J. Indian Inst. Sci., Nov.–Dec. 2006, **86**, 625–638
© Indian Institute of Science.

Enterprise risk evaluation and continuous mitigation using the fuzzy-multiattribute decision making–A conceptual approach

NIRAJ KUMAR* AND K. R. SRIVATHSAN
Indian Institute of Information Technology and Management-Kerala, Park Center, Techno Park Campus, Thiruvananthapuram 695 581, Kerala, India.
email: niraj-pg4@iiitmk.ac.in.

Received on December 8, 2005; Revised on September 26, 2006, and October 17, 2006.

Abstract

Software development processes generally follow easily identifiable stages and become increasingly more challenging. They differ from traditional manufacturing project stages in various ways, which make them very risky and uncertain. Traditional software risk management practices are more focused on qualitative judgment and experiences; however, with exponential growth in number and size, software companies require effective scientific methodology for risk management. The main objective of this study is to increase the effectiveness of risk detection and mitigation practices in the software development process. Another objective is to analyze these practices, identify the areas for improvement, and develop a mechanism for their quantification. We also aim to identify the processes, which cause problems and suggest strategy to eliminate or reduce the harmful effect of these processes in minimum possible cost and time. Analytical hierarchy process and fuzzy set theory are suggested as effective tools to achieve this objective.

Keywords: Risk management, AHP, fuzzy set theory, software engineering, CMM.

1. Introduction

Software development processes generally follow easily identifiable stages like project planning, requirement definition, design, development, testing, integration, installation, acceptance and support. However, they differ from traditional manufacturing project stages in various ways. Firstly, we need to give time and cost estimates to the customers in advance without actually knowing its exact nature, which make software projects very risky and uncertain. Secondly, to judge the requirements in diverse domain areas with almost the same set of manpower requires a lot of flexibility on the part of employees and management. Then fast rate of changes in technologies, customer requirements, possibility of unexpected number of employees leaving the organization make the task even more complicated. Integrating the complete solution, implementing it in successful ways, and making the end-user understand the software and changes in the traditional processes are all part of the software development life cycle. More and more global companies are outsourcing their software development projects to offshore software development destinations like India which are

*Author for correspondence. System Integration, CMC Limited, Fourth Floor, 28, Camac Street, Kolkata 700 016, India; Phone: 0-9332125744 (Mobile).

throwing new challenges in terms of requirements specifications, cultural differences, effective communication, security of valuable information, more rigorous legal, governance and compliance standards.

Because of challenges posed by software development risks many approaches have been proposed to improve the development process. These approaches include various computer-aided software engineering (CASE) tools, enterprise project management tools, conceptual tools like various quality, coding, process standards, models, and frameworks, and various software engineering models like traditional waterfall and spiral model.

In some cases, modeling and design tools are widely used and include Rational Rose [1, 2], versioning control tools like CVS, various testing and bug-tracking tools like Bugzilla, integrated development environment and automatic code generation tools like Visual Studio, Jcreator, Dreamweaver and others. Requirements development and management have always been critical in the implementation of software systems. Some automated tools are also available to support requirements management. The use of these tools not only provides support in the definition and tracing of requirements, but also opens the door to effective use of metrics in characterizing and assessing testing. Metrics are important because of the benefits associated with early detection and correction of problems with requirements [3].

Enterprise project management tools are found to provide a wide range of functions. Among these are scheduling, resource allocation, and cost estimating, budgeting, and collaborating. It is important to note that these tools emphasize project performance relative to resource consumption within a given set of time constraints (i.e. progress and dates). Some of the popular enterprise project management tools include Welcome product suite, Microsoft project 2002, Primavera P3e suite. While interest and investment in CASE and project management tools are rising steadily, actual experiences with tools have exhibited more ambiguity. There has been no systematic examination or formulation of the organizational changes surrounding CASE tools [4]. The major challenge is to develop quality software in a reliable and repeatable manner while improving productivity [5].

On the conceptual front all these challenges gradually led to development of capability maturity model (CMM) and concept of 'Balanced Scorecard'. The concept of 'Balanced Scorecard' [6] developed in the early 1990s by Kaplan and Norton represented an advance in the field of measuring enterprise performance, providing a framework for companies to evaluate both financial and 'non-financial', or 'extra-financial' measures such as quality, customer and employee satisfaction. This framework was widely used for enterprise-level planning, control and monitoring by software companies. However, it suffers from limitations of over reliance on expert judgments for decision making.

CMM [7] was developed in the early 1990s by Carnegie Mellon's Software Engineering Institute (SEI). Since then many versions of this model appeared and are widely accepted as most rigorous and best standard by software industry throughout the world. What led to instant success of this model is its ability to focus not only on external processes but also provide a framework for continuous improvements of internal processes in the company.

