DRAFT

full paper published in:

proc. of 17th International Conference "Software & Systems Engineering and their Applications"
November 30 - December 2, 2004 Paris, France

Paper published in the proceedings and presented at the conference

Identifying Software Project Risks with the Process Model

Jakub Miler*, Janusz Górski

Department of Software Engineering
Gdansk University of Technology
G. Narutowicza 11/12, 80-952 Gdansk, Poland
tel. +48 58 347-14-64, 347-19-09
fax. +48 58 347-27-27
{jakubm, jango}@eti.pg.gda.pl

Abstract

The paper presents a process model-based approach to software project risk identification. The approach involves explicit modeling of software processes and identifying risk by two dedicated techniques. The paper introduces a meta-model that allows expressing the process risk. Then, two systematic risk identification techniques are proposed and presented in the form of detailed procedures. Both techniques refer to the process model to focus the analyst's attention in the investigation of the process risks. The first technique uses metrics of process structure while the second focuses on the differences between the actual and the referential model. Further on, the paper reports on the case study of practical application of the proposed approach in a real-life software project. The case study results demonstrate that the proposed approach helped to reveal new, previously undetected risk factors judged important by the project managers. Finally, the paper summarizes the approach as well as provides the issues open for further research.

Keywords: project risk management, software project risk, risk identification, process modeling, risk metrics

* Research partially supported by the grant no. 4 T11 C 008 25 from Polish Research Council (KBN)

1. INTRODUCTION

The aim of any software project is to provide the stakeholders with a satisfactory solution of their problem within the schedule and budget limits. The risk of poor product quality and schedule or budget overruns is high which is confirmed by a number of cancelled, delayed or overpaid projects. Effective management of those risks is presently perceived as one of the most important areas of project management [1, 9]. Current risk identification practices adopt primarily two techniques: checklists and group effort (e.g. brainstorming). Checklists such as [3, 5, 12] help to control the identification scope and protect from overlooking significant risks but they are often too general and do not relate well to actual software processes. Group effort studied e.g. by J. Kontio [4] benefits from synergic use of human intuition and experience but it exhibits problems with scope focusing and control.

The paper proposes a model-based approach to process risk identification. The approach is characterized by the following features:

- explicit *process modeling* as well as providing for *model evolution*,
- interpretation of *model deficiencies* as process risks,
- supporting risk identification by referring to *model metrics* and consulting *referential models*,
- modeling risks with *risk patterns* [6].

The paper presents a meta-model focusing on modeling software processes together with risk-related information. Then it introduces two techniques of risk identification based on process model. Finally, the paper reports on the case study carried out to assess the effectiveness of the proposed approach.

The results presented in this paper originate from a wider context of research towards a holistic approach to identification and analysis of risk in software projects supported by dedicated tools. The description and the results of the research are available at [11].

2. MODELLING THE SOFTWARE PROCESS

The SPEM meta-model (Software Process Engineering Metamodel) proposed by OMG (Object Management Group) [8] has been adopted as a basis for modeling the development processes. The SPEM meta-model defines three fundamental elements of model structure:

- *Activity* – an action animated by a human(s) in a certain role(s) processing input artifacts into output ones,
- *Role* – a function, responsibility of a human animating the activity (an individual can act in many roles),
- *Artifact* – an item processed by activities (e.g. input materials, documents, tools and output products).

However, talking about risk requires extending the basic activity-artifact-role meta-model with entities that describe the desired qualities of the aforementioned elements:

- *Practice* – the practice followed in order to complete the activity,
- *Capability* – experience, skill, ability of a human in a given role,
- *Feature* – quality aspect of a given artifact.

Moreover, activities, artifacts and roles can be decomposed recursively. This provides for specifying the software development process to the level of detail that corresponds to the current risk identification perspective. The resulting meta-model of a software development process is presented in Fig. 1 as a UML class diagram. This meta-model is an extended version of meta-model proposed in [6].

