APPLYING AND EVALUATING AN MDA PROCESS MODELING APPROACH

Rita Suzana P. Maciel, Bruno C. da Silva and Ana Patrícia F. Magalhães
Federal University of Bahia, Salvador, Bahia, Brazil

Keywords: MDA, Process Specification, SPEM.

Abstract: In order to use the MDA approach, several software processes have been defined over recent years. However, there is a need for specifying and maintaining MDA software process definitions systematically which can also support process enactment, reutilization, evolution, management and standardization. Some empirical investigations have been performed concerning the usage of several MDA-related approaches. In this paper we describe our technique for MDA software process specification and enactment, including tool support. We also present case studies and the concluding results on the application of our approach for process modeling.

1 INTRODUCTION

Model Driven Engineering (MDE) is an approach specially focused on modeling techniques. MDE advocates that the models of a software system are not only used for documentation, but they actually serve as basis for the implementation phase. One of the most well-known initiatives in this scenario is *Model-Driven Architecture* (MDA) (OMG, 2003).

The use of MDA requires the definition of a software process that guides developers in the elaboration and generation of software models (Mellor, 2004). Such a kind of software process, which we refer to hereafter as *MDA Process*, evokes definitions such as modeling phases, metamodels, UML profiles and transformation rules, besides traditional process elements such as iterations, tasks, workproducts, roles and so on. Therefore, the specification of an MDA process is not a trivial task. In order to use the MDA approach several software processes have been defined over recent years (Koch, 2006; Maciel, 2006; OpenUP/MDD, 2008). Most of them adopt *ad hoc* notations and different concepts are used to define the activities and artifacts for the software development life cycle.

In this context, our experience was on the specification of an MDA process for the development of domain-specific middleware services (Maciel, 2006). When we started to specify such an MDA process we became aware of the difficulty in specifying the process with open standards such as EDOC (*Enterprise Distributed Object Computing*) (OMG, 2002) and RM-ODP (*Reference Model of Open Distributed Processing)* (ISO, 2004), and further searched for a transformation engine well-suited to our needs. Available MDA supporting tools usually have low flexibility in the sense that the MDA process has to adapt to the tool in order to have automated support.

Moreover, current MDA supporting tools are particularly interested just in defining and executing transformations which produce code and deployment artifacts from models (eg, AndroMDA, oAW). Indeed, other activities in a software process are usually not considered, such as definition of metamodels and UML profiles. Emerging from this experience and scenario we defined the *SPEM/MDA approach* (Maciel, 2009a; Maciel, 2009b) based on MDA and SPEM standards. Our approach extends the SPEM metamodel focusing on specific definitions of the MDA in order to make the instantiation of MDA concepts explicit as process elements. We also provide an integrated supporting tool for MDA process modeling and enactment, called *Transforms*.

There is a specific process called OpenUP/MDD (OpenUP, 2008) built on Eclipse Process Framework (EPF) which focuses specially on the model driven development. However, the OpenUP/MDD is a process instance of the SPEM metamodel concepts, while our approach is placed at

Suzana P. Maciel R., C. da Silva B. and Patrícia F. Magalhães A. (2010).
APPLYING AND EVALUATING AN MDA PROCESS MODELING APPROACH.
In *Proceedings of the 12th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 185-190
DOI: 10.5220/0002889401850190

a higher abstraction level. Consequently, we provide a more flexible and extensible way to model and specify (instantiate) model driven software processes according to SPEM 2 and MDA standards.

Additionally, Bendraou *et al.* (2007) proposed an extension to the SPEM 2.0 specification, called xSPEM, in order to allow process enactment. Although their work targets the enactment of process models they don't focus particularly on the process modelling or enactment of MDA development process.

Our main goal in this paper is to describe the application of our SPEM/MDA approach and also to report our resulting conclusions. Initially, we performed preliminary experiments to assess our approach. However, new empirical investigation became necessary and thus we have worked in different scenarios to evaluate our approach and tool.

The rest of the paper is organized as follows: Section 2 briefly presents our current approach and tool for MDA process specification and enactment; in Section 3 we describe the last case study we have performed detailing the results; Section 4 concludes the paper with final remarks.

2 SPEM/MDA APPROACH

Software process modeling using a unified and consistent terminology should make communication, understanding, reutilization, evolution, management and standardization of the process possible (Humprey, 1989). In order to achieve this, our approach uses SPEM as PML (Process Modeling Language). Figure 1 shows our metamodel extending some SPEM 2.0 concepts for the MDA context.