Table I
Key process areas of CMM levels [7]

CMM Level 2 KPAs	CMM Level 3 KPAs	CMM Level 4 KPAs	CMM Level 5 KPAs
Requirements management	Requirement development, Technical solution, product integration	Quantitative process management	Defect prevention
Software project planning	Verification, validation	Software quality management	Technology change management
Software project monitoring and control	Organization process definition and focus	X	Process change management
Supplier agreement management	Integrated software management	X	Continous improvement and optimization
Measurement and analysis	Organizational training, integrated project management	X	X
Software configuration management	Risk management, integrated teaming	X	X
Process and product quality assurance	Integrated supplier management, decision analysis and resolution, integrated supplier management, integrated teaming	X	X

The CMM establishes an yardstick against which it is possible to judge, in a repeatable way, the maturity of an organization's software process and compare it to the state of the practice of the industry. The CMM can also be used by an organization to plan improvements to its software development processes.

However, implementing the CMM framework is a very challenging problem for three fundamental reasons. First is the issue of recognizing all input/output parameters in real time, which influence various key process areas. Second, methodologies for quantification, optimization, and continuous improvement of these processes are vague. Third is the issue of synchronization of these processes with overall organizational objectives, profitability, efficiency, and growth.

Our primary focus in this study is to develop a model to analyze risk management (CMM level 3), quality management (CMM level 4), quantification of processes (CMM level 4), and defect prevention, continuous improvement and optimization (CMM level 5). Then based on analysis we also suggest improvement into various processes, which should be cost effective and suited for particular organization culture.

Software development process analysis is an important first step towards making the process more efficient and profitable. However, analysis is difficult not only due to its complex nature because of large number of subsystems involved and dynamic interaction and influence of one over another, but also because it is difficult to quantify their contribution and influence on the software development process as a whole. For example, technical expertise of manpower and quality of software developed are two important factors affecting any software company. However, technical expertise in itself can play an important role in the quality of software delivered, at the same time it may also lead to cost and time overrun, which may have negative consequences for the project. Similarly, the quality of software also depends upon the processes adopted for quality control and on the amount of time and funds available for the project. Also, the resources of an enterprise are limited so not all sources of risk can be immediately eliminated and priorities need to be established. Studying the various processes of a software development life cycle and measuring their impact in quantitative terms is one of the important objectives of this study. Again identification of key sources of risk and the ability to measure the level of its harmfulness on the system as a whole is another important objective of this study. In this paper, various risk sources for a software enterprise have been identified and a new approach based on analytical hierarchy process [8] and fuzzy set theory [9–11] has been suggested as effective tool for enterprise risk evaluation and continuous mitigation.

2. Literature review

Many researchers have tackled problems related to software development processes, practices and tools, multicriteria and multiattribute decision-making, risk management and fuzzy set theory.

Wiegers [12] points out unstated expectations as major source of software project failure which can lead to erroneous assumptions, unfulfilled dependencies, unexpected risks and disappointed customers. He emphasizes the importance of documenting the requirements thoroughly, precisely and without ambiguity. Mall [13] gives an overview of software engineering practices and covers almost every aspect of software engineering in brief.

Ming and Smidts [14] deal with ranking of software engineering measures based on expert opinion. Over reliance on expert opinion is a major limitation of this study. These theories, methodologies and tools need to be subjected to rigorous test of practices to live up to their perceived expectations and promises.

Many methodologies are available [9, 10] in the literature for multicriteria and multiattribute decision-making including data envelopment analysis [15], analytical hierarchy process [16], multiattribute utility theory [17], and Bayesian analysis and outranking methods [19]. The AHP, designed to solve complex problems of multiple criteria involving both qualitative and quantitative parameters, was proposed by Saaty in early 1970s. The application of AHP is based on four basic principles, namely, decomposition, prioritization, synthesis and consistency. Among others, Rong *et al.* [20], Hafeez *et al.* [21], and Kumar *et al.* [22] have demonstrated the effectiveness of AHP to solve real-life industrial problems. Kumar and colleagues [23–25] have tackled the problem of enterprise-level planning and control with multiple criteria which include parameters like capacity, productivity, profitability, environment and safety involving a number of decision-making units (DMUs).

The acceptance of term risk is universal and it penetrates every discipline of the society. Each discipline visualizes risk in its own understanding. The concept of risk management varies from macro to micro level. Several authors have highlighted the objectives of establishing risk assessment and mitigation process [26, 27]. However, due to the dynamic nature of interaction between various risk sources understanding of their relationship is imprecise which can be equated with fuzziness. Further, they require far more detailed description than is usually available for analysis. Application of the fuzzy set theory is an effective tool to handle these kinds of real-life situations.

The theory of fuzzy set proposed by Zadeh [11] provides a strict mathematical framework in which vague conceptual phenomena can be precisely and rigorously studied. It has led to a paradigm shift in the way problems are solved in various disciplines. It can also be considered as a modeling language well suited for situations in which fuzzy relations, criteria and phenomena exist.

