Risk Identification Using Case Based Reasoning in Software Project

Eunjin Chun*, Jongdae Han, Hyuksoo Han
Depart of Computer Science, Sangmyung University. 20 Hongjimun 2-gil, Jongno-gu, Seoul, Korea.

* Corresponding author. Tel.: +820232178705; email: hshan@smu.ac.kr
Revised manuscript submitted July 11, 2017; accepted October 12, 2017.
doi: 10.17706/jsw.12.9.744-750

Abstract: Software project has many uncertain factors and risk management has been recognized as one of key activities for project success. However, much of the present research focuses on finding the relation between risk factors and project outcome. Software project failures are often a result of insufficient and ineffective risk identification process. Many parts of identifying risks are subject to the knowledge and experience of expert and manual activities. Consequently, risk identification process can be time consuming and error prone.

To overcome these problems, a mechanism was proposed that uses CBR (Case Based Reasoning) to facilitate the reuse of past experience and lesson learned in similar projects. CBR is useful for extracting risk cases having high similarity with the target project in risk database. For the purpose of supporting the proposed mechanism, a descriptor which characterizes and represents a project is designed and an improved algorithm for comparing project similarity is provided. An illustrative example is presented to show how the proposed mechanism can be applied to the actual projects.

Key words: Case based reasoning, risk identification, risk management, risk database.

1. Introduction

Software project has many uncertain risks including ambiguous requirements, different skill levels of developers, and project invisibility in itself. Risk management identifies and manages potential and anticipated project risks in early stage of project. Controlling and mitigating the risks has been recognized as one of the key activities for project success [1]. However, much of the present research focuses on simply finding the relation between risk factors and project outcomes [1], [2] leaving little research on risk identification. In the present practice field, project managers and project stakeholders play an important role in identifying the risks based on their subjective opinions and knowledge. It is often time consuming and inconsistent, and even hard to ensure the reliability of the decision they've made.

Some organizations utilize a risk database to store and compare past experience. But since the structure of database is constructed only with a simple classification scheme based on project source and category, there is a limit for finding appropriate risks to target project efficiently. In this study, a systematic mechanism was proposed that utilizes CBR (Case Based Reasoning) method for finding and reusing the risks out of previous project practices. In CBR, a set of candidate risk is retrieved from risk database based on the project similarity and a reliable and accurate similarity measurement is important in the identification of risks.

In this study, a modified cosine similarity was developed which is calculated from project properties and

values and the degree of project attribution in the past project. For the purpose of supporting the proposed mechanism, a descriptor which characterizes and represents project is designed and also an algorithm for calculating similarity out of those descriptors is provided. Its usefulness was also illustrated through a case example of actual software project.

2. Related Research

2.1. Risk Identification Process

For the first step in the risk management, risks are identified and added to the known risk list. The output of the step is a list of specific risks that have potential impacts to the current project. A risk database is constructed with the classification scheme based on the project sources. As in Fig. 1. the candidate risks are compared and reviewed for reuse with the risks from risk database. Once identified, they are added to the final risk set with or without modification. New risks also should be included in the final risk set.

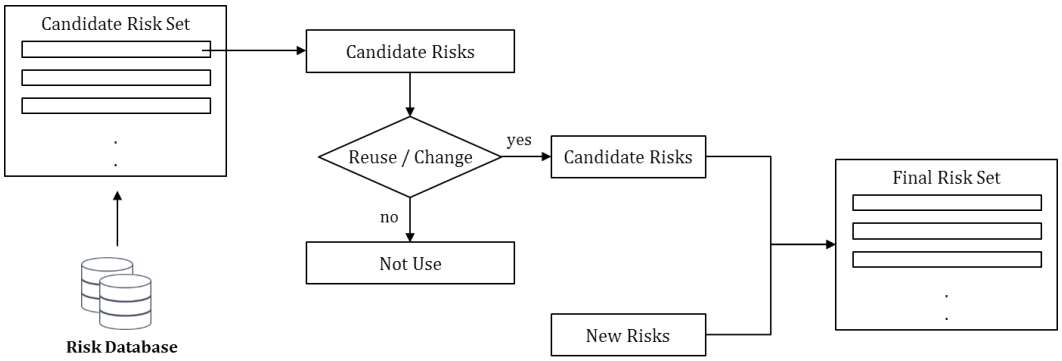

Fig. 1. Risk identification process in risk DB.