The process specification is divided into two dimensions: the static concepts, which comprise d*isciplines*, *tasks*, *steps*, *roles* and *workproducts*, called *method content* in the first part of Figure 1; and the dynamic concepts, that comprise *phases*, *iterations* and *taskUse*, called Process in the second part of Figure 1.

A static definition represents best practices that can be reusable in many process models. A *discipline* groups a set of related *tasks* that are performed by *roles*. A *role* defines responsibilities and competences of an individual or a group of individuals. A *task* may comprise many *steps* to describe meaningful work. During the process execution it can consume/produce *workproducts* as input and output artefacts. In this approach a *workproduct* can be specialized into four kinds of artefacts: *UML model*, transformed / generated in process execution; *transformation rule,* that contains the rules for automatic model transformation and code generation; *extra model*, textual specifications or supplementary notation to document a project; and *profile*, which gives the semantic for modelling a system.

Based on the static definitions many processes are modeled using the metamodel dynamic concepts. A process may comprise many *phases* specialized in CIM (Computational Independent Model), PIM (Platform Independent Model), PSM (Platform Specific Model) and Codification for the MDA context. Furthermore, an *ExtraPhase* can be specified representing an additional stage apart from modeling and codification. The modeling phases can be associated to *profiles* to give the semantic of a specific domain. Each *phase* can contain one or more iterations that specify the *taskUses* necessary to carry out a *task*.

To better define the static and dynamic concepts the approach specifies the use of three kinds of UML diagrams: class, which models all the concepts and is the base for the latter ones; use case, to associate roles to specific tasks; and activity, to model the process workflow behaviour in terms of process execution.

2.1 Supporting Environment

We have developed a supporting environment called *Transforms* taking into consideration the SPEM/MDA metamodel illustrated in Figure 1.

The Transforms tool has two modules with complementary features represented by the two main components in Figure 2 called *ProcessEditor* and *ProcessExecutor*. The first one is the module for specifying MDA processes by creating and/or editing *Method Content* and *Process* elements according to our metamodel. This module encompasses the visual modelling (*DiagramsEditor* component) of process definitions and also he work breakdown structure (*WorkBreakDown StructureEditors*).

The second module offers a supporting environment for the enactment of MDA processes previously defined by the Process Editor. All the processes are stored in the process repository and they are accessible through the *DataAccessInterface* provided by the *PersistenceController* component.

The subcomponent called *ProjectExplorer* is responsible for monitoring and executing a project following a process retrieved from the repository.

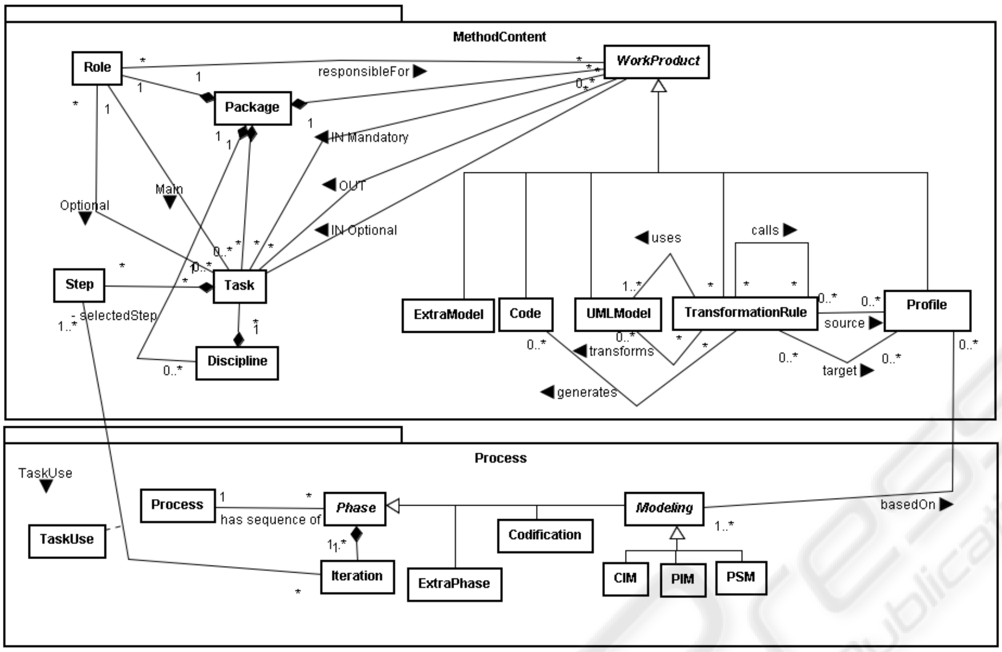

Figure 1: SPEM/MDA Metamodel.