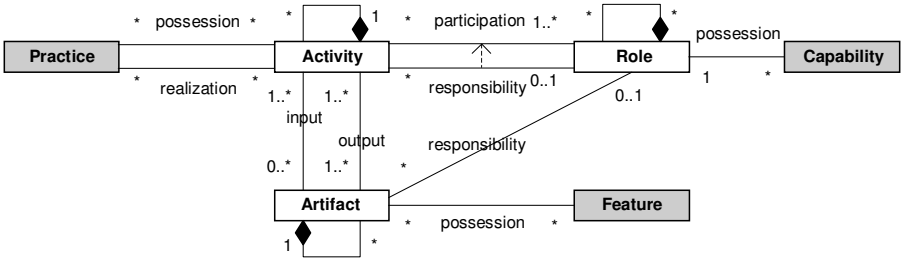

Fig. 1. Extended software process meta-model

As processes evolve naturally as well as due to deliberate improvement or reengineering effort it is necessary to provide for incorporating change in process models. M. L. Jaccheri and R. Conradi suggest that process models follow the two kinds of transformation normally used in configuration management: *revision* and *variant* [2]. To support traceability and comparative analyses of evolving process models, it is essential to preserve a mapping of model elements while transforming the models. To achieve this objective, the revision and variant concepts have been particularly interpreted and implemented.

Revision of a process model assumes that the mappings between original and revised model elements act similarly to inheritance known from object-oriented modeling. Moreover, elements left intact are included in a revised model by sharing them with the original model. This way, any change to original model affects automatically the models being its revisions.

Variant of a process model involves copying the original model with preserving mappings between source and copied model elements and subsequent unrestricted customization of the copy. Consequently, any changes to the original model do not affect its variants in any way.

Building a model of a software project from scratch is quite occupying. The concepts of revision and variant may be used to help to derive the model of actual development process from a chosen referential model.

3. IDENTIFYING PROCESS RISKS

Two techniques of risk identification based on the analysis of a process model are proposed:

- metrics of process structure,
- comparison to the referential model.

The first technique uses metrics of process structure to focus analyst's attention on the most important and most risky elements of the process. The second technique focuses on the differences between the actual and the referential models.

The two techniques are dedicated to different situations, but they can also be used complementarily. The first does not require having any referential model, which makes it easier and more universal to apply. The second technique works better in case of having a high-quality referential model.

3.1. Metrics of process structure

This technique involves finding out, by means of metrics, the most important and most risky elements of the process that are further investigated with a set of generic questions. Procedure of risk identification includes two steps detailed below.

3.1.1. Step 1: Calculating model metrics

The following sets were defined based on the relations between meta-model elements shown on Fig. 1:

- $A_{in}(Ar)$ – set of activities that have artifact Ar as input,
- $A_{out}(Ar)$ – set of activities that have artifact Ar as output,
- $A_{part}(R)$ – set of activities in which role R participates,

- $Ar_{in}(A)$ – set of input artifacts of activity A,
- $Ar_{out}(A)$ – set of output artifacts of activity A,
- $R_{part}(A)$ – set of roles participating in activity A,
- P(A) – set of practices possessed by activity A,
- F(Ar) – set of features possessed by artifact Ar,
- C(R) – set of capabilities possessed by role R.

The following metrics are calculated to find out which model elements further identification effort should focus on. In the following definitions, the symbol |A| denotes the cardinality of set A.

1. I(A) – importance of activity A

a) total importance of all output artifacts of activity A

$$I(A) = \sum_{Ar \in Ar_{out}(A)} I(Ar), \quad I(A) \in N \tag{1}$$

b) average importance of output artifact of activity A

$$\overline{I(A)} = \frac{I(A)}{\left|Ar_{out}(A)\right|}, \quad \overline{I(A)} \in R_+ \cup \{0\} \tag{2}$$

