risk and that would be used for the prioritization. The first method is based on probability of risk, and in this case risk would be estimated with the help of risk exposure and the second method is based on Mission Critical Requirements Stability Risk Metrics.

Sources of Estimate Uncertainty: For many years, software estimation experts have pointed out that an estimation value is one of a range of values with a specific probability of being realized. For example, DeMarco proposed the following definition:

"An estimate is a prediction that is equally likely to be above or below the actual result"

Estimate uncertainty occurs because an estimate is a probabilistic assessment of future condition. Estimators produce an estimate using an estimation model. The model may be formulated as a standard productivity rate for a specified task, a set of ad-hoc rules, or a mathematical formula. But however it is derived; the model itself can be a source of estimate error [2]. We consider this source as a key step of our proposed model along with three sources of estimate uncertainty and risk: measurement error, model error, and assumption error.

3. Proposed Software Risk Assessment and Estimation Model:

Risk assessment provides a snapshot of the risk situation and is part of a viable risk management program. Risk assessment begins with team training. The team meets prior to the risk assessment for team building and training in the details of risk management paradigm and risk assessment process. This training includes instruction in the risk management mechanism to be applied during the assessment as well as practice exercises using the mechanism. Another important part of the training period is familiarizing the assessment team with the program to assess **[1, 12]**. There are three key factors of risk assessment and theses factors are risk identification, risk analysis, and risk prioritization. As figure 1 shows the practice of risk management involves two primary steps each with three subsidiary steps. The first primary step of risk management is risk assessment and it involves risk identification, risk analysis and risk prioritization. And the secondary step of risk

management is risk control and it involves risk management planning, risk resolution, and risk monitoring. Our main emphasis would be on risk assessment.

1. Risk identification produces lists of the projects-specific risk items likely to compromise a project success. A typical risk identification technique includes examination of decision drivers, assumption analysis, and checklist.
2. Risk analysis assesses the loss probability and loss magnitude for each identified risk item and it access compound risk in the risk item interactions. Typical technique include performance models, cost models, network analysis etc.
3. Risk prioritization produces a ranked ordering of the risk items identified and analyzed. Typical techniques includes risk exposure analysis, risk reduction leverage (particularly involve cost-benefit analysis) etc.

The second primary step is risk control and it involves risk management planning, risk resolution, and risk monitoring.

1. Risk management planning helps prepare you to address each risk item including the coordination of individual risk item plans with each other and with the overall project plan. Typical technique includes checklist for risk resolution technique, cost benefit analysis, and standard risk management plan outlines, form and elements.
2. Risk resolution produces a situation in which the risk items are eliminated or otherwise resolved (for example risk avoidance via relaxation of requirements). Typical technique includes prototypes simulation, benchmark, mission analysis, and design to cost approach.

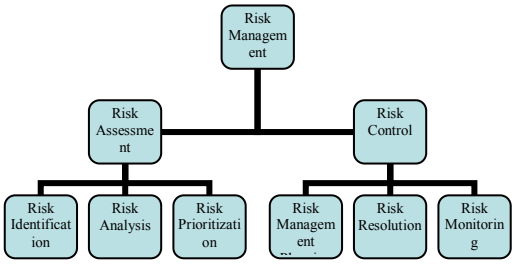

Figure-1 Steps involved in the Risk Management

3. Risk monitoring involves tracking the project's progress toward resolving its risk items and taking corrective action where appropriate. Typical technique includes milestone tracking and a top 10 risk item list that is highlighted at each weekly and monthly [2].

4 Explanation of the proposed SRAEM:

Initially the model estimates the sources of uncertainty using Measurement error, Model error and Assumption error. We have considered the concept of function point to explain the measurement error, Model error, and assumption error. Function point is an important software metrics which is used to calculate the approximate LOC, Cost and effort of software [8, 9, and 11]. The block diagram of the proposed model is given in figure-2

4.1 Measurement Error:

This error occurs if some of the input variables in a model have inherent accuracy limitations. As a result of Chris F. Kemerer [11], function points are assumed to be at least 12% inaccurate. Thus if we estimate a product size of 1300 function points, measurement error could mean that the real size is anywhere between 1144 and 1456. So applying a model of 0.5 person-days per function point means your estimate will have a range of uncertainty between 572 and 728 person days, with a most likely value of 650 person days.

