Information and Software Technology 78 (2016) 95–120

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

A pattern-based model-driven approach for situational method engineering

Halimeh Agh, Raman Ramsin*

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 20 December 2015
Revised 11 May 2016
Accepted 31 May 2016
Available online 2 June 2016

Keywords:
Software development methodology
Situational method engineering
Model-driven development
Process modeling
Pattern-based model transformation

ABSTRACT

Context: Constructing bespoke software development methodologies for specific project situations has become a crucial need, giving rise to Situational Method Engineering (SME). Compared with Software Engineering, SME has a long way to go yet; SME approaches are especially deficient as to support for modeling, portability, and automation. Model-Driven Development (MDD) has been effectively used for addressing these issues in Software Engineering, and is also considered a promising approach for resolving them in SME.

Objective: This paper aims to address the shortcomings of existing SME approaches by introducing a novel MDD approach, specifically intended for SME purposes, that uses a pattern-based approach for model transformation.

Method: Developing a MDD approach for SME requires that a modeling framework, consisting of modeling levels, be defined for modeling software development methodologies. Transformation patterns should also be specified for converting the models from one level to the next. A process should then be defined for applying the framework and transformations patterns to real SME projects. The resulting MDD approach requires proper evaluation to demonstrate its applicability.

Results: A framework and a semi-automated process have been proposed that adapt pattern-based model transformation techniques for application to the methodology models used in SME. The transformation patterns have been implemented in the Medini-QVT model transformation tool, along with two supplementary method bases: one for mapping the situational factors of SME projects to requirements, and the other for mapping the requirements to method fragments. The method engineer can produce the methodology models by using the method bases and executing the transformation patterns via the tool.

Conclusion: The validity of the proposed approach has been assessed based on special evaluation criteria, and also through application to a real-world project. Evaluation results indicate that the proposed approach addresses the deficiencies of existing approaches, and satisfies the practicality requirements of SME approaches.

1. Introduction

The challenges of modern software development have necessitated that software development methodologies be custom-built or adapted; this has led to the emergence of a discipline called Method Engineering (ME), formally defined as: "The engineering discipline to design, construct and adapt methods, techniques and tools for the development of systems"; Situational Method Engineering (SME) is an important subfield of method engineering that focuses on composing or adapting a software development methodology according to the specific characteristics of the project at hand (expressed in terms of situational factors) [1]. Various high-level approaches have been proposed for SME, the most prominent of which are: Paradigm-based [2], Assembly-based [3], and Extension-based [4]. The paradigm-based approach is rooted in metamodeling, in that the target methodology is produced by instantiating, abstracting or adapting an existing metamodel; in the assembly-based approach, the target methodology is constructed by reusing methodology parts to compose a new methodology or to enhance an existing one; and in the extension-based approach, an existing methodology is augmented by applying special-purpose extension patterns. SME has been evolving for two decades; however, existing SME approaches have not been able to fully address the various requirements of the SME domain, the most important of which include: Complexity management, automation, portability, and accuracy [5]. In Model-Driven Development (MDD),

* Corresponding author. Department of Computer Engineering, Sharif University of Technology, Azadi Ave., Tehran, Iran. Fax: +98 21 6601 9246.
E-mail addresses: agh@ce.sharif.edu (H. Agh), ramsin@sharif.edu (R. Ramsin).

http://dx.doi.org/10.1016/j.infsof.2016.05.010
0950-5849/

models are the pivotal concepts and artifacts, and automating the production of models through model transformations is the primary objective [6]. This mindset has many advantages, such as increasing productivity, managing complexity, and enhancing portability. As the major manifestation of MDD, the Model-Driven Architecture (MDA) prescribes the production of three models of the target software system: Computation-Independent Model (CIM), Platform-Independent Model (PIM), and Platform-Specific Model (PSM) [7]; this multi-level modeling framework, which relies on the notions of "Computation" and "Platform", has become a de-facto standard for MDD approaches. Model transformation is an important issue in MDD, as lower-level models are produced by transforming their higher-level counterparts. There are several approaches for model transformation in MDD, including graph-based, metamodel-based, by-example, and pattern-based [8].

