The risk analysis of software projects based on Bayesian Network

1,2Feng XU, 3,4Guijie QI, Yanan Sun
1,2First AuthorSchool of Management, Shandong University, Jinan P.R.China
2School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan P.R.China
3,4Corresponding authorSchool of Management Shandong University, qiguijie@sdu.edu.cn
4School of Business, Shandong University of Finance and Economics, Jinan P.R.China

Abstract

With the improving of software technology and the level of project management, the success of software project is not a special case. But this does not mean that software project can be finished easily. In fact, the success ratio of software project is still low. The existing of Software system aim to reduce the uncertainty of practical system, but the process of itself is full of uncertainty. So it is essential to identify and analyze the software risk. Now lots of researches focus on the ERP project risk analysis, which apply a number of qualitative methods and quantitative methods. But most of methods ignore the dynamic nature of ERP project, only paying attention to predict in advance or analyze afterwards, which cannot meet the challenge of ERP project management. So Bayesian network which can provide real-time analysis is selected to analyze the ERP project risks. Through literature review and investigation of Chinese software professionals, the paper finds out 12 risk factors. In terms of an ERP project of automobile mould factory in china, revising above-mentioned risk factors, according to easy expansibility of topological structure and strong self-learning ability of Bayesian network, the paper uses prior knowledge and posterior knowledge to analyze the risk of the ERP project in requirements period. Actual application shows that Bayesian network provide an effective system method for software project risk analysis.

Keywords: Bayesian network; risk analysis; IT project; software engineering.

1. Introduction

With the demand for software systems growing, there are more and more problems in the software industry, such as project delay, budget overruns or poor quality. Since 1994, with researching on IT projects, the Standish Group has promulgated <The Standish Group CHAOS Report> (http://www.standishgroup.com), which takes the longest time and largest energy to investigate the history data. From 1994 to 2004, the result shows that the success rate is very low as shown in figure 1. In 2005, the result for the third quarter of 2004 promulgated by the Standish Group was not so exciting either. The result showed that the completely failure rate was 18% in all project, and 53% of them were done with unsatisfactory time, costs or effect, only 29% of them had successfully achieved the project target.

![Figure 1. The survey result of IT projects from 1994 to 2000](image)

The core of ERP project is software development which is different from other products. The process of software development is just the process of engineering (no manufacturing). In addition, the
main needs in software development is not only the material resources, but also the human resources. And the software product has no materialization, but just codes and technical files. Based on the above-mentioned characteristics, the software project management is so unique compared with others. The software development needs many new techniques and old ones that have been validated, and the final product quantity is always low. It’s hard to form a standard technical process or mature process. Especially, ERP projects need long time, large-scale organization and coordination, as well as long development cycle, so there are many unforeseeable uncertainties which will not make the plan, costs, time and quality of software project to be predicted. The existing of Software system aim to reduce the uncertainty of practical system, but the process of itself is full of uncertainty. Therefore the problem of how to define, evaluate and measure ERP project risk and how to make solutions must be solved so as to minimize the impact of risk or reduce it to an acceptable level.

2. Recognition of the risk factor of ERP Project management

Boehm (1991) proposed most deadly top ten risk factors in software project management which are the staff’s shortcomings (talents, skills, responsibility, etc.), unrealistic project plan and budget, the wrong software functions and features, misunderstanding of the goal, user interface mistakes, frequent changes of the demand, the defects of completed external components, shortcomings in project management, the defect of real-time performance, the breach of computer science.

Marvin J Carr (1993) use classification to identify risk factors. He classified the software risk into three parts: product engineering, development environment and program restrictions. In detail, product engineering includes requirements, design, coding and testing, integration testing and engineering properties. Development environment includes the development process, development systems, management processes, management practices, working conditions. program restrictions include resources, contracts and procedures interfaces. Each of the sub-categories contains a number of specific risk factors. 54 specific risk factors were proposed totally.

Jones (1994) proposed 60 risk factors on enterprise, employees, customers, software technology and other aspects in software development environment. Also he specifically gave the frequency and the loss of events relative to each risk factor.

Anthony Kwok Tai Hui (2004) proposed 24 direct risk factors and other risk events including staff’s experience, morale, technology and environment needed for software project, as well as software interfaces and organizational structure.