c) number of input artifacts of activity A

$$I(A) = \left|Ar_{in}(A)\right|, \quad I(A) \in N \tag{3}$$

2. I(Ar) – importance of artifact Ar

- number of activities that have artifact Ar as input

$$I(Ar) = \left|A_{in}(Ar)\right|, \quad I(Ar) \in N \tag{4}$$

3. I(R) – importance of role R

- number of activities in which role R participates

$$I(R) = \left|A_{part}(R)\right|, \quad I(R) \in N \tag{5}$$

4. R(A) – risk of activity A

- importance of activity A divided by number of practices of activity A

$$R(A) = \frac{I(A)}{\left|P(A)\right|}, \quad R(A) \in R_+ \cup \{0\} \tag{6}$$

5. $R_{in}(Ar)$ – risk of input artifact Ar

- importance of artifact Ar divided by number of features of artifact Ar

$$R_{in}(Ar) = \frac{I(Ar)}{\left|F(Ar)\right|}, \quad R_{in}(Ar) \in R_+ \cup \{0\} \tag{7}$$

6. $R_{out}(Ar)$ – risk of output artifact Ar

a) risk of developing artifact Ar by activity A is a number of features of artifact A divided by total number of features of all input artifacts of activity A plus number of practices of activity A plus total number of capabilities of all roles participating in activity A

$$R_{out}(Ar, A) = \frac{\left|F(Ar)\right|}{\sum_{Ar' \in Ar_{in}(A)} \left|F(Ar')\right| + \left|P(A)\right| + \sum_{R \in R_{part}(A)} \left|C(R)\right|}, \quad R_{out}(Ar, A) \in R_+ \cup \{0\} \tag{8}$$

b) total risk of development of artifact Ar is the risk of development of artifact Ar summed for all activities that have artifact Ar as output

$$R_{out}(Ar) = \sum_{A \in A_{out}(Ar)} R_{out}(Ar, A), \quad R_{out}(Ar) \in R_+ \cup \{0\} \tag{9}$$

c) average risk of development of artifact Ar is the average risk of development of artifact Ar for all activities that have artifact Ar as output

$$\overline{R_{out}(Ar)} = \frac{R_{out}(Ar)}{|A_{out}(Ar)|}, \quad \overline{R_{out}(Ar)} \in R_+ \cup \{0\} \tag{10}$$

7. R(R) – risk of role R

- importance of role R divided by number of capabilities of role R

$$R(R) = \frac{I(R)}{|C(R)|}, \quad R(R) \in R_+ \cup \{0\} \tag{11}$$

3.1.2. Step 2: Studying the context of process elements

For the most risky model elements (according to the metrics calculated in previous step) the following questions about their context are asked (the questions refer to the terms introduced in section 2):

- Do you have *capability* and *input artifact* to perform *activity*?
- Do you have *activity*, *practice* and/or *capability* to acquire *input artifact*?
- Do you have *input artifacts* for *output artifact*?
- Do you have *practice* and/or *capability* to develop *output artifact*?
- Do you have *practice* and/or *capability* to build *feature* into *output artifact*?
- Do you have *capability* and *input artifact* to realize *practice*?
- Do you have *activity*, *practice* and/or *capability* to assign *role*?
- Do you have *practice* and/or *capability* to build *capability* into *role*?
- Do you have *role* responsible for *artifact*?
- Do you have *role* responsible for *activity*?
- Do you have *artifact* (e.g., guidelines, standards, acts, literature) defining *activity*, *artifact*, *role*, *practice*, *feature*, *capability*?

Negative answer suggests increased risk related to a given model element and indicates a risk factor for the entire project. To clearly specify the context of risk, the risk factor may be expressed with risk patterns [6].

3.2. Comparison to referential model

This technique is based on comparison of the analyzed model elements to semantically equivalent elements of the referential model. The result is a lists of missing and superfluous elements that indicate possible risk factors. The procedure of risk identification is detailed below.