3. Risk management—A must for every software enterprise

Most software development projects fail to deliver acceptable systems on time and within budget. Many studies were conducted to study software project failure rate at global level including the Conference Board Survey 2001. Their key findings suggest that 40% of software projects failed to achieve their business case within one year of going live and project costs were found to be on average 25% over budget from the original estimates. Traditionally, user requirements, inadequate user documentation, excessive schedule pressure, low quality, low user satisfaction, cost overruns were considered some of the major risk sources for software projects. However, increasing concern over security, legal problems, rising cost of software development, and HR-related problems necessitates that these factors should also be taken into account in risk management [4, 7, 18].

Much of the failure could be avoided by managers proactively planning for dealing with risk factors rather than waiting for problems to occur and then trying to react. Usually, this reaction is too little and too late, because by the time the problem is fully recognized, the schedule has already slipped, a considerable investment has been made, and the product quality has suffered due to introduction of errors or workaround. Risk management is an important tool to provide insight into potential problem areas and to identify, address, and eliminate them before they derail the project. Software risk management is important because it helps avoid disasters, rework, and overkill, but more importantly because it stimulates win–win situations [3]. A typical risk assessment and mitigation system may follow the following cycle (Fig. 1):

Typical internal and external factors causing risk in the software industry include the following:

(i) Improper planning: Uncertain requirement, unprecedented efforts—estimates unavailable, infeasible design, unavailable technology, unrealistic schedule estimates or allocation, lack of flexibility in planning process.

(ii) Inventory-related problems: Uncertain or inadequate subcontractor capability, uncertain or inadequate vendor capability, poor quality of software/hardware supplied, excessive inventory level.

(iii) HR-related problems: Inadequate staffing and skills, lack of match between employee skill sets and project requirements, improper and rigid hiring policy, inconsistency in employee perception level, improper method of employee performance evaluation, conflict between employee and top management, short-term goal and local optimization, preference to self interest over organization interest, improper training program.

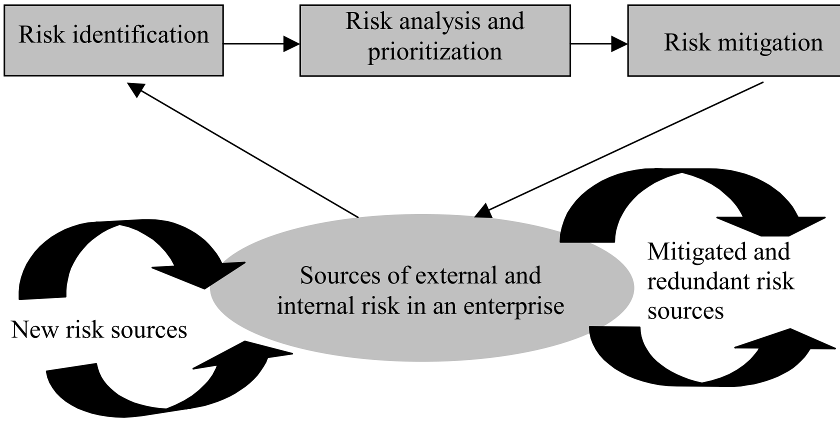

Risk identification
Risk analysis and prioritization
Risk mitigation
Sources of external and internal risk in an enterprise
New risk sources
Mitigated and redundant risk sources

FIG. 1. A typical risk assessment and mitigation system.

(iv) Customer-related problems: Risk from product rejection by the customer, risk from change in customer requirements, risk from project not delivered on time, risk from improper understanding of customer requirements, risk of losing other opportunities due to lack of customer satisfaction.

(v) Security-related problems: Risk from lack of technology for security handling, reactive approach of security in place of proactive approach, misapplication of scarce security resources, ineffective security goal setting, measurement, and achievement, misalignment between security goals and organizational drivers, risk due to theft or loss of valuable information due to security-related problems, risk from high cost of security management.

(vi) Quality-related problems: Poor product performance, lack of additional features, poor reliability of the product, nonconformance with specifications, lack of durability of product, lack of serviceability, lack of aesthetics, perceived quality of the product, lack of security features, lack of customer focus, faulty tool and techniques used for quality measurement, ease of training by the end-user, ability to integrate with legacy system of the customer, methodology of software engineering measures (bugs per line of code, code defect density, design defect density, failure rate, function point analysis, man hours per major defect detected, mean time to failure, requirement compliance, requirements specification change request, requirement traceability).

(vii) Communication-related problems: Risk due to delay in decision-making process, risk due to delay in information transfer, risk due to distortion of information, risk due to lack of proper communication between customer and team members.

(viii) Miscellaneous problems: Risk from fast technological change, risk from political uncertainty and government policy change, risk from lack of R&D effort and culture in the organization, ineffective utilization of available resources, financial risk, risk from recession from world economy, risk due to lack of overall system optimization, risk from foreign exchange fluctuations, risk from increasing software development cost, risk from litigation expense.

All the factors listed above are important sources of risk in the software industry. It is also important to understand that some of these sources are critical and their impact is profound across various processes of software development. Proper evaluation of all these factors and identification of key risk sources and subsequently their reduction or elimination with minimum cost and resources is necessary to make system more efficient and profitable and to gain competitive advantage in the market.