2.2. CBR (Case Based Reasoning)

CBR provides solutions that are derived from previous solutions which have close similarity to target project. This method is known to be useful in the area where the problems are difficult to formalize and the solution is merely depended on experts' knowledge. CBR enables the analyst to save time by reusing the previous knowledge and lesson learned. Key success factor of this method is to find the similarity metric between the two problems. The accuracy and correctness of the similarity metric is important for the reuse of appropriate knowledge.

3. The Proposed Risk Identification Process

The goal of our study is to develop a systematic framework for the software risk management process, especially applicable to medium-to-small sized organization. To help managers and analysts identifying risks, a risk database should be developed to maintain the identified risks. Projects are sorted by their similarity with now-to-started project and analyst is provided with identified risks from the project. The similarity score can be modified with weight factors provided by experts.

As various projects are performed by many organizations, many kinds of risk data is piled on risk database. These data should be managed and maintained as properties or assets of projects, thus being able to be used for possible risk suggestions for a new project. Risk probability is generally decided by project properties such as resource constraints, technical difficulties, and etc. Fig. 2. shows identification process described in our study.

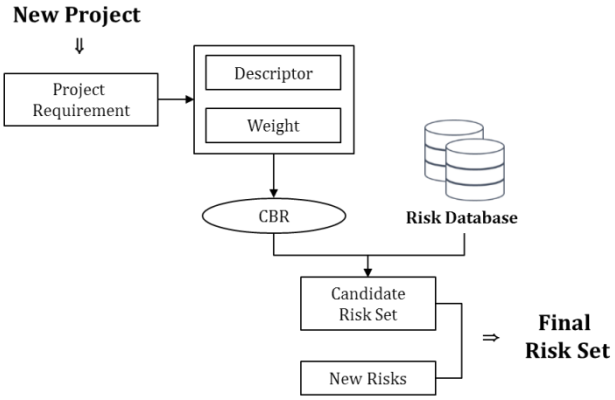

Fig. 2. Risk identification process.

3.1. Project Descriptor

A set of descriptor should be defined to compare with the previous projects and to find out the most similar projects. Because probable risks are introduced by various project properties, project property taxonomy can be adopted to define the descriptors. These descriptors will be act as a knowledge base for the further project performance. Our study synthesizes and reconstructs the project property from various studies [3]–[5] and proposes a risk description system as shown in Table 1. [6]–[11].

The proposed descriptor is defined as follows:

Project descriptor = <Category, Property Name, Property Value, Type>

Table 1. Project Property Descriptor

Category	Property Name	Property Value	Type
Problem Factors	Type of Problem	Application, COTS, Component, Data-centric, Control-centric, Embedded, Web, Others	Multiple Selection
	Newness of Requirement	Percentage of newly developed requirement items	Percentage
	Technical Challenge	Super high tech(less than 1 year), High tech(less than 2 year), Common tech(less than 5 year), Legacy tech(more than 5 year)	In-order
	Susceptibility to Change	High, Medium, Low	In-order
People Factors	Number of Peoples	Extremely large(more than 250), Very large(more than 125), Large(more than 25), Ordinary(more than 6), Small(less than 5)	In-order
	Level of Expertise	High(more than 4 year), Medium(more than 2 year), Low(less than 2 year)	In-order
	Experience	High, Medium, Low	In-order
	Organization	High, Medium, Low	In-order
Product Factors	Size	High, Medium, Low	In-order
	Deliverable	Strictly managed, Managed, Rarely managed	In-order
Resource Factors	Target System	Not known or propriety, Specified system(well-known but with special limitation), Generic computing system(PC or server with generic OS)	Selection
	Budget	Strictly fixed, Fixed, Flexible	In-order
	Deadline	Normal, Competitive, Time-critical, Emergency	In-order
Process & Tool Factors	Programming Language	High, Medium, Low	In-order
	Methodology	OOP, Structural, Functional, Agile	Selection
Outsourcing	Supplier Coverage	Supplier %	Percentage
	COTS Coverage	COTS %	Percentage
	Supplier Management	Strict, Flexible	In-order
Regulation	Regal Regulation	High, Medium, Low	In-order
	Contractual Regulation	High, Medium, Low	In-order

3.2. Calculating Similarity

Descriptors are used to find out which projects are more similar to a specific project. The similarity is

calculated by the cosine similarity metric, because it gives the better intuition between two projects and is easier to perform clustering compare to other similarity measures. Each property item is converted to a vector component, a vector per a project, and property values can have only the textual values rather than discrete values. Conversion logic was provided for each type of the property as shown in Table II.