4.2 Model Error:

This occurs because all empirical models are abstraction of reality. No estimation models can include all the factors that affect the effort required to produce a software product. Factors that affect error but are not included explicitly in the model contribute to the model error. For example such as 0.5 person-days per function point is usually obtained from results observed for recalled from previous projects. It is unlikely that any future projects will achieve the same ratio, but the model is expected to all right on average. If you base a model on past project data, you should calculate the associated inaccuracy by using the mean magnitude relative error. Thus if you have an estimation model with an inherent 20% inaccuracy and your product is 1300 function points in size, your estimate is likely to be between 130 and 390 person days.

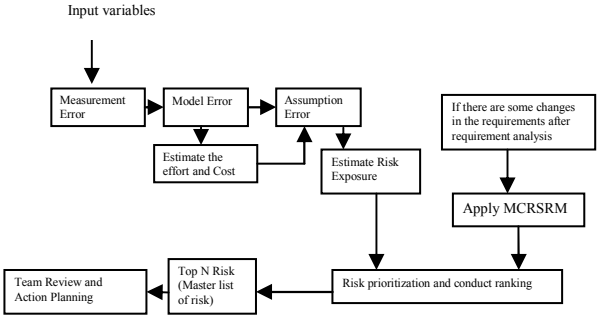

Figure-2
Proposed Software Risk Assessment and Estimation Model

4.3 Assumption Error:

This occurs when we make incorrect assumptions about a model's input parameters. For example, your assessment that a product size is 1300 function point rests on the assumption that you have correctly identified all the customer requirements. If you can identify your assumptions, you can investigate the effect of their being invalid by assessing both the probability that an assumption is incorrect and the resulting impact on the estimate. This is the form of risk analysis. For example you believe that there is a 0.3 probability that the requirement complexity has been underestimated and, if it has, you estimate another 100 function point. At this point the concept of risk exposure is used to calculate the effective current cost of a risk and can be used to prioritize risk that requires countermeasure. Mathematically it can be written as ***Probability of risk occurring X Total loss if risk occur***. In our example the Total loss= E2-E1. Where E1 is the effort if the original assumption is true and E2 is the effort if the alternative assumption is true. Suppose E1=540 person days and E2= (1300+100)*0.5=700 person days, then risk exposure= (700-540)*0.3= 48 person days. Assumption error basically helps you to estimate the risk exposure. Now the question is how we will prioritize the risk. We will explain the prioritization of risk with the help of the following example. Suppose in a software project we identified three different types of risk i.e. products recall situation, significant product rejection and competitive strike. The information about probability of risks occurring and the total loss if it occurs are given in the table-1.

Table-1

Risk	Probability of Occurring	Total loss if it occurs
Product recall situation	2%	80K
Significant product rejection	0.1%	1000K
Competitive Strike	10%	25K

The rank of risk is estimated using risk exposure and the value of the highest risk exposure indicate the most serious risk. Table-2 contains the calculated values of risk exposure and the ranking of risk.

As in the above table Competitive strike contains the highest value of risk exposure i.e. 2500, so it will get the first priority, then product recall situation has risk exposure equal to 1600, so it will get the second highest priority and similarly significant product rejection has third priority. So using this approach we can prioritize the given set of risk. Since risk exposure is not absolute but relative. We can compare different exposures to one another. One of the ways we can compare such exposures is to compare the exposure of a single event before and after managing the risk. We need a simple measure to assess risk reduction. So risk reduction leverage (RRL) is another quantitative means of assessing how risks are being managed. Mathematically we can write the RRL as (Risk exposure before - Risk exposure after)/ Cost of risk reduction.

In the proposed model there are two ways which are used to prioritize the risk. One method was based on risk exposure that we have recently explained and the other method is based on Mission Critical Requirements Stability Risk metric (MCRSRM). Now we will see how this software metric is used for estimation of risk. In this method the risk is computed when there are some changes in the requirements (addition, modification, or deletion). So total risk can be computed as $[b/a]_{i=1 \text{ to } n} + K\,[A\,[c/d]_i + B\,[d/b]_i + G\,[e/b]_i]$ .Where $[b/a]_i$= (Number of mission critical requirements)/ (Total number of requirements) at the input of phase number i. K_i is the penalty for adding, modifying or deleting of requirements during phase number i.

