A QUANTITATIVE METHODOLOGY FOR SOFTWARE RISK CONTROL

Chuk Yau
Software Quality Institute,
Griffith University,
Nathan, Australia

Abstract

Software risk control is becoming increasingly important in the software process. The huge losses reported by many software projects have become unacceptable to software development organisations and their clients. What is required is a method of determining, in the software life cycle, the risk that the project is facing. It is essential to identify the factors influencing the success of the project and estimate the possibility that these influences will cause the project to fail? This paper presents a methodology for software risk control. This methodology comprises several quantitative models for assessing and prioritising the software risks.

1. Software Project Risk Management

No project evaluation is completed without assessing the possible risk that an organisation will be exposed by developing a software project. Risk concerns future happenings [1,2]. Therefore, the assessment of risk depends upon some estimates which are given by a project manager at the planning stage. For example, the project management of a software project may conceive a risk that the project may not be completed by the target date. Each software risk constitutes two major risk components [3]. The first component is about the possibility (or likelihood) of the future event as the project overrun has not yet taken place. If the target date had come and passed, it would be possible to tell whether or not the project was late and would not be talking about the risk of this event. If there are sufficient statistical evidences, the possibility of an event can be represented by a probability. The second risk component is concerned with the consequence of such a risk. It is essential to determine what kind of loss will be resulted from the project overrun. If the project is not critical and the overrun will not cause great loss to the organisation, then the project does not carry a great risk. In general, the risk (R) of a software project is the loss (L) associated with missing any of the project objectives weighted by the possibility (P) of missing these objectives. Project risk R can be described as a function of P and L:

$$R=f(P,L)$$

Risk analysis is an activity which aims at evaluating the level of risk in order to reduce the potential loss. The risk analysis and the search for ways of coping with the risk should be regarded as ongoing efforts in any organisation that is striving to improve its software process. Figure 1 shows a risk management concept which addresses to the following three questions:

- How is risk measured?
- What is the measured risk meant?
- What can be done to reduce the loss?

Risk analysis can be a very difficult task to undertake. Without a well defined approach for modelling the risk situation of a software project, it is unlikely that the results of an informal risk analysis will reflect the true risks involved. A consistent approach of risk management will provide an organisation with a better foundation to conduct risk analysis continuously during the course of the project development. This paper introduces a risk management methodology which can provide answers to the above questions. The methodology comprises three major steps and will be described in detail.

Step 1: Establishing a Risk Analysis Model

(a) Determining the categories of the sources of risk and the categories of the consequences of risk for software projects. Each risk category should be divided into a number of risk factors that represent all related risk elements.

(b) For each risk category, establishing a risk implication table to describe the meanings of different levels of risk.

(c) Constructing a questionnaire for assessing the values of risk factors. The questionnaire must associate with the weight and the range of risk factor values of each risk factor, so that the project management can assign adequate values to different risk factors and calculate risk levels of different risk categories.

0-7803-2129-4/94 $3.00 **© 1994 IEEE**

Step 2: Calculating and Interpreting the risk levels

(a) Assign risk factor values based on the project systems requirements

(b) For each risk category, calculating the risk level which is a ratio of the total value of the assigned risk factor values to the maximum total value of the risk factor values of the category.

Step 3: Proposing possible risk control actions

(a) Establishing a risk matrix to prioritise different kinds of project risk.

(b) Identifying the possible problems by considering the relationships between project risks and software quality factors.

(c) Suggesting actions to reduce the possibilities of the critical problems.

2. Categorisation of Risk Factors

In order to establish a risk assessment model, it is useful to commence identify a set of risk categories which would apply to most projects of an organisation. There are two types of risk categories, namely source of risk and consequence of risk.

The categories of source of risk allow the management to assess the likelihood of failure. The risk levels of this kind of category are numbers between 0 and 1 representing the possibility of project failure. We use the term "possibility" rather than "probability" in this context due to the fact that a reliable statistical platform is hard to be established for generating accurate probability.

The categories of consequence of risk allow the project management to determine how serious a project failure will be. The risk levels of this kind of category should be normalised to the range between 0 and 1 representing the utility value of project failure.

Project managers are encouraged to determine their own risk categories for their organisations. Risk categories are devised for all projects of an organisation and is part of the standard project methodology. This section presents a set of generic risk categories which can be applied to most software projects. However, the six risk categories and their risk factors are not all inclusive. They can be supplemented with organisation-specific and project-specific risk categories.

(a) Categories of Source of Risk

Project Size and Complexity Category
- Number of Interface Systems;
- Number of Function Points;
- Number of user departments affected;
- System development cost;

Technical Complexity Category
- Data organisations and complexity;
- System processing mode;
- Programming language used;
- System Architecture.