Table 2. Conversion Logic

Type	Conversion Logic	Example
In-order	$10 \text{ x } \left(1 - \frac{(rank-1)}{(number-1)}\right)$ Where rank is rank of the value(the highest is always have the rank 1), and number is number of the possible values	If Medium is chosen, $10 \text{ x } \left(1 - \frac{(2-1)}{(3-1)}\right) = 5$ Therefore, answer is 5.
Percentage	$\frac{percentage}{10}$	75% => 7.5
Selection	$5 - 5 \text{ x } (\frac{1}{number-1})$ Where number is number of the possible values, except the value is identical to the value of the compared project, which yields 10 in case.	If value of the compared project is OOP, {OOP, Structural, Functional, Agile} => {10, 3.33, 3.33, 3.33}
Multiple Selection	$10 \text{ x } \frac{identical\_selection}{total\_selection}$ Where total_selection is total number of selected unique values of both project (union set) and identical_selection is the number of selected values consists the intersection set. But, compared project is fixed that 10.	If value set of the compared project is {Data-centric, Web, COTS} and that of the target project is {Web, Application}, 2.5

The formula for cosine similarity is as following:

$$Similarity = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2}\sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad (1)$$

where *Ai* and *Bi* are components of each vector. Because the value of components cannot be negative, similarity always has a value between 0 and 1. The higher value means the higher similarity between the vector A and B.

Basic cosine similarity applies the same importance weight evenly to all the vector components in project property. But when identifying risk, certain properties are more important than the others. For example, if an organization has suffered financial trouble recently, properties such as Budget and Deadline can have greater importance than COTS. Therefore, a way of giving weights was suggested to each component to reflect the relativity among them. Following is the modified cosine similarity metric:

$$Similarity_{weighted} = \frac{\sum_{i=1}^{n} {W_i}^2 A_i B_i}{\sqrt{\sum_{i=1}^{n} (W_i A_i)^2}\sqrt{\sum_{i=1}^{n} (W_i B_i)^2}} \quad (2)$$

where W_i is calculated weight factor for a project property.

```
get preliminary comparison property
for all properties:
    input weight factors that experts calculated
as a given project COMPARED PROJECT:
    for all projects in DB:
        for all properties:
            convert property value to vector component (min 0, max 10)
            calculate the product of each weight factor and vector component
        calculated weighted similarity
```

Fig. 3. Algorithm for calculating modified cosine similarity

The weight factor is given by project experts and represents the degree of importance to the risks. Because over value of weight factor may cause distortion to the result, we suggest limiting the weight factor no greater than (2). Fig. 3. shows the algorithm for the calculation of similarity.

4. Case Application

To validate our study, an example case for project risk identification is performed on the control software of automobile smart key. The following are a set of requirements about the case project:

a) The required functions will be enhanced based on the previous model.
b) It is assumed that no COTS packages are used.
c) The project requires remote ignition function to be installed. This function is considered as super high technology.

4.1. Project Descriptor

After analyzing requirements, the project descriptor of the target project is identified as in Table III.

Table 3. Descriptor of Target Project

Project Name	Newness of Requirement	Technical Challenge	Susceptibility to Change	Number of People	Level of Expertise	Experience	Organization	Size	Deliverable	Target System
Target project	30%	Super high (<1)	Medium	Large (>25)	High (>4)	High	High	High	Strictly managed	Specified system

Budget	Deadline	Programming Language	Methodology	Supplier Coverage	COTS Coverage	Supplier Management	Regal Regulation	Contractual Regulation
Fixed	Competitive	Medium	Agile	30%	60%	Flexible	High	High

4.2. Calculating Similarity

The descriptor of target project is compared with the projects in risk database. Table IV shows some of the descriptors in risk database.

Table 4. Descriptor of Some Projects in Risk Database

Project Name	Newness of Requirement	Technical Challenge	Susceptibility to Change	Number of People	Level of Expertise	Experience	Organization	Size	Deliverable	Target System
Project A	85%	Common (<5)	Low	Extremely large(>250)	High (>4)	High	High	High	Strictly Managed	Specified system
Project B	25%	High (<2)	High	Large (>25)	Medium (>2)	High	High	High	Managed	Specified system
Project C	15%	Super high (<1)	High	Small (<5)	Low (>0)	Low	High	Low	Rarely Managed	Specified system
Project D	65%	Legacy (>5)	Low	Very large (>125)	Medium (>2)	Medium	High	Medium	Managed	Specified system

Budget	Deadline	Programming Language	Methodology	Supplier Coverage	COTS Coverage	Supplier Management	Regal Regulation	Contractual Regulation
Strictly fixed	Time-critical	Medium	Agile	70%	55%	Strict	High	High
Fixed	Competitive	Medium	Agile	40%	60%	Flexible	High	High
Flexible	Normal	Medium	Agile	85%	65%	Flexible	Medium	Medium
Flexible	Normal	Medium	Agile	65%	50%	Flexible	Medium	Medium

Then the modified cosine similarity measure is applied and calculated using the algorithm defined in Fig. 3. With given requirements, our experts estimated weight factors as following:

[1,0,0.5,1,2,2,1,1,1,2,1,0.5,1,1,1,1,1,0.5,1,1]

For each corresponding project properties described above, respectively. Table V shows how modified

cosine similarity made difference with naive cosine similarity. The result suggests that Project A is more appropriate candidate for reusing identified risks than Project B.