4. Proposed methodology

Methodology to be adopted for this study is four fold—first to identify the major risk factors, then able to quantify most of them, prioritize the factors based on their potential for causing risk to the organization and finally developing a general-purpose risk management software. These four stages can be summarized as

- **Identification** of the various sources of risk in the organization
- **Quantification** of these sources and estimation of their effect on the software development system as a whole.
- **Prioritization** of these sources according to their effect and importance in causing harm to the organization. Then, suggesting means to their elimination or reduction according to practical feasibility and requirement of the management.
- **Development of a general-purpose software**, which by giving suitable input, will be able to quantify and prioritise various risk sources. Also, it will be able to give estimates about how much elimination of any particular risk source expected to benefit the management and how much cost it is likely to incur.

The proposed methodology can be implemented as follows:

4.1. *Identification and quantification of key risk sources*

The constraints of time and cost make it impossible to eliminate all forms of risk in any enterprise, so an effective way to eliminate key sources of risk and continuous improvement of risk control process is required. Identifying and quantifying the key sources of risk that should be eliminated is an important first step towards achieving this. Figure 2 displays various potential risk sources identified and classified for a typical software enterprise. Then harmfulness of each type of risk on the system as a whole is proposed to be determined.

Some factors like risk from project rejection by customers, risk due to theft or loss of valuable information, risk from high cost of security management, risk from excessive inventory level, financial risk, risk from currency fluctuations, risk from increasing software development cost are relatively easy to quantify and estimate in monetary terms and their harmfulness on the system as a whole can be determined. For example, consider risk due to security-related problems. The cost of security-related problems on a software enterprise are a function of fixed and variable costs. Fixed costs include those costs that are independ-