3.2.1. Step 1: Mapping the analyzed model at the referential model

This step involves a mutual mapping of semantically equivalent elements of both models. In general, such a mutual mapping is a many-to-many relationship. It is further restricted by the assumption that the mapped elements must be of the same type (an activity can be mapped only at an activity, an artifact at an artifact and so on). If the analyzed model were built independently from the referential model the mapping is to be developed from scratch. However, the mapping is much easier to define if the analyzed model were created as a revision or variant of the referential model, as proposed in section 2.

3.2.2. Step 2: Finding differences between models

In this step all elements and relationships of both analyzed and referential model are examined. Identified differences are collected on the following lists:

- list of missing activities, artifacts, roles, practices, features and capabilities,
- list of missing relationships between model elements,
- list of superfluous activities, artifacts, roles, practices, features and capabilities,
- list of superfluous relationships between model elements.

The differences noted in the lists indicate the potential project risk factors. Analogously to the technique of risk identification based on metrics, the context of risk may be detailed with risk patterns [6].

4. CASE STUDY

4.1. Methodology

The case study aimed at repeated identification of risks in a real-life software project with the proposed approach. A software project involving participants from several countries and scheduled for over a dozen months has been chosen for the case study. The project objective was to build a complex, distributed information system based on a novel architecture and business model. During the case study the project remained in the initiation phase. The project description and plans were used in the case study.

The case study comprised 2 phases:

- Preliminary risk identification carried out in January 2004 based on formal project description,
- Second risk identification with respect to the improved process carried out in April 2004 based on the Quality Plan, partial Development Plan and the same project description as in Phase 1.

The case study was evaluated by:

- subjecting the risk identification results to the judgment of the project managers based on their intuition and individual experience,
- examining the ratings and priorities the project managers assigned to the identified risks at a risk analysis session,
- assessing the scope of improvement initiated by the project managers after the risk identification.

4.2. Results of Phase 1

To begin with, a process model following the meta-model defined in section 2 was built based on the project description. Due to the initial phase of the project, the development process was planned at a rather general level. The most detailed activities covered several months. The project description did not define any qualitative features of the activities, artifacts and roles as those elements were left to be defined later in the plans of the particular project areas. The final model comprised 55 activities at 3 levels of detail, 49 artifacts and 45 roles.

The process risks were first identified using model metrics. Of all metrics presented in section 3.1.1 only the metrics of importance were applicable to the model. The metrics given by equations 3 and 4 indicated two activities and five artifacts for further investigation which resulted in identification of four risk factors. For all those factors, their scenarios have been developed with the help of risk patterns [6]. The example of an identified risk factor is given below together with the corresponding scenario (first expressed with the help of risk patterns, then expressed as a natural language statement).

Risk factor: *Distributed, multinational development of business models*

Risk scenario (in terms of risk patterns): If **New Business Modeling**<*activity*> loses **Consider regional differences in reality**<*practice*> then **System Requirements Specification (Vision)**<*artifact*> loses **Conformity to target reality**<*feature*> and **Use Case Design**<*artifact*> loses **Conformity to target reality**<*feature*> and then **Pilot One**<*artifact*> loses **Conformity to target reality**<*feature*>.

Risk scenario (natural language): *Business modeling is skewed by local viewpoints and results in missed target reality of the pilot implementation of the system.*

The risks were further identified by comparing the analyzed model to the Rational Unified Process (RUP) [10] taken as a referential model. RUP was particularly chosen as being well structured, defined in detail yet generally applicable and finally compatible with the development process of the studied project. Due to the limited resources for the case study, a complete mapping of the analyzed model on the RUP referential model was not developed. Instead, the most evident differences were taken into consideration. This way, three additional risk factors were identified. One of them is given below together with its exemplary scenario.

Risk factor: *Configuration & Change Management is not explicitly defined*

Risk scenario (in terms of risk patterns): If **Configuration & Change Management**<*activity*> is not performed then **System Integration**<*activity*> loses **Keep the set of integrated subsystems coherent**<*practice*> and then **Pilot Deployment**<*activity*> takes more time than expected.

