Demonstration of the Online Method Engine

Kevin Vlaanderen, Sandor Spruit, Fabiano Dalpiaz, and Sjaak Brinkkemper

Utrecht University, Department of Information and Computing Sciences, P.O. Box 80.089, 3508 TB, Utrecht, the Netherlands, {k.vlaanderen,a.g.l.spruit,f.dalpiaz,s.brinkkemper}@uu.nl

Abstract. During the past decade, much research has been performed in the areas of method engineering and process improvement. As a result of this research, we are developing the online method engine (OME). The OME is a knowledge management system that provides support during process improvement initiatives, using a set of assessments, an extensive method base, and automatic method assembly mechanisms. The system is designed around four main acivities: knowledge dissemination, method assessment, method improvement, and method enactment. In this paper, we demonstrate the core components and the main scenarios for the usage of the online method engine.

Keywords: method engineering; software tools; online method engine

1 Introduction

In an attempt to describe and reuse software development practices, several researchers have proposed method bases to capture this knowledge [5, 11]. The method fragments in these method bases are being configured and re-engineered using several techniques [8, 12], and implemented through a variety of tools [6]. Some of these techniques are in turn applied to the method engineering activity itself [4].

The online method engine (OME) is the implementation of scientific concepts in the field of method engineering (ME) and software process improvement (SPI), in combination with concepts from the field of knowledge kanagement (KM)). The main goal of the OME is to provide a system that facilitates the sharing of knowledge related to method fragments for software development, assessing methods that are being used in practice, improving methods based on assessment results, and adopting (improved) methods in practice. The main philosophy behind the OME is that incremental (or stepwise) improvement of processes reduces risks related to process change and improves the change of success [10, 3, 1].

Following the four goals of the OME, the system is based on four layers of functionality; knowledge dissemination, method assessment, method improvement, and method enactment. Each layer increasingly relies on the previous ones, and all of the layers rely on a central method base, which forms the backbone of the system. The quality of the system is ensured by a constant process of validation and review.

2 Architecture

Traditionally, method bases contain fragments of method knowledge, which reside on the M2 level (according to the MOF framework) [9]. In the context of the OME, the method base contains some additional elements. In the first place, it contains a set of situational indicators that can be used to describe an organizational unit. These situational indicators describe the specifics of the organizational context in terms that are unrelated to the specific method implementation. The situational indicators can be used for other purposes, but this will be described later on.

Complimentary to the situational indicators, the method base contains a set of capabilities that can be used to characterize a specific method implementation in a specific domain. The capabilities relate directly to a domain model, which captures the relevant process areas in a specific domain.

The situational indicators and the capabilities are related through a domain-dependent mechanism called the situational factor effect. A situational factor effect describes how a specific value of a situational indicator typically influences the relevance of a capability [2]. These situational factor effects are also captured in the method base.

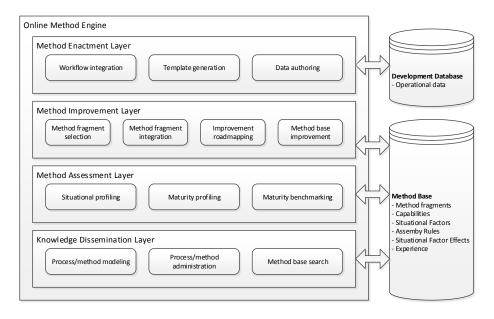


Fig. 1. Functional Architecture of the OME [17]

The final core element of the method base consists of a set of method fragments. Method fragments describe a coherent piece of method knowledge that facilitates reaching a specific goal during software development. A method fragment is described in terms of activities and deliverables. Within the OME, method fragments are currently

170

Demonstration of the Online Method Engine

modeled using process deliverable diagrams (PDDs) [18], which are a combination of a UML activity diagram and a UML class diagram, connected through a set of links.