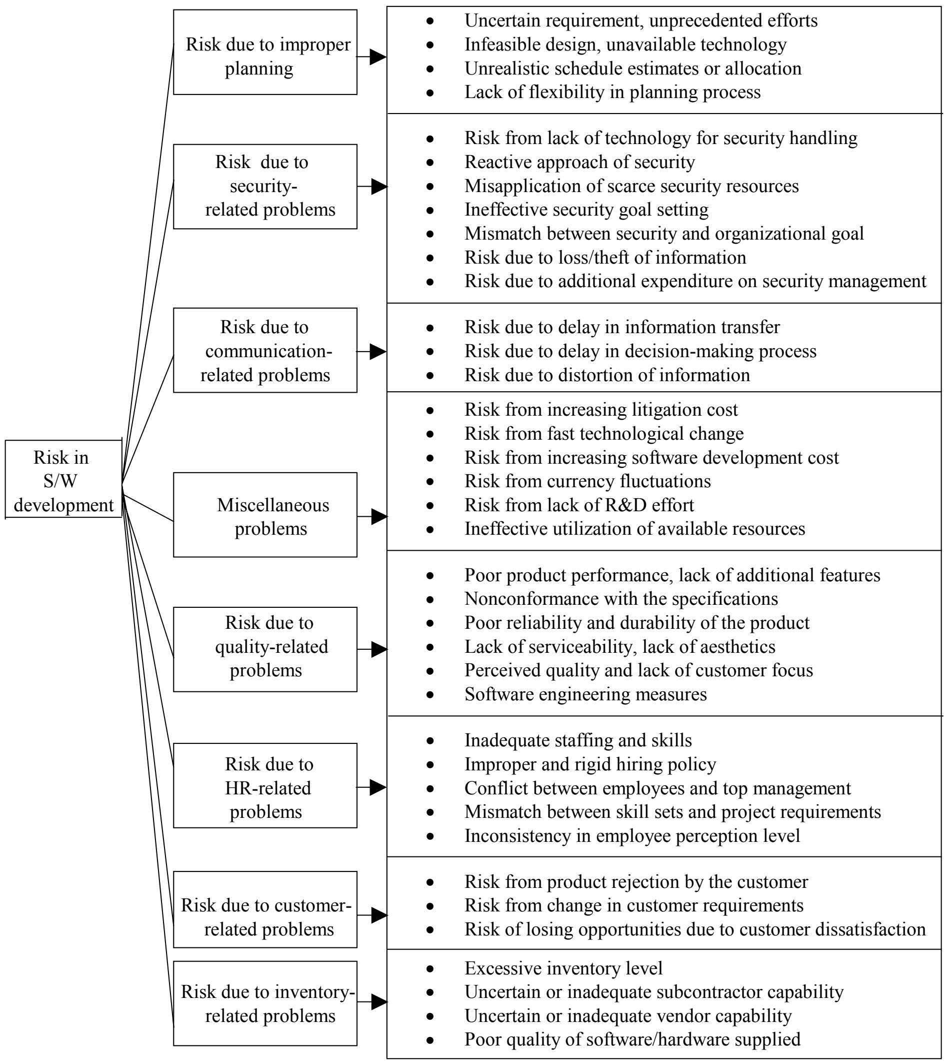

Risk in S/W development
Risk due to improper planning
Uncertain requirement, unprecedented efforts
Infeasible design, unavailable technology
Unrealistic schedule estimates or allocation
Lack of flexibility in planning process
Risk due to security-related problems
Risk from lack of technology for security handling
Reactive approach of security
Misapplication of scarce security resources
Ineffective security goal setting
Mismatch between security and organizational goal
Risk due to loss/theft of information
Risk due to additional expenditure on security management
Risk due to communication-related problems
Risk due to delay in information transfer
Risk due to delay in decision-making process
Risk due to distortion of information
Miscellaneous problems
Risk from increasing litigation cost
Risk from fast technological change
Risk from increasing software development cost
Risk from currency fluctuations
Risk from lack of R&D effort
Ineffective utilization of available resources
Risk due to quality-related problems
Poor product performance, lack of additional features
Nonconformance with the specifications
Poor reliability and durability of the product
Lack of serviceability, lack of aesthetics
Perceived quality and lack of customer focus
Software engineering measures
Risk due to HR-related problems
Inadequate staffing and skills
Improper and rigid hiring policy
Conflict between employees and top management
Mismatch between skill sets and project requirements
Inconsistency in employee perception level
Risk due to customer-related problems
Risk from product rejection by the customer
Risk from change in customer requirements
Risk of losing opportunities due to customer dissatisfaction
Risk due to inventory-related problems
Excessive inventory level
Uncertain or inadequate subcontractor capability
Uncertain or inadequate vendor capability
Poor quality of software/hardware supplied

FIG. 2. Risk identification and breakdown structure for software development process.

ent of the software developed like cost of maintaining firewalls, gateways, etc. These costs can result in increased capital and operating costs for software enterprise. Variable costs are those that vary directly with the software developed. Methodology that examines the cost impact of security-related problems can be based on crude estimates of the order of magnitude of fixed and variable costs. For this, reliability analysis of failure due to security-related problems is done and hazard function is determined.

Then cost model of security-related problems on software system can be given as:

$$GRPC = \frac{T_d}{MTBF} \times [FC + (VC \times MTTR)]$$

where,

GPRC: Security-related problem costs;
T_d: Designed scheduled operating life;
MTBF: Mean time between failures;
FC: Fixed cost for a single security-related failure;
VC: Variable costs for a single security-related failure per man days of down time;
MTTR: Mean time to repair the subsystem and bring it to its designed capacity.

Similar reliability models for other kinds of failure can be developed and their impact in the monetary term can be calculated.

Some other factors like lack of flexibility in planning process, risk from project not delivered on time, traditional software engineering measures, efforts estimate, various quality and reliability metrics, project schedule estimates have a number of empirical and analytical models which can be correlated with monetary gain or loss. Other factors are harder to quantify and innovative and focused research is required to quantify them in monetary term or judge their impact on the system as a whole. Expert judgment can be used as alternative option if no other suitable quantification model is feasible. Our literature review revealed that any of the present-day software engineering or quantification model is far below the level of challenges posed by the requirement of such a system.

Not all sources of risk are key sources. By key sources is meant the sources of most harmful types of risk that arise within software development process and those that are likely to cause other forms of risk and waste, that is those that are strongly correlated with the generation of other forms of risk and waste and whose elimination will suppress the generation of the strongly correlated forms of risk and waste. The risk due to different sources dynamically interacts with each other, but poor quality of information and vague benchmarks make it difficult to determine the degree of correlation. Hence, the various sources of risk are not easily evaluated and summarized into key risk sources. Understanding of the relations between the forms of risk is thus imprecise, which can be equated with fuzziness. To a great extent, software development process evaluation problem is a fuzzy unstructured decision problem.

To analyze the fuzzy relations among various risk types and identify the key sources of risk, the AHP and fuzzy set theory are proposed as effective tools. The proposed method consists of three steps: evaluating, clustering and ranking. First, a risk evaluation index system is established through the AHP to systematically measure the harmfulness of each risk source to determine which are most harmful. Here, attempts are being made to quantify the important source of risk and where quantification is not possible based on expert judgement decision is taken.

4.2. *Clustering*

After evaluation of software development process and identification of the various forms of risk and its degree of harmfulness, fuzzy clustering is used to cluster more harmful forms of risk on the basis of their fuzzy correlation and categorize the various risk sources into key risk sources (Fig. 3). Expert grading method is used to estimate the correlation between objects. The correlation between two risk sources is (0, 1). The bigger the number, the stronger the correlation is. After generating tolerance matrix and transforming it into fuzzy equivalence matrix, more harmful sources of risk that are strongly correlated can be clustered into a single risk source. This stage is likely to bring down the number of unmanageable risk sources into manageable few.

4.3. *Ranking*

The priorities by which key forms of risk should be eliminated are ranked based on the following important principles:

- Overall optimization of the software development process
- Consistency with the software enterprise development strategy
- Consistency with corporate technology level
- Consistency with skill set of the employees and management
- Its cost on the software company and time period required for improvement (shorter the better)
- Best quality product development and customer satisfaction
- Efficient use of various inputs going into software development process.