Risk scenario (natural language): *Without explicitly defined change management process the pilot may not be integrated and deployed on time.*

The risk identification step was completed by comparing the analyzed model to the referential model derived from the Steve McConnell's 'Complete List of Schedule Risks' [5, 6]. Following the same procedure as in the previous step, four additional risk factors were identified. One of them is given below together with the exemplary scenario.

Risk factor: *Long duration of the project and mainly part-time employment of the staff.*

Risk scenario (in terms of risk patterns): If **Project**<*activity*> loses **Maintain personnel continuity**<*practice*> then **Project**<*activity*> loses **Personnel**<*role*> and then **Project**<*activity*> loses **Avoid excessive schedule pressure**<*practice*> and **Pilot One**<*artifact*> loses **Completeness**<*feature*>.

Risk scenario (natural language): *The project can have difficulties with keeping part-time employed staff resulting in workforce shortages, more effort for available staff and limited scope of the pilot.*

In total, 11 risk factors were identified in Phase 1 of the case study. The results were then compared to the 9 risk factors indicated in the project description (identified by the project management). 6 out of the 11 risk factors identified in Phase 1 of our case study were also indicated in the project description. Still 5 of them were new regarding the project description.

Partial correlation of the detected risk factors with the factors indicated earlier by the project management confirms that the proposed method is consistent with the intuition and experience of the managers of software projects. The 5 new risks were then communicated to the Project Management Board which judged them important and initiated activities aiming at the process improvement. The process was improved by defining its deficient areas in detail in the newly issued documents and initiating new activities related to the redefined process.

4.3. Results of Phase 2

The second risk identification attempt focused on the managerial issues such as operational management, quality management, communication management, software management and so on. A partial model of those areas was built from the available data. The model comprised 85 activities at 3 levels of detail, 16 roles and 37 artifacts.

Due to incompleteness of the process model the risk identification technique using model metrics could not be effective. Instead, the technique of comparison to a referential model was applied. For the same reasons as in Phase 1, the Rational Unified Process (RUP) [10] was selected as a primary referential model.

As a result, some 26 risk factors were identified. The examples of identified risk factors in particular managerial areas are given below together with possible scenarios.

- Quality management – establishing success criteria and metrics

Risk factor: *Immeasurable success criteria, weak metrics*

Risk scenario (in terms of risk patterns): If **Measurement Plan**<*activity*> loses **Use quantifiable and objective metrics**<*practice*> then **Pilot One**<*artifact*> loses **Defined scope**<*feature*>and then **Pilot One**<*artifact*> loses **Completeness**<*feature*>.

Risk scenario (natural language): *Product scope is arbitrarily redefined and intended product scope is not covered.*

- Document Management – general risks

Risk factor: *No explicitly defined procedures for maintaining traceability of key business and design decisions*

Risk scenario (in terms of risk patterns): If **Document Management**<*activity*> loses **Maintain traceability of key decisions**<*practice*> then **Documentation**<*artifact*> loses **Consistency**<*feature*> and then **Subsystem**<*artifact*> loses **Compatibility**<*feature*>.

Risk scenario (natural language): *Key business and design decisions are vague, inconsistent or contradictory in different documents resulting in incompatibility of partial commitments.*

- Communication Management – communication by project portal

Risk factor: *Poor performance of hardware and software platform of project portal*

Risk scenario (in terms of risk patterns): If **Project portal**<*artifact*> loses **Platform performance**<*feature*> then **Personnel**<*role*> loses **Motivation**<*capability*> and **Communication**<*activity*> loses **Follow defined communication paths**<*feature*> and then **Communication**<*activity*> takes more time than expected.

Risk scenario (natural language): *Discomfort in portal usage causes users' rejection and hasty construction of alternative communication means that impacts communication.*

The identified risk factors were taken as input to the risk analysis session carried out by the Project Management Board - PMB (some risk factors were merged together which resulted in total of 20 risks). The list of risks was also extended with the risks identified independently by PMB members. The risk analysis session involved rating and prioritizing the identified risks by some 15 members of the PMB. As a result, a list of project top 10 most important risks was elaborated. Of those 10 risks 7 were identified with the help of the proposed approach.