3 Scenarios

3.1 Scenario A: Method Assessment

Maturity	Α	в	С	D	E	F
Requirements management						
Requirements gathering	Implemented	Implemented	Missing	Missing	Missing	Irrelevant
Requirements identification	Implemented	Implemented	Missing	Irrelevant		
Requirements organizing	Implemented	Missing	Missing			
Release planning						
Requirements prioritization	Implemented	Implemented	Irrelevant	Implemented	Missing	
Release definition	Implemented	Missing	Missing	Irrelevant	Irrelevant	
Release definition validation	Missing	Missing	Irrelevant			
Scope change management	Implemented	Missing	Irrelevant	Irrelevant		
Build validation	Implemented	Implemented	Missing			
Launch preparation	Implemented	Implemented	implemented	Irrelevant	Missing	Missing
Product planning						
Road map intelligence	Implemented	Implemented	Implemented	Missing	Missing	
Core asset roadmapping	Implemented	Implemented	Missing	Irrelevant		
Product roadmapping	Implemented	Implemented	Missing	Irrelevant	Missing	
Portfolio management						
Market analysis	Implemented	Implemented	implemented	Implemented	Missing	
Partnering & contracting	Implemented	Implemented	Implemented	Missing	Irrelevant	
Product life cycle management	Implemented	Missing	Missing	Irrelevant	Irrelevant	

Fig. 2. Example Areas of Improvement Report

In general, a process improvement effort starts with an assessment of the current processes. The assessment approach employed within the OME is based on the situational assessment method (SAM) [2]. This method consists of three phases; data collection, calculcation, and feedback. Data collection is performed through two questionnaires; one to determine the organizational context, and one to determine the current capabilities of the organization. During the calculcation phase, the former results are transformed into a current capability profile (CCP) and the latter into an optimal capability profile (OCP). The delta between these two profiles results in an areas of improvement matrix (AIM). During the feedback phase, an evaluation is performed that is used to improve the quality of the knowledge base (i.e. the method base).

The SAM is realized through two forms within the OME. The first form is used to capture the organizational context. It consist of 24 questions spread out over 5 pages. Each question has a short description, a set of answers, and possibly some help text to

indicate the type of answer expected. The second form is used to capture the current capabilities. This form contains 68 questions, which are spread out over 4 pages. Each page represents a business function, i.e. a layer from the SPM competence model [19]. The results of the questionnaires can be reviewed using two seperate reports; a situational profile report and a capability maturity report. The combined results are shown through an areas of improvement report, which shows both a condensed as well as an expanded version of the AIM. An example of the condensed *areas of improvement* report is shown in Figure 2.

3.2 Scenario B: Method Discovery

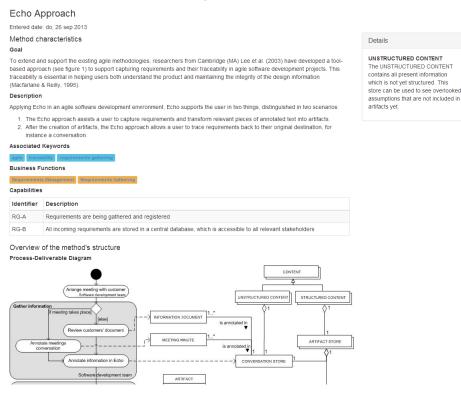


Fig. 3. Example Method Fragment Description

The results of the method assessment activity indicate the areas within the current process that are open for improvement. This is done in the form of a set of capabilities that are divided over various process areas, such as requirements prioritization or product roadmapping, and ordered based on an associated maturity level. This makes it possible to match these missing capabilities, or a subset of them, to existing methods that are stored in the method base.

The OME facilitates method discovery by providing tools to search the method base based on several filters. These filters include the focus area, the business function,

the relevant capabilities, and keywords. The result is a set of method fragments that fit these filters. Each result can be expanded into a detailed description of the method fragment. The main components of these descriptions are a textual summary of the method fragment, a detailed description that can contain both text and illustrations, a list of the relevant situational factors and capabilities, a PDD, descriptions of the main activities and deliverables, and a list of reference. An example excerpt of a method description for the Echo approach [7] is shown in Figure 3.

Within the OME, the user is aided in this discovery process as much as possible. Elements within the PDD and items within the situational factor and capability lists are hyperlinks to more detailed descriptions. This makes it possible to obtain a quick overview, while allowing the user to go into more depth or to find related method fragments.

