A Risk Management Approach Based on Situational Method Engineering

Guilherme Vaz Pereira, Fabrício Severo, and Lisandra Fontoura.

Universidade Federal de Santa Maria (UFSM) – RS – Brasil
{guigavazpereira, severo.fabricio, lisandramf}@gmail.com

Abstract. Software development is a complex activity. It is involved with different scenarios and risks which must be managed through a systematic approach to Software Risk Management (SRM). The best software process depends on the project's particularities. Situational Method Engineering (SME) focuses on building project specific method/process according to the situation at hand. This paper proposes an approach to prevent software project risks through software process tailoring based on SME concepts. Preventive actions are inserted in the tailored process to prevent risks.

Keywords: risk management, process tailoring, SME, octopus model.

1 Introduction

A software process defines activities to be performed in software development, each activity's characteristics and the relationship among them [18]. There are several software process models such as RUP [15] and XP [4] however each project requires particular actions thus tailored software processes are required.

Process Tailoring can be defined as the act of adjusting or particularizes a standard process definition to an environment less general [8]. However the managers often make it in an ad-hoc fashion based only on experiences [3].

To deal with these challenges, Situational Method Engineering (SME) [11] approach focuses on the project specific method/process construction according to project features (specific situations) from method fragments stored in method base.

This paper proposes an approach to manage risks through tailoring processes call Octopus SME - Risk Management Approach (OSRiMA). This is based on SME concepts and uses a multi-criteria selection and prioritization of method fragments through Analytic Hierarchy Process (AHP) technique [16], according to the project risks and the project context.

The paper is organized as follows: Section 2 presents concepts about Risk Management. Section 3 presents SME. Section 4 describes the Octopus SME - Risk Management Approach (OSRiMA) and details about method fragments for risk prevention. Section 5 presents examples of the proposed approach. Section 6 and 7 describe related works, and conclusions respectively.

adfa, p. 1, 2011.

2 Software Risk Management

A risk is an undesirable event which can affect the project and have losses associated [9]. SMR includes practices which allow avoiding compromising time, quality, cost and functionality, among others desirable events. Thus the organizations must use a systematic and structured approach for risk management.

There are four steps in risk management [9]: (1) Risk Identification; (2) Risk Analysis; (3) Risk Planning; and (4) Monitoring and Controlling.

The Risk Identification aims to gather information about all risks which can affect the software development. The Risk Analysis prioritizes identified risks according to the probability of risk occurring and losses associated with it to determine which risks will be treated. After this, plans should describe actions to avoid the risk or to reduce impact. At least, the risks must be monitored and controlled.

In this work, we focus on the Identification, Analysis, Planning steps in the risk management using SME and multi-criteria selection and prioritization.

3 Situational Method Engineering (SME)

SME [11] proposes the building of a specific development method/process for each project according to the situational at hand.

This building is from reusable fragments, “pieces” of methods, stored in a repository call method base. These elements are typically extracted from best practices, process models, process patterns, and reference models [10]. They are retrieved from method base according to a specific situation.

For the purpose of this paper, the notion of method fragments is from [10]. It is widely supposed that a method fragment is an element generated from a metamodel usually by instantiation.

The situation can be defined as the characteristics related to organization and project [2]. It indicates which fragments are appropriate to the project at hand [10].

In short, SME revolves around identification of the best fragments according to situation and their linking together as appropriate. It is a possible solution to the problem of selecting the best process for an organization and project [11].

4 Octopus SME - Risk Management Approach (OSRiMA)

In this work we propose an approach for risk management in software projects based on SME call Octopus SME - Risk Management Approach (OSRiMA). It focuses on the steps 1, 2 and 3 of SRM (section 2).

SME focuses on a specific development process for a project according to its specific situation. In our approach the specific process building is through stored method fragments containing preventive actions in relation to the project risks. Furthermore, the project specific situation is defined by project context (Octopus Model) and project risks. Figure 1 illustrates the sequence of activities proposed by OSRiMA.

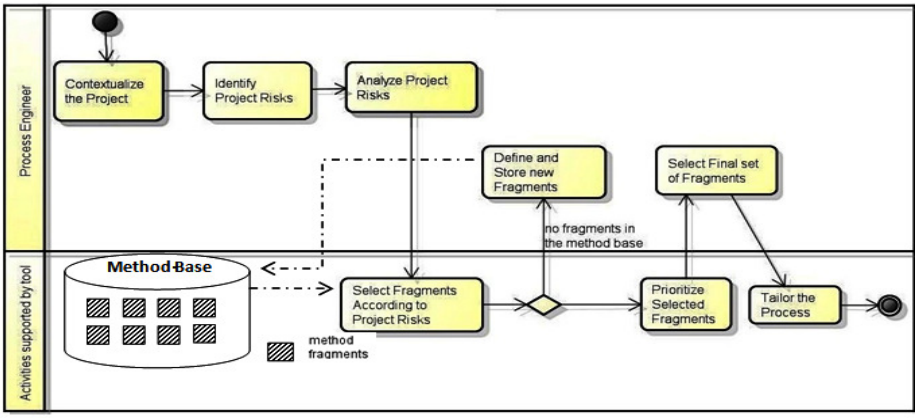

Fig. 1. Overview of the Octopus SME - Risk Management Approach.