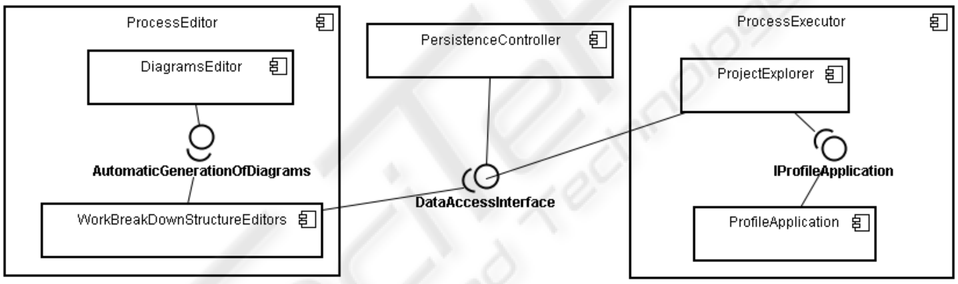

Figure 2: Transform Tool Architecture.

The subcomponent called *ProfileApplication* performs an automatic profile application whenever there is a UML profile associated with the corresponding modeling phase.

These modules were implemented as an Eclipse RCP (*Rich Client Platform*) product. Thus, it was possible to reuse the basic graphic structure as well as several available plug-ins, such as: ATL (*Atlas Transformation Language*), for elaboration and execution of model-to-model transformations; GMF (*Graphical Modeling Framework*), for creation and generation of graphic editors of our customised diagrams in conformance with our SPEM/MDA metamodel; and also the UML2Tools, for modeling the UML workproducts integrated with the process executor.

3 EVALUATING MDA PROCESS MODELING

This section presents the evaluation work we performed on a case study in order to verify the applicability of our approach and tool. The case study results are described in the following subsections.

As our initial experience we performed two case studies on the evaluation of our approach. Both of them focus on the technique of process specification based on the SPEM/MDA metamodel.

The first one (Maciel, 2009a) explored the specification of an MDA process for the development of specific middleware services previously defined in the literature (Maciel, 2006).

The second one (Maciel, 2009b) was the specification of an MDA process for web-based applications and it was performed together with an organization in the industry. The scenario and results of a new case study are presented in the following subsection.

3.1 Assessing the Authoring of MDA Process

Our previous case studies were performed in our laboratory and/or in conjunction with people from PRODEB company as described in (Maciel, 2009a; Maciel, 2009b). However, in the current case study we elaborated a different scenario as our goal was also to evaluate the applicability of the Transforms tool regarding the MDA process specification without our intervention.

The participants of this study were graduate students with job positions in the industry. Most of them were system analysts working for software factories and banks.

First of all, we gave them some lectures about MDA concepts, technologies and also training on our approach and tool totalizing 16 hours divided into 4 days in one week (4h/day). Then we organized them into 4 groups and gave each group the Transforms tool and asked them to specify an MDA process motivated from their organizational needs and reality.

The specification time was not restricted. They had all the time they needed to organize themselves in their groups to do the job and deliver the process. After the process delivery, we also applied a questionnaire and started to work analyzing the process specification.

The questionnaire which was answered individually by each participant was divided in two parts: the first one concerning the SPEM/MDA approach for process specification and modeling; and the second one regarding the *MDA Process Editor* from Transforms tool. The questionnaire had a total of 15 questions. The collected results and our analysis are presented in the following two subsections.

3.1.1 Results Concerning the SPEM/MDA Approach

The charts in Figures 3 and 4 show some results from the questionnaire. The first one (Figure 3) presents the answers related to questions about the simplicity of defining *tasks*, *roles*, *workproducts* and *phases*. While the second one (Figure 4) demonstrates that the participants were divided in suggesting or not the creation of new stereotypes for *phases* and *workproducts*.

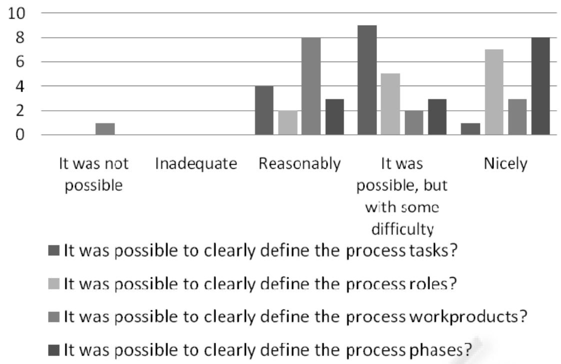

Figure 3: Answers about simplicity of defining tasks, roles, workproducts and phases.