3.3 Scenario C: Method Improvement

An important goal of the OME is to assist during a process improvement effort. In many cases, there are various solutions available to solve a specific problem, such as requirements prioritization. Not all of these solutions are applicable to any given situation, and it is hard for the method engineer to determine which solutions are likely optimal. The knowledge available within the method base allows for an automated approach to selecting appropriate method fragments. This selection can be based on the structural aspects of the method fragment (activities, deliverables), required capabilities, and situational factors.

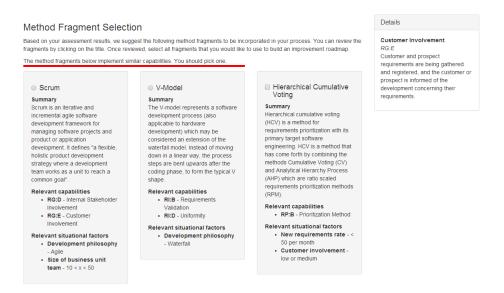


Fig. 4. Method Fragment Selection

The OME can propose a set of relevant method fragments based on the requirements of the user. It is possible to select method fragments that focus on one or more specific business functions, that implement capabilities up to a certain maturity level, or for the entire process. The selected fragments are presented to the user so that they can be reviewed using the tools described above (under method discovery). A prototype of the selection step is shown in Figure 4.

Once the user has selected the method fragments that he deems relevant, these fragments need to be assembled. In most cases, this means that they need to be integrated within the existing process. This activity can be partially performed automatically based on the structural aspects of the method fragments. In many cases, conflicts will arise during this assembly. For instance, multiple method fragments can include similar or incompatible deliverables. A report of these issues is presented to the user for further review.

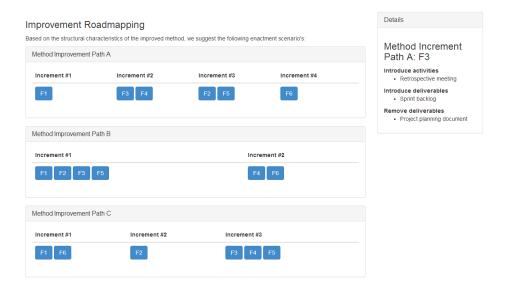


Fig. 5. Improvement Roadmapping

The overarching goal of the OME is to allow for incremental improvement. This is a very important characteristic, as the implementation of many process changes at the same time is often unrealistic and unfeasible within an organization. Process changes are always prone to resistance among employees, unforeseen complications, and changes within the environment. Therefor, the system generates a series of implementation plans [14]. These plans consist of a series of increments. Each increment consists of a set of small changes, such as inclusion or removal of activities and deliverables. Plans are generated based on a set of parameters, including the available resources and temporal constraints such as the need for a certain capability to be implemented within a certain amount of increments [14].

The generated plans are presented to the user as a set of timelines. Each timeline contains the proposed increments with a summary of their contents. Once again, the user can review these changes to gain a detailed understanding of their contents and impact (see FIgure 5 for an example of these timelines).

3.4 Scenario D: Method Enactment

The final activity within a process improvement effort, apart from the review, is the enactment of process changes. Although this is an activity that is mainly dependend on social and managerial aspects, it is possible to support parts of it through automated means. Within the context of the OME, enactment support focuses on the generation of templates and the automated migration of development tools.

3.5 Scenario E: Method Administration

All functionality within the OME is based on the contents of the method base. This method base consists of method fragments, capabilities, situational factors, situational factor effects, and experience reports. For the creation of method fragments, we employ specialized tools instead of developing functionality within the OME itself. MetaEdit+[13] is used to model the process-deliverable diagrams, which can be annotated with relevant capabilities. Textual descriptions can be created with appropriate textual editors. References are stored in the BibTex format.

4 Discussion and Future Research

Both the conceptual design as well as the initial prototype of the OME have been validated in earlier studies [16, 15]. In its current iteration, the OME does not fully support all of the described scenarios. Development follows the layers as described in Figure 1 from bottom to top. The method base has been adequately implemented, as is functionality related to method discovery. The method assessment layer has also been realized, making it possible to assess current methods and link the results to method fragments proposal.

On the method improvement layer, we have realized partial planning functionality. This makes it possible to generate a set of improvement plans based on a goal method. However, more research is needed to incorporate the removal and replacement of method fragments, and to support more complex situations including improvement based on an existing method.