Table 5. Project Similarities with Target Project

Projects in Risk DB	Cosine Similarity	Modified Cosine Similarity
Project A	0.9032	**0.9544**
Project B	**0.9659**	0.9417
Project C	0.7533	0.5923
Project D	0.8596	0.8931

5. Conclusion

In this paper, a mechanism was proposed to use CBR to retrieve the risk candidates from risk DB to reuse past experience and lesson learned in similar projects. A set of project properties was suggested to find out the most similar project, and set up an algorithm to obtain similarity.

The proposed approach has several advantages. First, CBR enables more efficient retrieval of risks from similar past project. It provides a great opportunity of reusing knowledge and lesson learned of past projects.

Second, analyzing the target project based on project descriptor provides more systematic and thorough examination of the project. It also reduces the possibility of missing some critical risks.

Third, modified cosine similarity can improve accuracy and correctness of similarity which will play an important role in the success of using CBR.

For the future work, we will support entire risk identification process and automate the initial risk candidates from risk Database for efficiency of risk identification.

Acknowledgment

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2016-R0992-16-1014) supervised by the IITP(Institute for Information & communications Technology Promotion)

References

[1] Hu, Y., *et al*. (2013). Software project risk analysis using Bayesian networks with causality constraints. *Decision Support Systems*.

[2] Avdoshin, S. M., *et al*. (2016). *Software Risk Management: Using the Automated Tools*. *In Emerging Trends in Information Systems.* Springer international publishing, 85-97.

[3] Kang, D. W., *et al.* (2011). Knowledge-based process tailoring automation. *Journal of KIISE: Software and Applications*, *38(6)*, 304-316.

[4] Park, Soo-Jin, *et al.* (2006). A process tailoring method based on artificial neural network. *Journal of KIISE: Software and Applications, 33(2)*, 201-219.

[5] Kim, Woo-Ri, *et al.* (2010). The research on applying FMEA to evaluate the safety of tangible game-focusing on Wii accident cases. *Journal of Korea Game Society, 10(3)*, 25-35.

[6] Yu, W. A., *et al.* (2003). Knowledge and case-based reasoning for customization of software processes - A hybrid approach. *International Journal of Software Engineering and Knowledge Engineering, 13*, 293-312.

[7] Ginsberg, M. P., *et al.* (1995). *Process Tailoring and the Software Capability Maturity Model (sm). No.*

CMU/SEI-94-TR-024. Carnegie-mellon univ pittsburg pa software engineering inst.

[8] Shenhar, A. J., *et al.* (1996). Toward a typological theory of project management. *Research policy, 25(4)*, 607-632.

[9] Jalote, P. (2000). *CMM in Practice: Processes for Executing Software Projects at Infosys.* Risk Managemet (pp. 159-174). Addison-Wesley Professional.

[10] Daniel, S. (2001). Software acquisition management guidelines. *Thesis. Submitted for the Degree of. Master of Science*. Linkoping University, Sweden.

[11] The Ministry of Information and Communication. (2005). *Standard Software Acquisition Processes for Public Sectors.* TTAS, Korea.

Eunjin Chun is a senior student in the Department of Computer Science at SangMyung University. She will enter M.S. program in 2017 at SangMyung University, Korea. Her research interests are SW engineering, MSR(Mining Software Repository) and risk management.

Jongdae Han received his computer science and engineering B.S. in 2005 and Ph.D. in 2013 from Seoul National University, Korea. He is currently an assistant professor in the Department of Computer Science at SangMyung University, Korea. His research interests are in team composition for software development, distributed software development, and repository data mining.

Hyuksoo Han received his computer science B.S. in 1985, M.S. in 1987 from Seoul National University, Korea and Ph.D. in 1992 from University of South Florida. He is a professor in the Department of Computer Science at SangMyung University, Korea. His current research interests are in the areas of SW process, SW safety and risk management. He is currently the director of SSARC (Software Safety Assurance Research Center).