System Management Category
- Data processing staff experience and knowledge;
- Security and access control utility;
- System development methodology requirement;
- System conversion requirement;
- Organisational resource commitment.

System Quality Assurance Category
- Organisational commitment on software quality;
- Experience in software quality assurance;
- Independency of quality team.

(b) Categories of Consequence of Risk

Business Management Category
- Will the strategic management decisions rely on the system?
- Will the tactical management decisions rely on the system?
- Will the system cause significant changes to the overall information flow of the organisation?
- Will the system cause significant changes to the operations practices of the organisation?

Financial Impact Category
- Will company's assets be controlled by the system?
- Will business turnover depend on the new system?
- Will the day-to-day business operation heavily rely on the system?

3. Risk Implication

In real life, some descriptive ratings, such as "extremely high" and "very low", are always used to represent the levels of risk. If the rating of a risk factor of a category of the source of risk is "extremely high", it implies that the possibility of project failure caused by the risk factor is extremely high. On the other hand, an "extremely high" rating for a risk factor of category of the consequence of risk implies that the possible loss caused by the project failure will be severe.

Figure 1 shows three different possibility distribution of

loss against the risk ratings of a project given by three different types of people, namely adventurism, normal and conservative. For different organisations, the descriptive ratings of risk levels may associate with different set of value ranging from 0 to 1.

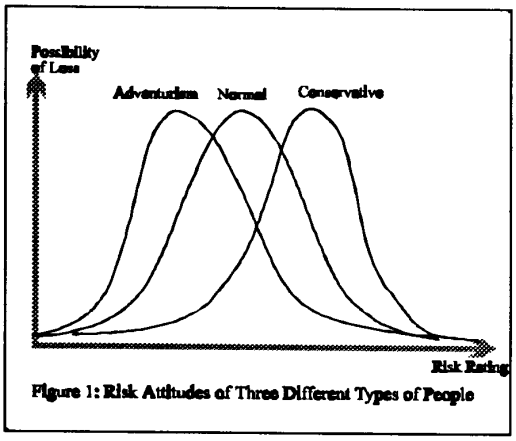

Figure 1: Risk Attitudes of Three Different Types of People

For different risk categories, the same risk level may represent different risk situations. The implication difference between different risk categories can be described by the following examples.

Situation 1:

Category: Project Size and Complexity
Risk level: 0.96 (Extremely high)
Implication: It is a very large software project which includes many subsystems of different software and hardware platform to be integrated altogether and requires extensive supports from different parties from inside and outside of the organisation. However, there is a straight due date but the organisation can not be able to commit sufficient resource to complete the project.

Situation 2:

Category: Systems Technical Aspect
Risk Level: 0.96 (Extremely high)
Implication: The software project has technical problems that beyond the state of the art. There is no advanced technology available for solving the problems. There are also time and resources constraints associated with the project.

4. Risk Assessment

For a software project, a risk questionnaire will be devised from the organisation's risk questionnaire by adding some project specific-risk factor. The risk questionnaire needs to be assigned risk values to help rate the project according to the potential risk. The project manager is often responsible for completing the questionnaire. The answers of the questionnaire should be based on the systems requirements. Some detail information may be obtained from other involved parties, such as users and accountants. In the course of completing the questionnaire, the project manager may want to change certain questions, weights, or risk value to improve the questionnaire for a particular project. Table 1 shows a completed questionnaire for the project size and complexity category.

Questions for Risk Factors	Risk Value	Weight	Response
(a) Number of systems interfaces?			
<= 1	0	5	
2 - 5	1	5	
5 - 10	2	5	✓
10 - 20	3	5	
21 - 30	4	5	
30 >	5	5	
(b) Number of function points			
< 100	0	4	
101 - 500	1	4	
501 - 2000	2	4	
2001 - 5000	3	4	✓
> 5000	4	4	
(c) Number of user departments involved?			
1	0	4	
2	1	4	✓
3	2	4	
4	3	4	
5	4	4	
> 5	5	4	
(d) Systems development cost?			
< $50K	0	3	
$50K - $100K	1	3	
$100K - $200K	2	3	
$200K - $500K	3	3	✓
$500K - $ 1M	4	3	
> $ 1 million	5	3	

TABLE 1: Questionnaire for the Project Size and Complexity Category.