For the method enactment layer, no functionality has been implemented sofar. Techniques are currently under development to translate method changes into concrete enactment actions.

References

1. Baddoo, N.: De-motivators for software process improvement: an analysis of practitioners' views. Journal of Systems and Software 66, 23–33 (Apr 2003)

- Bekkers, W., Spruit, M., van de Weerd, I., van Vliet, R., Mahieu, A.: A situational assessment method for software product management. In: Alexander, T., Turpin, M., van Deventer, J. (eds.) Proceedings of the European Conference on Information Systems. pp. 22–34. Pretoria, South-Africa (2010)
- 3. Diaz, M., Sligo, J.: How software process improvement helped Motorola. Software, IEEE 14(5), 75–81 (1997)
- Hug, C., Front, A., Rieu, D.: A Process Engineering Method Based on Ontology and Patterns. In: Proceedings of the International Conference on Software and Data Technologies (2008)
- Jarke, M., Gallersdörfer, R., Jeusfeld, M.A., Staudt, M., Eherer, S.: ConceptBase A deductive object base for meta data management. Journal of Intelligent Information Systems 4(2), 167–192 (Mar 1995)
- Karlsson, F., Å gerfalk, P.J.: MC Sandbox: Devising a tool for method-user-centered method configuration. Information and Software Technology 54(5), 501–516 (May 2012)
- Lee, C., Guadagno, L., Jia, X.: An agile approach to capturing requirements and traceability. In: International Workshop on Traceability in Emerging Forms of Software Engineering (2003)
- Nehan, Y.R., Deneckère, R.: Component-based Situational Methods: A framework for understanding SME. In: Situational Method Engineering: Fundamentals and Experiences. vol. 244, p. 161. Springer (2007)
- Object Management Group: Meta Object Facility (MOFTM). Tech. rep., Object Management Group (2011)
- Pino, F.J., Pedreira, O., García, F., Luaces, M.R., Piattini, M.: Using Scrum to guide the execution of software process improvement in small organizations. Journal of Systems and Software 83(10), 1662–1677 (Oct 2010)
- 11. Ralyté, J.: Reusing scenario based approaches in requirement engineering methods: CREWS method base. In: International Workshop on Database and Expert Systems Applications (DEXA). pp. 305–309. No. Dexa 1999, Ieee (1999)
- Ralyté, J., Rolland, C., Plihon, V.: Method enhancement with scenario based techniques. In: Proceedings of CAiSE. pp. 103–118 (1999)
- Smolander, K., Lyytinen, K.: MetaEdit—a flexible graphical environment for methodology modelling. In: International Conference on Advanced Information Systems Engineering. pp. 168–193 (1991)
- Vlaanderen, K., Dalpiaz, F., Brinkkemper, S.: Finding Optimal Plans for Incremental Method Engineering. Manuscript accepted for publication. In: International Conference on Advanced Information Systems Engineering (2014)
- Vlaanderen, K., Dalpiaz, F., van Tuijl, G., Spruit, S., Brinkkemper, S.: Online Method Engine: a Toolset for Method Assessment, Improvement, and Enactment. Manuscript accepted for publication. International Journal of Information System Modeling and Design (IJISMD) (2014)
- Vlaanderen, K., van de Weerd, I., Brinkkemper, S.: On the Design of a Knowledge Management System for Incremental Process Improvement for Software Product Management. International Journal of Information System Modeling and Design (IJISMD) 3(4), 21 (2012)
- Vlaanderen, K., van de Weerd, I., Brinkkemper, S.: Improving software product management: a knowledge management approach. International Journal of Business Information Systems (IJBIS) 12(1), 3–22 (2013)
- van de Weerd, I., Brinkkemper, S.: Meta-modeling for situational analysis and design methods. In: Handbook of Research on Modern Systems Analysis and Design Technologies and Applications, chap. III, pp. 35–54. Information Science Publishing (2008)
- van de Weerd, I., Brinkkemper, S., Nieuwenhuis, R., Versendaal, J., Bijlsma, L.: Towards a Reference Framework for Software Product Management. 14th IEEE International Requirements Engineering Conference (RE'06) pp. 319–322 (Sep 2006)

176