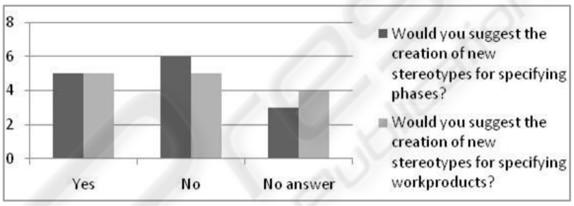

Figure 4: Suggesting or not new stereotypes.

Besides the aspects discussed from the results summarized in the above charts, it was also possible to observe that all the processes defined by the participants had well-defined modeling (CIM, PIM and PSM) and codification *phases*, including *tasks* and *steps, roles* assignment, associated *workproducts* and also transformation rules. Therefore, it is possible to conclude that our approach enabled process definition with expected characteristics of a traditional software process while also adding the peculiarities of an MDA process.

3.1.2 Results Concerning the using of the MDA Process Editor

The following figures show some results from questions regarding tool support in the MDA process specification.

Figure 5 indicates that most of the participants agree that the effort put into the process specification would not be repeated in the future when using the Transforms tool for specifying new process. That is, the time spent learning the approach and tool would not be repeated. Also, we can consider that the process definitions on the method content stay available for reuse. This can possibly reduce time and future effort in process specifications.

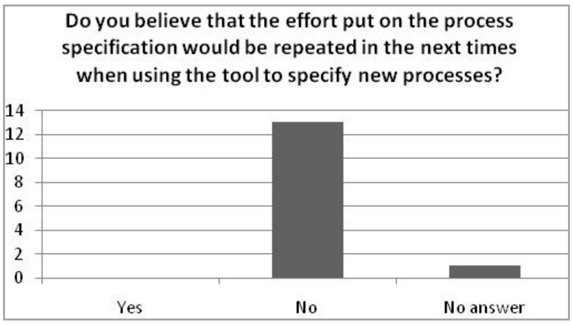

Figure 5: Repeated effort.

Some participants had difficulty understanding some process definitions as presented in the tool, however, none of them rated the process comprehension as presented in the tool negatively. The effort put into the process specification is valuable but necessary. Most of them classified such effort as reasonable but not enough to rate as a negative point. Besides, part of that effort would not be repeated on future occasions when using the tool as observed in the results summarized in Figure 6.

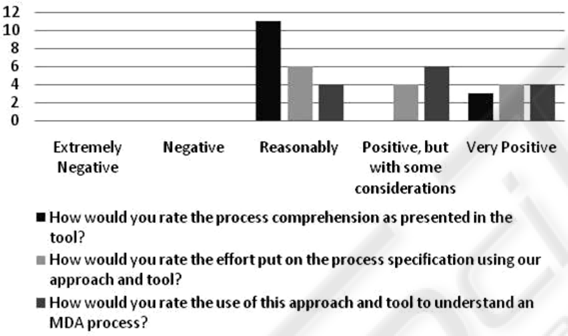

Figure 6: Process presentation, expended effort and process comprehension.

Another question was related to the preference regarding the kind of the process visual representation. The participants were equally divided between the diagram and work breakdown structure visualization. In general, we observed that people who have at least a basic notion of UML prefer the visual representation with diagrams, while the others prefer the hierarchical representation of the work breakdown structure.

During the experience of using the MDA Process Editor of the Transforms tool most of the participants used the on-line help which was released on the web. It can be explained by the fact that tool usage embeds several SPEM and MDA concepts. From this observation we can invest more time on improving the tool usability facilitating the use of the SPEM/MDA concepts of our metamodel.

Finally the last chart in this study is presented in Figure 7. It shows the time spent during the MDA process specification using the tool. Few of the users spent 4 hours on it, most of them spent more than 8.

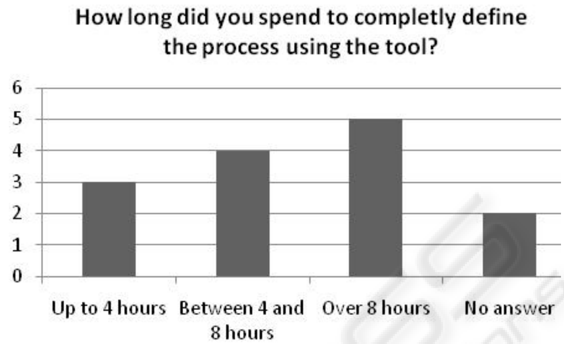

Figure 7: Spent time on process specification.