In the ranking process, first the primary elimination measure for each key risk source is determined. By evaluating these measures with regard to enterprise conditions, the key risk source to be eliminated can be decided and this will correspond to the most appropriate measures for the enterprise optimization process. Fuzzy comprehensive evaluation can be used for this purpose.

Based on the analysis results important risk sources are analyzed closely and improvement measure is proposed to be selected after which the benefits in terms of efficiency and profitability of software development process can be determined.

As seen from the flow diagram of Fig. 3, if the previous three stages are followed and appropriate elimination plan is implemented in focused manner, harmfulness from this particular key risk source is likely to be eliminated or considerably reduced. It should be monitored on a continous basis and all steps are repeated to find the next potential key risk sources. All these steps are repeated till risk factors from all key sources are eliminated or considerably reduced. In due course, many of the above-mentioned risk sources are likely to be nonexistent or irrelevent, while many new ones can arise and should be included in the model, which may be different for a particualar enterprise.

4.4. *Development of computer software for risk evaluation and quantification*

User-friendly computer software can be developed, which can quantify, cluster and rank the various sources of risk by giving suitable input. Accordingly, management will be able to generate useful information for elimination of risk sources and their effect on the software development process.

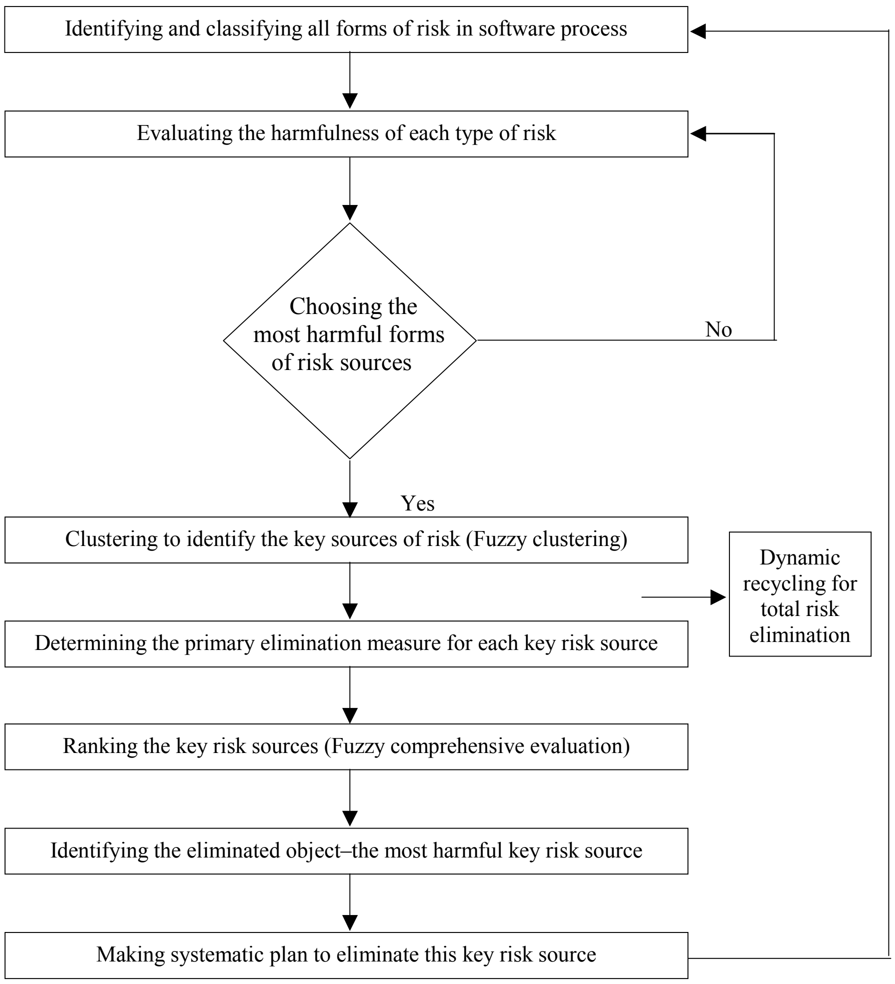

Identifying and classifying all forms of risk in software process
Evaluating the harmfulness of each type of risk
Choosing the most harmful forms of risk sources
No
Yes
Clustering to identify the key sources of risk (Fuzzy clustering)
Dynamic recycling for total risk elimination
Determining the primary elimination measure for each key risk source
Ranking the key risk sources (Fuzzy comprehensive evaluation)
Identifying the eliminated object–the most harmful key risk source
Making systematic plan to eliminate this key risk source

FIG. 3. Flow diagram of software risk-evaluation system [modified from [8]].

5. Model validation and tool development

Validation and implementation of the model to the scale, which is required for this kind of research, is yet to be fully realized. However, preliminary attempt of its validation based on information collected from a middle-level consultant of a fast-growing CMM-level 5 company has been done. This company is situated at Technopark, Thiruvananthapuram, and it relies on audits, reviews and testing for risk and quality management. However, no specific information regarding quantification of quality and customer satisfaction level has been provided.