Linda Wallace and Mark Keil (2004) said that the software risk exists in six dimensions: the team, organization, environment, demand, planning and controlling, the user, the project’s complexity. Each dimension included a number of risk factors.

With reference to the literature summary mentioned above, we made a survey for kinds of roles such as project managers, QA, testers, consultant, programmer, systems analyst, etc., having been working on software for more than 4 years from 6 different ERP projects. On this basis, this paper proposed the following risk factors:

- Lack the key technology. When the software project need some specific technology, such as demand analysis, algorithm designing, there is no one can do it.
- Rely on the key person. Without effective knowledge management, all the members can not reach the level that the project requires in a short time, so there always is over-reliance on a small number of core members.
- The lack of responsibility and low morale. This refers to lacking of discipline, staff instability, and low employee satisfaction.
- Lower productivity. This refers to means the serious overruns on project progress and cost.
- The lack of customer support. This means the lack of clear customer demands and low satisfaction for software products.
- The wrong criterion, refer to the wrong or hazy definition of success or fail project.
- The lack of necessary communication means that there is no effective communication mechanism between the project stakeholders.
- Lower CMMI Level means poor capacity about software project management and lacking software project management system.
Poor change control ability. This means when changes happen during the project implementation period, there is no systematic control for it.

High project complexity. This means the relationship between many activities are too complex to predict or control.

The pressure on the project time limit. This refers to the unreasonable compression on project time resulting in that the project cannot be finished.

Immature development technology means that members are not familiar with the development platform, products or development tools.

The above-mentioned risk factors are shown in table 1. According to the related events we can determine the level of risk factors, while the occurrence of related events will give rise to the associated risk factor. This relationship will form a Bayesian network structure.

Table 1. The list of risk factor

<table>
<thead>
<tr>
<th>Risk factors</th>
<th>Related events</th>
<th>The associated risk factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack the key technology</td>
<td>progress delay, Wrong designing, High error rate, Wrong product</td>
<td>Lower productivity</td>
</tr>
<tr>
<td>Rely on the key person</td>
<td>progress delay</td>
<td>The lack of responsibility and low morale</td>
</tr>
<tr>
<td>The lack of responsibility and low morale</td>
<td>High error rate, Staff mobility, Heavy workload</td>
<td>Lower productivity</td>
</tr>
<tr>
<td>Lower productivity</td>
<td>Long period, Increased workload, Increased project cost</td>
<td>The pressure on the project time limit</td>
</tr>
<tr>
<td>Lack of customer support</td>
<td>Lose customer demand, High error rate, Long period</td>
<td>The lack of responsibility and low morale, Poor change control ability</td>
</tr>
<tr>
<td>The wrong criterion</td>
<td>The wrong workload estimation, The wrong staff arrange</td>
<td>The pressure on the project time limit</td>
</tr>
<tr>
<td>The lack of necessary communication</td>
<td>High error rate, Poor understanding of the objectives</td>
<td>The pressure on the project time limit, Poor change control ability, The lack of responsibility and low morale, Lack of customer support</td>
</tr>
<tr>
<td>Lower CMMI Level</td>
<td>Lack of development norms</td>
<td>The pressure on the project time limit, The wrong criterion</td>
</tr>
<tr>
<td>Poor change control ability</td>
<td>High error rate, Heavy workload and cannot be measured, Test delay, High possibility of catastrophe losses</td>
<td>Lower production efficiency</td>
</tr>
<tr>
<td>High project complexity</td>
<td>Increasing number of exchanges, Increasing effect of other risk factors</td>
<td>All risk factors</td>
</tr>
<tr>
<td>The pressure on the project time limit</td>
<td>High error rate, High fatigue, Lack of development norms</td>
<td>Lower production efficiency, The lack of responsibility and low morale, Poor change control ability</td>
</tr>
<tr>
<td>Immature development technology</td>
<td>High error rate, Poor quality assurance, Wrong processing methods</td>
<td>The pressure on the project time limit, Lower production efficiency</td>
</tr>
</tbody>
</table>
3. The reasons of using Bayesian network

The risk analysis methods can usually be divided into qualitative analysis and quantitative analysis.