In the activity "Contextualize the Project" the process engineer contextualizes the project according to the situational factors from Octopus Model [14]. In "Identify Project Risks", the risks that may affect the project are identified. Then in the activity "Analyze Project Risks" the risks are analyzed according to the probability of each risk be materialized.

In "Select Fragments According to Project Risks" the support tool retrieves the stored method fragments associated with each project risk. If there is not a suitable fragment for a risk, it is possible to define new elements and store them in the method base by activity "Define and Store the New Fragments".

The retrieved method fragments are prioritized according to the project context by our multi-criteria selection and prioritization support tool in the activity "Prioritize Selected Fragments". They are prioritized through a prioritization algorithm based on Analytical Hierarchy Process (AHP) [16] for each project risk. In short, the higher similarity between the fragment use-context and project-context, more appropriate and the better is the fragment score in the prioritized list for each project risk.

The goal of the activity "Prioritize Selected Fragments" is to guide the process engineer to choose the best fragments to include in the organization's standard software process. In "Selected Final Set of Fragments" the process engineer can execute the final selection of method fragments.

The support tool creates the tailored process through the activity "Tailor the Process". This activity describes how the process can be tailored to integrate the selected methods fragments to prevent the risks, resulting in the process defined for a project.

4.1 Octopus Model

Octopus Model [14] is a model for contextualizing the software development through eight factors. It provides guides about the adoption of practices recommended by agile process approaches in a development process trough the "agile sweet spot" (ideal

conditions where agile practices they are most likely to succeed). So we consider that for situation where agile practices are not advised (a context out of "agile sweet spot"), practices from planned approaches are recommended.

From the eight factors it is possible to characterize the software project context and define the method fragments use-context. The Octopus Model factors are: "Size", "Criticity", "Business Model", "Stable Architecture", "Team Distribution", "Rate of Change", "Age of System", and "Governance". Due to the limited space we are not able to describe the details of these factors.

4.2 Method Fragments for OSRiMA

Our method fragments were defined based on concepts of Rational Unified Process (RUP) [15] and ISO/IEC 24744 [12]. They are extracted from other processes models such as RUP [15], XP [4], and process patterns such as Coplien [5], among others. A fragment describes preventive actions in relation to project risks through its tasks. Furthermore, is possible to represent method fragments with agile, planned and hybrid use-context through factors of the Octopus Model.

Figure 2 presents an example of method fragment proposed for the OSRiMA. The "TailoringGuide" attribute is about the fragment use-context (Octopus Model factors) and the risk(s) which the fragment can prevent. The higher similarity between the fragment use-context and project context, the better is the fragment.

Method-Fragment	*Fragment name.*
Purpose	*Here should be the fragment purpose, the fragment goal.*
Source	*Fragment source.*
Discipline	*One discipline related to software development projects.*
Phase(s)	*One or more development phases practiced in the organization.*
Tasks(s)	*Task(s) associated with the fragment.*
Worker(s)	*Roles involved in the fragment execution.*
TaskTechnique Mapping	*Techniques and guides to execute the fragment tasks.*
Action(s)	*Relations with the product-fragment (process artifacts). Can be "readOnly", "update", "create" or "delete".*
TailoringGuide	***Size:** value for "size".* ***Stable Architecture:** value for "stable architecture".* ***Business Model:** value for "business model".* ***Team Distribution:** value for "team distribution".* ***Rate of Change:** value for "rate of change".* ***Age of System:** value for "age of system".* ***Criticity:** value for "criticity".* ***Governance:** value for "governance".* --- *Here should be one or more risks associated with the fragment. In others words, risks which the fragment can prevent.*

Fig. 2. Model of method fragments.

5 Illustrating the OSRiMA

The Figure 3 shows examples of fragments associated with project risks. It presents the fragments which would be selected to prevent the risks "Lack of User Involve-

ment" and "Non-realistic Schedule and Budget". Following the proposed approach, the fragments are prioritized according to the project context after the selection according to project risks.

These fragments and associations are suggestions and the organization can define fragments and its associations according to needs and experiences.