Due to its potential benefits, MDD has already been used for SME purposes [5]; however, as discussed in Section 5.2, existing approaches are deficient in that many of them lack adequate support for automation of model transformations, and all of them are over-dependent on the method engineer's expertise for determining the method fragments suitable for constituting the target methodology. Moreover, these methods need to be further improved as to the quality attributes typically targeted in MDD: complexity management, portability, productivity, and reusability. These attributes, as applicable in the context of SME, are briefly explained below:

- Complexity management: many SME tasks are inherently complex; prominent examples include: identifying the requirements of the target methodology, specifying method fragments for each requirement, determining the relationships among the method fragments, and modeling the constructed methodology. SME approaches should provide adequate means for managing the complexity of SME tasks and the artifacts produced.
- Portability: capability to transition the produced methodology onto different platforms (enactment environments).
- Productivity: SME is typically a prelude to the main software development activities; therefore, it is important that high-quality situational methodologies be developed and maintained rapidly and cost-effectively.
- Reusability: capability to reuse software process knowledge; this can be realized through eliciting method fragments from existing methodologies and storing them in method bases which act as knowledge repositories.

We propose a novel pattern-based MDD approach for SME with the specific aim of addressing the above issues. The approach consists of: 1) a multi-level, pattern-based MDD framework for methodology modeling and transformation (which we have chosen to call PBMDD4SME, short for "Pattern-Based Model-Driven Development for Situational Method Engineering"), and 2) a process for applying the framework to the engineering of software development methodologies based on the specifications of the project situation at hand. The framework defines three levels of models and a set of pattern-based transformation rules. The transformation rules have been implemented in the Medini-QVT model transformation tool [9], and have been augmented with two supplementary method bases; one for mapping the situational factors (which express the project situation) to methodology requirements and the other for mapping the requirements to method fragments. This enables the method engineer to produce the models and compose the target methodology in a semi-automated manner through following a model-driven, tool-supported process.

We have used the pattern-based transformation approach [8] in our proposed SME framework, as it facilitates automation and can be suitably adapted to the SME context. A *pattern* is a successful *solution* to a recurring *problem* in a given *context*. In the pattern-based model transformation approach, each transformation pattern defines a general problem of interest in the source model and its corresponding solution (transformed counterpart) in the target model [10]. Several such pattern sets already exist, but they are used for model transformation in a software development context, not for SME (examples are provided in Section 2). Our proposed transformation patterns are specifically intended for use in the SME context, and have been described by using the general template suggested in [10]; the template consists of: 1) the pattern's name, purpose, motivation, and applicability, 2) the problem sets resolved by the pattern, 3) features of the solution, and 4) an illustration of how the pattern is typically applied.

In order to evaluate our proposed approach, we have compared it with other MDD frameworks and other model-driven SME approaches based on specially defined criteria. In addition, PBMDD4SME and its process have been applied to a real-world SME project, and the results have demonstrated its merits in a practical context. Evaluation results show that the proposed approach addresses the deficiencies encountered in existing SME approaches, improves on the quality attributes, and is applicable to real-world project situations.

The rest of this paper is structured as follows: Section 2 provides a survey of the related research; Section 3 introduces the proposed PBMDD4SME framework and presents the two method bases that are used in the proposed approach; Section 4 presents a process for applying the proposed framework to SME problems; Section 5 provides the results of evaluating the proposed approach; and Section 6 presents the concluding remarks and suggests ways for furthering this research.

2. Related research

Many of the concrete methods and frameworks currently used in SME, such as the OPEN methodological approach [11,12], the ISO/IEC-24,744 framework [13,14], and the MMC approach [15], are not model-driven; whereas MDD can be of significant benefit to these approaches, mainly due to its positive effect on complexity management and automation. The literature related to this research includes works on the application of MDD for SME purposes, as well as MDD works focusing on pattern-based transformation.

In general, there are two categories of SME works related to this research:

1. SME approaches that use MDD explicitly: In [16], a methodological MDD framework is proposed for SME that is applied in three phases: Method design, method configuration, and method implementation. In [5], a MDD framework by the name of Model Driven Situational Method Engineering (MDSME) is proposed, which defines four modeling levels: Enactment-Independent Model (EIM), Paradigm-Independent Model (ParIM), Paradigm-Specific Model (ParSM), and Platform-Specific Model (PSM). In [17], methodology engineering methods based on MDD have been surveyed. In [18], MDD has been applied to method tailoring for specific project situations.
2. SME approaches that are not model-driven by name, but highly resemble classical MDD approaches: In [19], a three-level scheme has been used for describing the target methodology, which resembles the multi-level modeling scheme typically used in MDD approaches; the levels include: General level, model level, and project level. In [20], a method has been introduced to develop a methodology based on specific requirements; the method is applied in three steps, which involve multi-level specifications: Method requirements engineering, method design, and method implementation.