Qualitative risk analysis approach to define the source of risk, and give an imprecise numerical analysis on the possibility or the loss of the risk, then with a comprehensive evaluation of the two above-mentioned dimension in different ways, the division of the risk level is made to measure the risk’s size and importance. Next we can separate the different risks into different levels so that some measures can be taken to deal with the risks according to their risk levels. At present, methods of qualitative risk analysis commonly used include Risk Assessment Code (RAC), Total Risk Exposure Code (TREC), and Short-Cut Risk Assessment Method (SCRAM) and so on.

Based on qualitative analysis, quantitative analysis methods give the quantitative indicators of risk factors and probability of occurrence by mathematical methods or algorithms. Commonly used methods of quantitative analysis include Monte Carlo Simulation Approach (MCSA), Venture Evaluation Review Technique (VERT), Fault Tree Analysis (FTA), Influence Diagram (ID), Probability Risk Assessment (PRA), Dynamic Probability Risk Assessment (DPRA), Artificial Neural Network (ANN), Bayesian network (BN) and so on.

Clyde Chittister (1994) proposed three questions about software risk analysis to answer: Where is the problem? What is the probability of occurrence? What are the consequences of the problem? Basically methods of qualitative analysis can solve the first question, and give some conceptual answers to the third question. But it cannot give the specific values of quantitative indicators, so that it cannot solve the probability problems. Therefore methods of qualitative analysis can serve as a small software project risk analysis tools, or as a supporting tool for large-scale software project risk analysis.

Methods of quantitative analysis based on the criteria above-mentioned can be divided into:

- Methods of quantitative analysis Based on the tree, such as FTA, PRA, DPRA, etc. These methods can solve the above questions by using probability of events in the model to reflect the consequences. But the independence evaluation between the events can not be given using these methods. Also the factors that affect the human behavior can’t be modeling clearly.

- Methods of quantitative analysis based on the network such as VERT, ID, ANN, BN, etc. These methods are more complex than those based on the tree, but closer to the nature of relationships between things as regarding the risk factors as a system to research. However, there is a big difference between the above methods. For example, ID is on the basis of the data theory, reasoning rigorously, resulting in high reliability, but when facing network with large number of middle nodes, the reasoning process is so complex that you can not update the node’s confidence level reversely, but only from the top-level node to the target node. ANN has strong function of learning and analyzing, and is good at facing uncertain problems such as risks, with accurate result, But the training of neural network requires a large amount of sample data, and because of the invisible middle node, the results always are forecasted, and can’t be tracked or simulated.

BN can give a very good answer to the three questions proposed by Clyde Chittister. It’s closer to the characteristics of software projects than FTA or other methods, as well as to supply a gap of the above-mentioned methods at a certain extent. As a flexible method, BN can put different information and evidence in nodes. If sufficient data can be provided as evidence to support the samples, even the network topology can be changed. This is very important because real-time updating of the causal relationship created and captured during the software development process become possible. And Bayesian network has a very powerful function on reasoning and posteriori learning. Experiences can be spread according to the basic mathematical theory so that we can update the node in any direction. And in the case of missing some data this method can still maintain studying and analyzing ability. Therefore BN is more appropriate than others for software project risk analysis and evaluation.

4. The processes of the risk factor analysis based on BN

The core processes of software development can be divided into business modeling, requirements, analysis and design, implementation, test, and deployment (www.rational.com). Actually there are
many other processes in an ERP project. At the same time, there are many differences on kinds, number and origins of risk factors between different ERP projects of processes. ERP project starts from the requirements of customers. In most cases, customers’ requirements need to be induced by software developers. Requirements analysis is the process that developers recognize the feasibility and consistency of requirements of customers, and developers need to communicate with customers again and again. If there are any faults in this process, it will be expanded gradually. So we must pay attention to the risk in requirements process.

This paper builds a Bayesian network analysis model for risk factors on the base of data from the requirement process of an ERP project of automobile mould factory in China.

4.1 To build a BN analysis model of risk factors

A BN is a network of nodes connected by directed links with a probability function attached to each node. The network (or graph) of a BN is a directed acyclic graph (DAG), i.e., there is no directed starting path or ending path at the same node. If a node doesn't have any parents (i.e., there is no link pointing towards it), the node will contain a marginal probability table; If a node has parents (i.e., one or more links pointing towards it), the node contains a conditional probability table (CPT).