Based on information, which is related with breakdown structure for software enterprise in Fig. 2, data was collected in April 2005, which identifies two most important risk sources

Table II
Two most important risk sources as identified by model validation

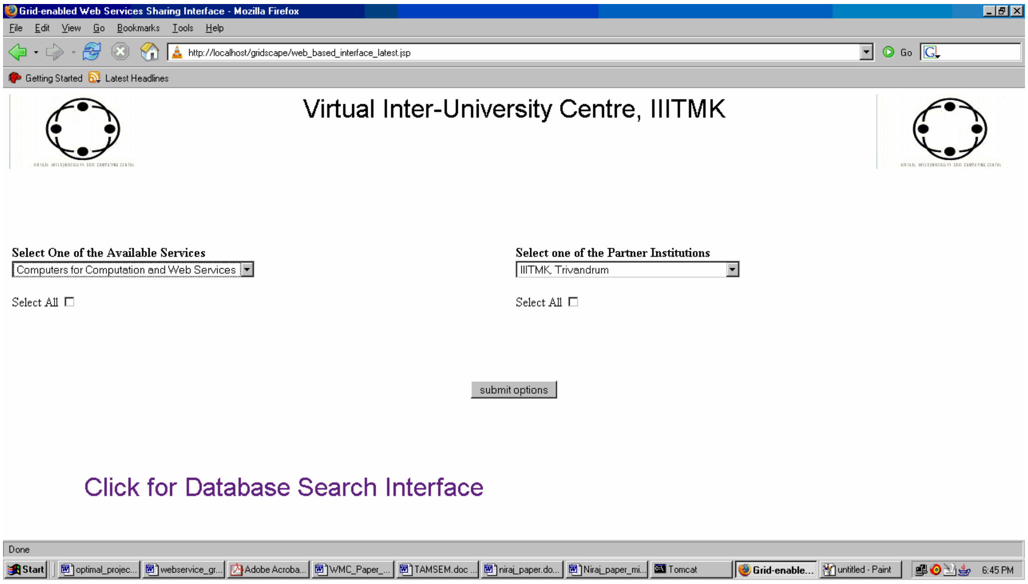
Category	Two most important risk sources
Improper planning	➢ Uncertain requirements ➢ Estimates unavailable
Customer-related problems	➢ Risk from change in customer requirements ➢ Product rejection by the customer
Security-related problems	➢ Reactive approach of security ➢ Risk due to loss of valuable information
Quality-related problems	➢ Nonconformance with specifications ➢ Poor product performance
Miscellaneous problems	➢ Ineffective utilization of available resources ➢ Risk from lack of R&D culture
Overall	➢ Customer-related problems ➢ Miscellaneous problems

in each category (Table II). However, these results are not conclusive, as the authors did not verify practices and processes of the company.

A web-based system for risk evaluation, quantification and multicriteria decision-making using Java-based technologies (JSP/Servlets), Mysql database and Apache tomcat application server were already started (see Fig. 4 for screenshot of one such interface) and plan to make it available in public domain after its completion.

6. Conclusion

Enterprise risk management is one of the most important strategic business tools to manage effectively a variety of risks to gain competitive advantage and add value to the firm. This


Grid-enabled Web Services Sharing Interface - Mozilla Firefox

http://localhost/gridscape/web_based_interface_latest.jsp

Virtual Inter-University Centre, IIITMK

Select One of the Available Services: Computers for Computation and Web Services

Select All ☐

Select one of the Partner Institutions: IIITMK, Trivandrum

Select All ☐

Click for Database Search Interface

FIG. 4. Web-based interface for enterprise risk evaluation.

study has identified the various external and internal risk sources in a software enterprise. Authors have proposed a scientific model based on AHP and fuzzy set theory to prioritize various risk sources and developed a framework for their continuous elimination by adopting enterprise-specific strategy. The paper also emphasizes the dynamic interactions between these factors and their suggested importance, quantification to judge the impact on software enterprise as a whole. This model is highly flexible and customizable according to specific enterprise need. Fuzzy clustering is proposed to cluster risk factors, which are closely correlated to each other and are likely to have common source of problems to enable their effective mitigation. However, quantifying (preferably in monetary terms) various risk sources and improvements required for their effective mitigation can be some of the potential directions in which this work can be extended.

Based on preliminary attempt of model validation with a CMM level-5 company it can be said that uncertain requirements, change in requirements, reactive approach of security, nonconformance with specifications and ineffective utilization of available resources are some of the major risk sources for this enterprise. Software enterprises can apply this new approach in their project and enterprise risk management to improve efficiency, performance, profitability and to meet rising enterprise management challenges.