3.2 Lessons Learned and Study Constraints

Empirical assessment usually takes into account the amount of data collected from the subjects. However, in the case of an activity of process specification it is difficult to involve a high number of people in the experiments. There are few professionals in organizations involved with this kind of task. In general, many people enact on processes but few of them specify or model a software process. This observation has already been identified in our previous studies and it is also confirmed in our recent one. Accordingly, empirical assessment in this area facilitates more qualitative analysis than quantitative. Therefore, the assessment of the process enactment is easier to collect bigger amounts of data, facilitating a better quantitative analysis in contrast with assessing process specification.

Additionally, metrics to assess the process enactment are more mature in the literature and are also easier to apply in the software industry, such as metrics to evaluate productivity and cost. Nevertheless, metrics to assess the process modeling and specification (apart from its enactment) need to be better explored in the software engineering community.

We should also highlight that the conclusions obtained from our studies are restricted to the particular set of participants. In other words, our analysis regarding the advantages and drawbacks in using our approach and tool may not be directly generalized to other contexts. However, these studies have allowed us to make useful assessments about whether the specification of MDA processes with a supporting tool is worth studying further. As one of PML goals is software process automation, MDA may favor this.

In addition, the study have also allowed us to make a useful evaluation of the applicability of the SPEM/MDA approach and the Transforms tool concerning the specification of MDA processes, as detailed in Section 3.1.

4 CONCLUDING REMARKS

In general, the low level of flexibility of MDA supporting tools makes the adoption of current processes in organizations more difficult. Thus, this can be one of the factors that hinders a wider adoption of MDA in industry. A survey has been performed and led by Lancaster University into what affects the success of MDE in industry (EA-MDE, 2010). We believe that MDA adoption in industry depends on the integration among well-established enterprise software processes, MDA transformation engines and approaches.

This paper presented our initiative which is based on the need for process specification including MDA concepts using a well-known standard called SPEM. Our approach (Section 2) provides a means for MDA process specification and enactment using a supporting tool (Section 2.1). The case studies we performed (Section 3) produced interesting results regarding the applicability of our approach (Section 3.1). The SPEM/MDA metamodel and also our supporting environment are suitable for the empirical investigation that we have presented.

Future work includes (i) the adoption of an approach for model-driven testing using the Tranforms tool; (ii) investigation and building new features for project management; and (iii) exploring usability improvements of our tool. We also have ongoing work on the execution of a new case study concerning the MDA process enactment using our approach and tool.

ACKNOWLEDGEMENTS

This work is partially funded by Fapesb, project number 8694/2006, and grant number 0020/2006.

REFERENCES

Bendraou, R., Combemale, B., Cregut, X., and Gervais, M. 2007. Definition of an Executable SPEM 2.0. In *Proc.s of the 14th Asia-Pacific Software Engineering Conference*, APSEC. IEEE Computer Society, Washington, DC, 390-397.

ISO, 2004. Use of UML for ODP system specification. Working Draft. ISO/IEC JTC1/SC7.

Koch, N., 2006. Transformation techniques in the model-driven development process of UWE, In: *Workshop Proc. of the 6th intl Conference on Web Engineering*. ICWE '06, vol. 155. ACM, New York, 3.

EA-MDE, 2010. Empirical Assessment of the Efficacy of the MDE. Available at: http://www.comp.lancs.ac.uk/~eamde/

Humprey, W., Kelner,M. (1989). “Software Modeling: Principles of Entity Process Models”. SEI. Software Engineering Institute, Carnegie Mellon University. Pittsburgh, Pennsylvania, (CMU/SEI-89-TR-2).

Maciel, R., Silva, B., and Mascarenhas, L., 2006. An Edoc-based Approach for Specific Middleware Services Development, In: *4th Workshop on MBD of Computer Based Systems*, Postdam, Germany. Proc. IEEE Press, p:135–143.

Maciel, R., Silva, B., Magalhães, A. and Rosa, N., 2009a. An Approach for Model-Driven Development Process Specification, In: *Proc. of the 11th Intl. Conference on Enterprise Information Systems*, Milan, Italy.

Maciel, R., Silva, B., Magalhães, A. and Rosa, N., 2009b. An Integrated Approach for Model Driven Process Modeling and Enactment, In: *Proc. of the 23th Brazilian Symp. on Software Engineering*, Fortaleza, Brazil.

Mellor, S. *et al.*, 2004. *MDA Distilled*. EUA, Addisson-Wesley.

OMG, 2002. EDOC - UML Profile for Enterprise Distributed Object Computing Specification. OMG Adopted Specification (ptc/02-02-05).

OMG, 2003. *MDA Guide*. Version 1.0.1 (omg/2003-06-01).

OpenUP Component – MDD, 2008. Available at: http://www.eclipse.org/epf/openup_component/mdd.php.