Formally, a Bayesian network can be defined as follows:

Definition: A Bayesian network is a pair (G,P), where G=(V,E) is a directed acyclic graph (DAG) over a finite set of nodes (or vertices), V, interconnected by directed links (or edges), E, and P is a set of (conditional) probability distributions. The network has the following property:

Each node representing a variable A with parent nodes representing variables B1, B2,..., Bn is assigned a conditional probability table (CPT) representing P(A | B1, B2, ..., Bn).

The nodes represent random variables, and the links represent probabilistic dependences between variables. These dependences are quantified through a set of conditional probability tables (CPTs): Each variable is assigned a CPT of the variable given its parents. For variables without parents, this is an unconditional (also called a marginal) distribution.

According to above-mentioned definition and survey data, the network of the risk factor of requirements process is shown in figure 2. (In this paper we select a part of topological structure of risk factor)

Figure 2. The network of risk factor of requirements process

To finish the BN model, we should assign a CPT to each node. BN have self-learning ability that can use some algorithm to get CPT. According to the quality of sample data, there are different algorithms to get CPT, such as gradient ascent algorithm (GAA), expectation-maximization algorithm (EMA). Firstly we need to define the states of nodes that are shown in table 2.
Table 2. The states of nodes in Figure 2

<table>
<thead>
<tr>
<th>Nodes</th>
<th>States of nodes</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kperson</td>
<td>Low, Medium, High</td>
<td>The level of depending on key-persons</td>
</tr>
<tr>
<td>Ktechnology</td>
<td>Yes, No</td>
<td>Lack of key-technology</td>
</tr>
<tr>
<td>Dtechnology</td>
<td>Low, Medium, High</td>
<td>The degree of maturity of development technology</td>
</tr>
<tr>
<td>CMMI</td>
<td>1,2,3,4,5</td>
<td>The level of CMMI-SE/SW</td>
</tr>
<tr>
<td>Metrics</td>
<td>Right, Wrong</td>
<td>The metrics standard for software project</td>
</tr>
<tr>
<td>Pcomplexity</td>
<td>Low, Medium, High</td>
<td>Project complexity</td>
</tr>
<tr>
<td>Scustomer</td>
<td>Low, Medium, High</td>
<td>The level of Support of customer</td>
</tr>
<tr>
<td>Communication</td>
<td>Low, Medium, High</td>
<td>The level of Communication in project team</td>
</tr>
<tr>
<td>Productivity</td>
<td>Low, Medium, High</td>
<td>Productivity</td>
</tr>
<tr>
<td>Workload</td>
<td>Small, Medium, Large</td>
<td>The size of workload</td>
</tr>
<tr>
<td>Ccontrol</td>
<td>Low, Medium, High</td>
<td>The level of Change control</td>
</tr>
<tr>
<td>Ptime</td>
<td>Low, Medium, High</td>
<td>The pressure of Project time</td>
</tr>
<tr>
<td>DefectsRate</td>
<td>Low, Medium, High</td>
<td>Defects rate</td>
</tr>
</tbody>
</table>

Secondly, we use the following method to get the initial value of CPT. If the nodes have single-parent node, according to the intensity between the nodes with parent node conditional probability of the node can be given three values. Corresponding to level 1, level 2 and level 3, the conditional probability of the nodes is 0.6, 0.7 and 0.8, that is, every level corresponds to 0.5 add 0.1. Contrary, if the parent node does not occur, the conditional probability corresponds to 0.5 minus 0.1; if the nodes have multi-parent nodes, the conditional probability corresponds to the sum of all parent nodes. Of course, this value should be normalized. For example, the network of nodes is shown in figure 3.

![Figure 3. Initial value of CPT calculation diagram](image)

In figure 3 C and E with only one parent node, D with two parent nodes, according to above-mentioned method, we can get the following results:

\[
P(C \mid A) = 0.6, P(C \mid \overline{A}) = 0.4; P(E \mid B) = 0.8, P(E \mid \overline{B}) = 0.2
\]
\[
P(D \mid AB) = 0.5 + 0.2 + 0.1 = 0.8, P(D \mid \overline{AB}) = 0.5 - 0.2 - 0.1 = 0.2,
\]
\[
P(D \mid \overline{AB}) = 0.5 - 0.2 + 0.1 = 0.4, P(D \mid AB) = 0.5 + 0.2 - 0.1 = 0.6
\]

The initial value can be used to self-learning by BN for CPT, or be used to the value of CPT directly, when sample data is scarce. Though this method is not reasoned strictly, to some extent it can reflect the survey results from the analysis model that is proposed by Anthony Kwok Tai Hui (2004).