Since different risk categories symbolise different characteristics of a software project, it is difficult to normalise the risk levels of different things. Any attempt to aggregate all risk levels of different category to a single risk level may lead to an incorrect outcome. For example, the systems technical risk may be "extremely high", say possibility of failure = 0.96, and the organisational risk may be "low", say possibility of failure = 0.1. The average possibility would be 0.53 which implies that the overall possibility of project failure is just "moderate". If the management considers the technical aspect is absolutely more important than any other aspects, the overall possibility might still be "extremely high" or at least "very high". Therefore, it is very dangerous to use the average possibility to determine the overall project risk. In order to achieve better interpretation of risks, the risk levels of different categories should be calculated and evaluated separately. The formula for calculating risk level of a category for the source of risk is:

$$P_k = \frac{\sum_{i=1}^{n} A_i \cdot W_i}{\sum_{i=1}^{n} Max(A_i) \cdot W_i}$$

where P_k is the possibility of failure due to the risk category k,
n is the number of risk factors of category k,
A_i is the assigned risk value of risk factor i,
$Max(A_i)$ is the maximum risk value of risk factor i,
W_i is the weight of the risk factor i.

For the project size and complexity category, if the descriptive rating of possibility of failure = 0.46 is moderate, then the risk implication is:

"It is a large software project which includes several subsystems and requires supports from the management of the user department. Project completion time and resources are negotiable. The possibility of failing some project objectives is moderate."

This risk implication is a general description for interpreting the possibility of failure. The project manager can revise the wording of the risk implication to reflect the specific project situation.

The utility values of failure (U) of the categories of consequence of risk can be calculated by the formula for calculating the possibility of failure.

$$U_k = \frac{\sum_{i=1}^{n} A_i \cdot W_i}{\sum_{i=1}^{n} Max(A_i) \cdot W_i}$$

5. Risk Interpretation

As mentioned earlier, project risks is the loss associated with missing any of the project objectives weighted by the possibility of missing these objectives. The last two sections have shown the ways of calculating the risk levels of categories of source of risk and consequence of risk. Since different risk categories represent different risk elements of software projects, it is recommended to consider the risk categories selectively. The various combinations between the categories of source of risk and the categories of consequence of risk represent different kinds of project risks. It is interesting to ask in what way that the sources of risk will result in consequences of risk. A very simple logic can be expressed as follow:

IF the project failure is due to a particular "source of risk"
THEN the organisation will suffer from the problem related to a particular "consequence of risk".

For example:

IF the project failure is due to the systems technical complexity
THEN the organisation will suffer from financial loss.

Although the above risk analysis logic describe the relationship between the sources and consequence of risk, it lack of indication about what may happen in between the source and the consequence. Project management may be interested in the following questions:

"What are the actually software engineering or quality problems caused by the source of risk which lead to the consequences ?"

To answer this question, a set of software engineering and quality factors should be considered for improving the logical relationship. The risk logic can be re-expressed as follow:

IF the project failure is due to a particular "source of risk"
THEN the project may have problems related to some software engineering and quality factors which lead to the loss of a particular "consequence of risk".

For example:

IF the project failure is due to the systems technical complexity
THEN the project may have reliability problems which may lead to financial loss to the organisation.

6. Risks Prioritisation

It is essential to prioritise the project risks, so that adequate risk control actions can be proposed to minimize the possible loss caused by the potential problems. The following risk priority index can be used to rank project risks.

$$RPI_{ij} = \frac{(P_i x U_j)}{\underset{\forall ij}{Max}(P_i x U_j)}$$

where RPI_{ij} is the risk priority index based on the category of source of risk i and the category of consequence of risk j;
P_i is the possibility of failure of the category of source of risk i;
U_j is the utility value of project failure of the category of consequence of risk j;

The RPI does not represent the risk levels but indicates the relative importance of a particular kind of project risk to the other kinds of project risk. Therefore, the risk implication tables should not be applied on RPI.

In the example matrix (Table 2), all P_i's and U_j's are artificially set for the demonstration purpose. These values should be obtained by following the steps of risk assessment. The risk matrix is an important result of risk analysis which provides project managers with clear guidance to identify and control different kinds of project risk.

As shown in Table 2, the source of risk, "Systems Technical Complexity" which is 0.62, and the consequence of risk, "Operational and Management" which is 0.67, lead to the RPI = 0.86. According to the meanings of "Systems Technical Complexity" and "Operational and Management", the project manager may conclude that it will be difficult to develop a system with a high degree of survivability. In order to satisfy this quality need, he has to specify quality requirements to ensure survivability. According to the Risk Priority Index (RPI = 1) of "Quality Assurance" and "Operational and Management", "Accuracy" and "Correctness" are the most important quality needs in this example.

The entries of quality attributes in the risk matrix, such as Flexibility, Integrity, Reliability, and Accuracy [4], are determined by the project manager based on his understanding of the sources and consequences of risk and experience in software development. The structure of the risk matrix can be regarded as a framework which allows project managers to determine their own interpretation of risk for formulating quality requirements.