PROJECT RISK	METHOD FRAGMENT
Lack of User Involvement	Engage Customer [6]
	On Site Customer [7]
	Planning Game [7]
	Sprint Planning Meeting [19]
	Workshop on Customer Involvement [6]
Non-realistic Schedule and Budget	Iteration Planning [7]
	Planning Game [7]
	Size the Schedule [6]
	Work Queue [6]

Fig. 3. Association between risks and method fragments.

6 Related Works

Kornyshova et al. [13] propose the inclusion of multi-criteria selection in SME approaches and analyzes some techniques to do it. This work do not defines a fixed set of situational criteria, thus the situational factors selection depends of only on the engineer experience and can changes according each project or person. Furthermore do not apply the SME approach for a specific purpose like we did for SRM.

Gericke et al. [7] proposes a situational method for governance, risk prevention and compliance information systems with focus on method fragments. It does not define criteria to contextualize method fragments in relation to the project context and do not provides guides about the analysis and selection of the best fragments in relation to the project situational characteristics.

Fragments for SME are proposed in Abad [1]. These elements are limited to the agile context. Furthermore the authors do not present an application example using your fragments nor an approach or guides to select and prioritize them.

7 Conclusions and Future Works

Literature and experience shows that development processes/methods must be configurable depending on the situational at hand. We propose a systematic approach for risk prevention through process tailoring based on SME concepts call Octopus SME - Risk Management Approach (OSRiMA).

OSRiMA considers the project context to suggest the best (most appropriate) method fragments. The fragments are retrieved from method base according to the project risks and are prioritized by algorithm based on the AHP technique according to project context.

We focus on method fragments (definition, selection and prioritization), rules for the composition of method fragments into situational method with the organization's

standard software process is subject to further works. Furthermore we will work to improve the method base and the support tool to select and prioritize fragments.

References

1. Abad, Z. S. H., Sadi, M. H., Ramsin, R.: Towards Tool Support for Situational Engineering of Agile Methodologies. In: Proceedings of the 2010 Asia Pacific Software Engineering Conference (APSEC '10), pp. 326-335, Washington (2010).
2. Aharoni A. and Reinhartz-Berger, I.: A Domain Engineering Approach for Situational Method Engineering. In: Proceedings of the 27th International Conference on Conceptual Modeling (ER '08), pp. 455-468, Berlin (2008).
3. Alegría, J. A. H., et al.: An MDE approach to software process tailoring. In: Proceedings of the 2011 International Conference on Software and Systems Process, pp. (2001) 43-52,New York(2001).
4. Beck, K: Programação Extrema (XP) Explicada: Acolha as Mudanças. Bookman, Porto Alegre (2004).
5. Coplien, J. O.: Software Patterns. SIGS Books and Multimedia(1996).
6. Cunningham, W. Portland Pattern Repository. (2004). <http:// http://c2.com/ppr/>.
7. Gericke, A. et al.: Situational Method Engineering for Governance, Risk and Compliance Information Systems. In: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology (DESRIST '09). ACM, New York (2009).
8. Ginsberg, M. P. and Quinn, L. H.: Process Tailoring and the Software Capability Maturity Model. Software Engineering Institute. Carnegie Melon University, Pennsylvania (1995).
9. Hall, E. M.: Managing Risks: Methods for Software Systems Development. Addison Wesley (2003).
10. Henderson-Sellers, B. et al.: Comparison of Method Chunks and Method Fragments for Situational Method Engineering. In: Proceedings 19th Australian Software Engineering Conference. (ASWEC2008), pp. 479-488, Los Alamitos (2008).
11. Henderson-Sellers, B. and Ralyté, J.: Situational Method Engineering: State-of-the-Art Review. Journal of Universal Computer Science, vol. 16(3), pp. 424-478(2010).
12. International Organization for Standardization: ISO 24744:2007 Software Engineering - Metamodel for Development Methodologies (2007).
13. Kornyshova, E. et al.: Method Chunks Selection by Multicriteria Techniques: an Extension of the Assembly-based Approach. In: Ralyté, J., Brinkkemper, S., Henderson-Sellers, B. (eds.) IFIP International Federation for Information Processing, vol. 244, Situational Method Engineering: Fundamentals and Experiences, Boston Springer, pp. 64-78 (2007).
14. Kruchten, P.: Contextualizing Agile Software Development. In: Proceedings of the EuroSPI 2010 Conference, pp. 1-12, Grenoble (2010).
15. Rational Software Corporation: Rational Unified Process. Cupertino, USA (2003).
16. Saaty, T. L.: The Analytic Hierarchy Process. McGraw-Hill International, New York (1980).
17. Schwaber, K., Beedle, M. Agile Software Development with Scrum. Upper Saddle River: Prentice Hall (2001).
18. Xu, P. and Ramesh, B.: Using Process Tailoring to Manage Software Development Challenges. IT Professional, vol. 10, pp. 39-45 (2008).