Finally, we use EM algorithm to get CPT, because we select a simply topological structure in this paper, we use Delphi method to get CPT of nodes.
4.2 To predict the probability of risk factor

After we have built the BN analysis model, we can predict the probability of risk, which can help us to take precautions against and reduce risks in requirements process. In this paper we use Hugin Expert, which is popular software for building BN, to carry out above-mentioned process. Before ERP project starts, according to the survey data from ERP project, Figure 4 illustrate the probability of risk factor.

![Figure 4. The result of the probability of risk factors](image)

According to the results in figure 4, when ERP project is ready to carry out the requirements process, the project team, especially the project manager, must pay attention to control the project time, because the probability of \(P_{time}\) in low level is 22.31%, which suggest that project is likely to delay. At the same time the project also should pay attention to defects rate, because the probability of Defects Rate in low level is 34.28%, which suggest that the defects are probable to appear.

4.3 To track the change of the probability of risk factor in real time

We use prior knowledge to predict the probability of risk factor, so it can direct our work and take steps to respond the risk before the project start. For the flexibility of BN, we can update the BN in real-time with the data from the project. For example, after ERP project starts, project team find project change cannot be controlled well, this means that the probability of \(C_{control}\) in Low level is 1. After updating, the result of the probability of risk factor is shown in figure 5.

![Figure 5. The real-time result of the probability of risk factors](image)

Comparing with figure 4, we can find the probability of \(P_{time}\) in low level reduce from 22.31% to 17.06%, and the probability of Defects Rate in low level reduce from 34.28% to 25.68%, this means that the project has to delay and the user will not be satisfied with the quality of the project delivery. If the project team take steps to control the project change strictly, and at the same time they increase the productivity to high level, the results is presented in figure 6.
Comparing with figure 5, the probability of Ptime in low level rise from 17.06% to 52.91%, and the probability of DefectsRate in low level rise from 25.68% to 41.82%. The project is probable to be finished in time and provide the higher quality delivery by improving the level of productivity and change control. Therefore, using BN, we can know about the result of the measures that we have taken, or the efficiency of all kinds of measures, which can help us make a decision.

4.4 To find the causes for risks

During the implement of the project, we usually face all kinds of problem, so we usually rack us brains to find the origin of the problem. Because BN can calculate the posterior probability, BN can help us to find the root of problem. For example, ERP project team is confronted with the problem of the high defects rate. This means that the probability of Ptime in high level is 1. Using BN, The results are presented in figure 7.

Comparing with figure 4, when the defects rate is in high level, the probable reasons come from the low level of project change control or scope management. Because the result of the probability of Control in low level rise from 37.92% to 50.11%, and at the same time the result of the probability of Workload in large level rise from 28.88% to 37.46%. Therefore, according to the change in figure 7, ERP Project team can find out some key risk factors to take measures. BN is a tool that can be used to find out the causes for occurred risk.

5. Conclusions

As an effective mathematical model on probability inference, Bayesian network has many advantages in controlling risks of software projects. We can use priori knowledge to identify the probability of risk, and then establish measures to deal with before project starting. In the implementation process, the real-time risk analysis is essential to estimate the impact on project time-limit, quality, etc. At the same time, we can evaluate the effect of optional measures to determine the best decision-making. Taking advantages of Bayesian network’s superior ability on posterior analysis, combined with the implementation situations, we can rapidly identify and analyze the risk factors resulting in the project delay, decline of quality or cost overruns, and make measures quickly. In addition, Bayesian network topology structures are fit for the changing characteristics of risk factors
of software projects due to their easy expansibility and strong self-learning ability. So applying Bayesian network to analyze the risk factors of software project is more suitable than other methods, and the results from this method are much fit for the real situation. Actually the above conclusions have been proved correct in practical applications.

6. References