7. Risk Control Actions

The ultimate objective of risk management is to reduce the possibility of project failure so as to minimise the potential loss. Among all identified risk areas, organisations must focus on the high priority ones. Risk control actions can be proposed based on two dimensions.

The first dimension is related to specific risk factors listed in the questionnaires. For example, the questionnaire of quality assurance category indicates that the number of quality staff is not enough and the quality assurance team lacks of a high degree of independence. These identified problems may be controlled by the following actions:

- Recruit more quality staff;
- Pre-scheduling key quality activities; and
- Re-define the responsibilities and authorities of the quality team.

The second dimension is concerned with the general software engineering and quality factors. For example, the risk matrix indicates that if the project have accuracy and correctness problems, the organisation's management activities will be severely affected. The following are possible actions which should be conducted to minimise the accuracy and correctness problems:

- Develop the system by evolutionary prototyping;
- Arrange regular independent verification and validation;
- Establish operational scenarios at the early stage of the development;

Table 3 shows the top ten software risk items and the possible control actions based on Barry Boehm's survey. They are good examples of how project risks can be handled. However, for different development environments, other possible actions may also be found effective in dealing with project risks. Project managers should propose control actions based on their risk analysis.

8. Concluding Remark

The risk control method described in this paper combines the questionnaire approach and the possibility theory. The most important feature of this method is its flexibility. It provides a set of questions that can be added to or removed, depending on the project characteristics and the risk assessor's preference and experience in his organisation. The quantitative models are very simple but practical.

There are many areas that the methodology can be improved to overcome the subjectivity of human judgement. The weighting system presented in this paper is not mathematically sound. It is possible to improve the weighting mechanism by adopting the Analytic Hierarchy Process [5,6].

At the moment, risk assessment is based on gut feeling and personal judgement rather than statistical evidence. Fuzzy concept [7] can be employed to determine the confidence level of the risk assessment.

The risk control part can be enhanced by using a knowledge base approach which can help project managers to construct their own risk matrixes with the consideration of the software quality needs of relevant parties of their organisations.

9. Reference

[1] Carette, R.N., (1989), "Software Engineering Risk Analysis and Management", New York, McGraw-Hill.

[2] Boehm,B.W., (1989), "Software Risk Management", IEEE Computer Society Press.

[3] Roetsheim,W.H., (1988), "Structured Project Management", Prentice-Hall.

[4] Deutch M.S. and Willis R.R., (1988), "Software Quality Engineering: A Total Technical and Management Approach", Prentice-Hall.

[5] Saaty T.L., (1980), "The Analytic Hierarchy Process", McGraw Hill.

[6] Zahedi F., "The Analytic Hierarchy Process - A Survey of the Method and its Applications, Interface, Vol.16, No.4. P.95-108.

[7] Kandel, A., (1986), "Fuzzy Mathematical Techniques with Application", Addison Wesley.

Consequences of Risk / Sources of Risk	Operational and Management Category U_1=0.67, High	Financial Impact Category U_2 = 0.27 (Low)
Project Size and Complexity Category P_1 = 0.46 (Moderate)	RPI = 0.64 Flexibility; Integrity	RPI = 0.26 Efficiency of development
Systems Technical Complexity Category P_2 = 0.62 (High)	RPI = 0.86 Survivability	RPI = 0.35 Reliability
Systems Management Category P_3 = 0.5 (Moderate)	RPI = 0.69 Manageability	RPI = 0.28 Efficiency of development
Quality Assurance Category P_4 = 0.72 (High)	RPI = 1.0 Accuracy; Correctness	RPI = 0.40 Maintainability

TABLE 2: Risk Matrix

Risk Item	Possible Risk Control Approaches
Personnel shortfalls	Staffing with top talent; Job matching; Morale building; Regular training
Unrealistic schedules and budgets	Multisource cost and schedule estimation; Design to cost; Incremental development; Software reuse; Requirement scrubbing
Developing the wrong software functions	Mission analysis; User survey; Operations concept formulation; Organisation analysis; prototyping; Early users' manual
Developing the wrong user interface	Prototyping; Operational scenarios; User characterisation
Gold plating	Requirements scrubbing; Prototyping; Cost/benefit analysis; design to cost
Continuing stream of requirements changes	High change threshold; Incremental development
Shortfalls in externally furnished components	Benchmarking; Compatibility analysis; Inspections; Reference checking;
Shortfalls in externally performed tasks	Reference checking; Pre-award audits; Award fee contracts; Team building; Competitive design or prototyping
Real-time performance	Simulation; Benchmarking; Modelling; Prototyping; Instrumentation; Tuning
Straining computer science capabilities	Technical analysis; Prototyping; Cost/benefit analysis; Reference checking

TABLE 3: Top Ten Risk Items and Control Approaches