Table-2

Risk	Probability of Occurring	Total loss if it occurs	RE	RP
Product recall situation	2%	80K	1600	II
Significant product rejection	0.1%	1000K	1000	III
Competitive Strike	10%	25K	2500	I

In the above table RE is the Risk Exposure and RP indicates the risk priority. If the following information are giving at phase i=1.

a= total number of requirements =100, b=total number of Mission Critical Requirements (MCR) =40, c=number of MCR added during phase 1=5, d= number of MCR modified during phase 1=10 (3 which were downgraded from MCR to just requirements, and the remaining 7 are still MCR, and e= number of MCR deleted during phase 1=7 then total risk during phase 1 can be calculated as 30/100+ 1[3[5/40]+2[10/40]+1[7/30]] =40/100+ 42/40; where 40/100= the risk at the input of phase 1 and 42/40 is the added risk during phase 1, due to adding , modifying and deletion of MCR. Similarly

we can calculate the risk during the second phase. The detailed information about the metrics for software risk management is available in [4, 5, and 15]. It can be seen that this model gives the incremental risk for every phase and also the total cumulative risk as the project progresses from phase to phase.

After computation of risk and its prioritization the next step is team review and action planning. The team review opens with a review by each program manager of the current status of all risks and the action item from the previous review. Risk may be deleted from the risk item list but this is accomplished by agreement between the two program managers. Next, candidate new risks for inclusion on the list are reviewed. In the first occurrence of the team review, where no joint list of risks exists, each program manager selects approximately 5 to 10 risks from their respective master lists of risks to be included in the first version of the list. The next step is the ranking process, is to agree precisely on the comparison criteria. The risks are compared on the basis of which is more important to the program but importance can have many dimensions, such as cost, schedule, and fitness of the final product. Last step of this model is action planning. Action planning for risks is used for determination and implementation of actions necessary to manage a program risks.

5. Conclusions:

This paper is based on the risk assessment and estimation of software projects. Risk identification, risk analysis and risk prioritization are the main sub-parts of risk assessment. This paper gives you the detailed description of the risk estimation and risk prioritization. This model is different from the existing model because the existing model does not support the issues related to requirement and they do not also use the sources of estimate uncertainty. From the proposed model it is easy to calculate the risk at different phase as the software projects progresses from phase to phase.

References:

1. Roger L. Van Scoy, "Software Development Risk: Opportunity, not problem", Technical report, September 1992.
2. Barbara Kitchenham, Stephen Linkman, " Estimates, Uncertainty, and Risk", IEEE Software, pp.69-74, 1997
3. Barry W. Boehm, "Software Risk Management: Principles and Practices", IEEE Software, pp.32-41, 1991.
4. Donald Reifer, " Ten Deadly Risks in Internet and Intranet Software Development", IEEE Software, 2002.
5. A Risk Management Standard, ALARM, IRM:2002
6. Ronald P.Higuera, David P. Gluch, Audrey J. Dorofee, Richard L. Murphy, Julie A. walker, and Ray C. William, " An Introduction to Team Risk Management (Version 1.0), Software Engineering Institute, Technical Report, May 1994.
7. Ray C. Williams, George J. Pandelios, and Sandra G. Behrens, "Software Risk Evaluation (SRE) Method description (Version-2.0), Technical report December-1999.
8. Mohd. Sadiq, and Shabbir Ahmed, "Computation of Function Point of Software on the basis of Average Complexity", Proceedings of 2nd International Conference on Advanced Computing and Communication Technologies, **ICACCT2007**, Panipat, Haryana. Pp.591-594.
9. Mohd.Sadiq, Shabbir Ahmed,"Relationship between lines of code and Function Point and its application in the computation of Effort and Duration of a software using software equation", Proceedings of International Conference on Emerging Technologies and Applications in Engineering, Technology and Sciences , ICATETS2008, Rajkot, Gujrat.
10. Susan A Sherer, "The Three dimensions of Software Risk: technical, Organizational, , and Environmental,", IEEE-2005.
11. Chris F. Kermerer, "Reliability of Function Points Measurements, A Field Experiment, Communication of the ACM, pp. 85- 97, Vol.36, February 1993.
12. Tom Demarco, Tim Lister, "Risk Management during Requirements", IEEE Computer society, IEEE Software, pp.99-100, 2003.
13. K. Appukkutty, Hany H. Ammar, Katerina Goseva Popstajanova, "Software requirements Risk Assessment Using UML", IEEE, pp.1-4, 2005.
14. Poornima Ramachandra, Haeng-Kon Kim, Byeongdo Kang, Yan Ha, and Roger Lee, "Risk Management through Architecture Design", Proceedings of the Fourth International Conference on Software Engineering Research Management and Applications (SERA-06), IEEE-2006.
15. Joseph S. Sherif, "Metrics for Software Risk Management", ISMN#0-7803-3274-1, pp.507-513.
16. Say-Wei Foo , Armugam Muruganatham, " Software Risk assessment Model", ICMIT 2000, IEEE, pp-536-544.