Acknowledgements

The authors would like to acknowledge Prof. Ashis Bhattacherjee, Department of Mining Engineering, Indian Institute of Technology, Kharagpur, and anonymous reviewers for their invaluable suggestions and encouragement. Thanks are also due to Mr K. Sreenivasa Rao, Assistant Editor, *Journal of the Indian Institute of Science*, Bangalore for his efforts and suggestions for improvements in the manuscript and in the presentation of the paper. They also would like to thank Mr Amrendra Kumar, Department of Mining Engineering, Indian Institute of Technology, Kharagpur, and Dr Abhiram Verma, Department of Mining Engineering, B. E. College, Sibpur, West Bengal, for their help in the presentation of this paper.

References

1. Object Management Group, http://www.omg.org/gettingstarted/what is uml.htm
2. Rational Software Corporation, http://www.ibm.com/developerworks/rational/library/253.html
3. Rosenberg Linda, Hammer Theodore, and Gallo Albert, Continuous risk management at NASA, *Applied Software Measurement/Software Management Conf.*, San Jose, California (1999).
4. Orlikowaski Wanda, CASE tools as organizational change: Investigating incremental and radical changes in systems development, *MIS Q.*, **17**, 309–340 (1993).
5. Menendez Jose, *The software factory: Integrating case technologies to improve productivity*, Report LEAN, Center for Technology, Policy, and Industrial Development, Massachusetts Institute of Technology, p. 84 (1996).
6. The Balance score card, http://www.balancedscorecard.org/
7. Software Engineering Institute, www.sei.cmu.edu/sei-home.html.
8. T. L. Saaty, *The analytical hierarchy process*, McGraw-Hill (1980).
9. L. A. Zadeh, Fuzzy sets, *Inf. Control*, **8**, 338–353 (1965).
10. H. J. Zimmermann, *Fuzzy set theory and its applications*, Kluwer, pp. 241–260 (1991).

11. S. Osman Mohammed, Omar M. Saad and Ali A. Yahia, A fuzzy interactive approach to generate a finite set of efficient solutions for multi-objective programming problems with their degree of efficiency, *J. Fuzzy Math.*, **8**, 745–752 (2000).
12. Karl Weigers, See you in court, *Process Impact*, p. 6 (2003).
13. Rajib Mall, *Fundamentals of software engineering*, Prentice-Hall of India (1999).
14. Li Ming, and Carol S. Smidts, A ranking of software engineering measures based on expert opinion, *IEEE Trans. Software Engng*, **29**, 811–824 (2003).
15. A. Charnes, W. W. Cooper, and E. Rhodes, Measuring efficiency of decision making units, *Eur. J. Op. Res.*, **2**, 429–444 (1978).
16. Z. Sinuany-Stern, A. Mchrez, and Y. Hadad, An AHP/DEA methodology for ranking decision making units, *Int. Trans. Op. Res.*, **7**, 109–124 (2000).
17. R. L. Keeney, and H. Raffia, *Decisions with multiple objectives: Preferences and value tradeoffs*, Wiley (1976).
18. Thomas Bayes, An essay towards solving a problem in the doctrine of chances, *Phil. Trans. R. Soc.*, **53**, 370–418 (1763).
19. B. Roy, How outranking relation helps multiple criteria decision making, in *Multiple criteria decision making* (J. L. Cochrane and M. Eeny, eds), University of South Carolina Press, Columbia, SC, USA, pp. 179–201 (1973).
20. C. Rong, K. Takahashi, and J. Wang, Enterprise waste evaluation using the analytic hierarchy process and fuzzy set theory, *Int. J. Prod. Plann. Control*, **14**, 90–103 (2003).
21. K. Hafeez, Y. Zhang, and Malak Naila, Determining key capabilities of a firm using analytical hierarchy process, *Int. J. Prod. Econ.*, **76**, 39–51 (2002).
22. N. Kumar, A. Bhattacherjee, D. Chakravarty, and D. Sarkar, Efficiency measurement of mines using DEA and AHP, TAMSEM, *Int. Conf. on Technology Management for Sustainable Exploitation of Minerals and Natural Resources*, Indian Institute of Technology, Kharagpur, India, February 5–7, 2004 (2004).
23. Niraj Kumar, *Assessment of performance appraisal for mines using data envelopment analysis and fuzzy set theory–A case study from coal mining*, M.Tech. Thesis, Department of Mining Engineering, Indian Institute of Technology, Kharagpur, India (2002).
24. N. Kumar, A. Bhattacherjee, and D. Sarkar, Performance appraisal of coal mines using data envelopment analysis and fuzzy set theory, *Mintech*, **23**, 18–25 (2002).
25. Debasish Sarkar, Ashis Bhattacherjee, and Niraj Kumar, Performance evaluation of underground coal mines: A case study, *19th World Mining Congress*, New Delhi, pp. 917–928 (2003).
26. Jootar Jay, A risk dynamic model of complex system development, Ph.D. Thesis, Massachusetts Institute of Technology, p. 204 (2002).
27. V. Carr, and J. H. M. Tar, A fuzzy approach to construction project risk assessment and analysis: construction project risk management system, *Adv. Software Engng*, **32**, 847–857 (2001).