4.4. Summary

All the analytical tasks in the case study were carried out by one of the authors. However, the key part (risk identification) was performed according to precisely defined procedures of the risk identification techniques. Thus it may be presumed that every risk analyst would obtain comparable results. However, a complete correlation (reproduction) of the results is difficult to achieve, because while being considerably formalized the proposed approach still engages the analytical capabilities and the individual perception of the analyst.

An effort on every phase of the case study was carefully measured. Phase 1 required 30 man-hours, 22 of which were spent on initial building of the process model while risk identification took 2-3 hours for each of the techniques and referential models. Phase 2 took 8 hours, 3 of them used to model the managerial areas of the project and the remaining 5 to identify project risks. The effort of the risk identification steps may be reduced by taking advantage of the already built dedicated supporting tool [7], which is currently adapted to the proposed approach.

5. CONCLUSIONS

Software projects constantly involve considerable risk which remains undetected and decreases their chance of success. Current risk identification techniques of checklists and group effort such as brainstorming exhibit limitations and they could be supplemented with new techniques referring explicitly to the process structure.

The paper presented a process model-based approach to software project risk identification that involves explicit modeling of software processes and identifying risk by two dedicated techniques. For modeling purposes, the paper proposed an extension of SPEM meta-model (Software Process Engineering Metamodel) from Object Management Group that allows expressing the process risk. Two systematic risk identification techniques refer to the process model to focus the analyst's attention in the investigation of the process risks. It provides for controlling the scope of the analyses and ensures their feasibility in reasonable time (both techniques are subject to automated tool support).

The approach was validated in a two-phase case study. The results of the case study demonstrate that the proposed approach helped to reveal new, previously undetected risk factors judged important by the project managers. The approach requires that the process is defined in the level of detail sufficient to build its model which is then used during risk identification. A high-quality relevant referential model is also necessary if the model comparison technique is to be used. As the proposed approach is based on models and their metrics we expect the delivered results to be highly independent of the analyst's intuition and experience. It also provides a good base for automatic tool support.

Plans for further research include additional case studies with industrial partners, investigation of new metrics to improve risk identification, building the proposed techniques into a supporting tool [7] and finally using the approach in different domains such as e-health and/or e-commerce.

REFERENCES

[1] CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development, V1.1, Tech. Rep. CMU/SEI-2002-TR-004, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, PA, 2002

[2] Jaccheri M.L., Conradi R., Techniques for Process Model Evolution in EPOS, IEEE Trans. on Soft. Eng., Vol. 19, No. 12, 1993, pp. 1145-1156.

[3] Jones C., Assessment and Control of Software Risks, Yourdon Press, New Jersey, 1994.

[4] Kontio J., Software Engineering Risk Management: A Method, Improvement Framework, and Empirical Evaluation, PhD Thesis, Helsinki University of Technology, Finland, 2001.

[5] McConnell S., Rapid Development, Microsoft Press, 1996.

[6] Miler J., Górski J., Risk Identification Patterns for Software Projects, Found. of Comp. and Dec. Sci., Vol. 29, No. 1-2, 2004, pp. 115-131

[7] Miler J., Górski J., An environment supporting risk management in software projects, proc. of I National Conference on Information Technologies, Gdansk, Poland, 2003 (in Polish)

[8] Software Process Engineering Metamodel Specification Version 1.0, Object Management Group, November 2002.

[9] PMBOK Guide, 2000 Edition, Project Management Institute, 2000.

[10] Rational Unified Process version 2002.05.00.25, http://www.rational.com/rup

[11] RiskGuide project website, http://mkzlway.eti.pg.gda.pl/RiskGuide

[12] Sisti F. J., Joseph S., Software Risk Evaluation Method, Tech. Rep. CMU/SEI-94-TR-19, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, PA, 1994.