Empirical Evaluation of UML-based Model-Driven Techniques

Maurizio Leotta, Filippo Ricca, Marco Torchiano, Gianna Reggio

Abstract:

In this poster, we sketch our research plan about a "massive" empirical evaluation of model-driven techniques following the first two already conducted steps in that respect (an exploratory survey and a series of controlled experiments concerning maintainability). We intend to experiment UML-based model-driven techniques in several contexts (e.g., desktop and Web applications), focusing on several software characteristics (e.g., maintainability and productivity) and employing empirical methods such as controlled experiments, surveys, case studies.

Digital Object Identifier (DOI):

`http://dx.doi.org/10.1109/RCIS.2013.6577735`

Empirical Evaluation of UML-based Model-Driven Techniques

Maurizio Leotta[1], Filippo Ricca[1], Marco Torchiano[2], Gianna Reggio[1]
[1]Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi (DIBRIS), Università di Genova, Italy
[2]Dipartimento di Automatica e Informatica (DAUIN), Politecnico di Torino, Italy
maurizio.leotta@unige.it, filippo.ricca@unige.it, marco.torchiano@polito.it, gianna.reggio@unige.it

***Abstract*—In this poster, we sketch our research plan about a "massive" empirical evaluation of model-driven techniques following the first two already conducted steps in that respect (an exploratory survey and a series of controlled experiments concerning maintainability). We intend to experiment UML-based model-driven techniques in several contexts (e.g., desktop and Web applications), focusing on several software characteristics (e.g., maintainability and productivity) and employing empirical methods such as controlled experiments, surveys, case studies.**

Keywords*—*Model-Driven Development; Research Plan; Industrial Survey; Controlled Experiment.

I. Introduction

Model-driven is a software development approach based on the concept of model. In practice, models play the role of primary artefacts of the development process. Although model-driven techniques are popular in academy, their introduction into industry seems to be relatively slow (this is one of the results obtained in our survey [6]). Probably, one of the main problems is the difficulty to convince managers of MD*[1] benefits. The main claimed benefits of MD* are improvements in productivity, portability, maintainability and interoperability [2]. However, such claims necessitate empirical evidence. Unfortunately, in the literature, there are few empirical works attempting to evaluate these aspects in a systematic way [3].

In this poster, we sketch our three-year research plan, consisting of a work in progress and planned tasks, having the ambitious goal to fill (at least partially) the "empirical gap" described above. Our intention is to continue in the direction we have undertaken, that is: conducting a "massive" empirical evaluation of UML-based model-driven techniques in several *contexts* (e.g., desktop, web and embedded applications), with several *tools*, focusing on several software *characteristics* (e.g., maintainability and productivity) and utilizing several empirical *methods* such as controlled experiments, surveys, case studies. So, we formulate a set of research questions, expressed in a unified parametric form: *evaluating the effect of MD* using the* <E> *empirical method, with respect to the* <C> *software characteristic, in the context of* <X> { *and with the* <T> *tool* } [2]. This parametric research question will be instantiated from time to time with the chosen parameters. Each instance will define a specific concrete research question that will constitute the basis of a new research path. Thus, our overall research plan is composed by a list of research paths.

On this topic, we have already conducted: (i) an informal literature review used for directing our research activities and helping us to take important decisions (e.g., choosing the more promising MD* tools for executing the experiments) and (ii) two initial steps of our plan:

– ***An on-line questionnaire-based industrial survey*** (completed) performed with 155 Italian software professionals, with the aim of investigating the usage of modelling in the context of software development and model-driven engineering [6]. The survey focused also on used modelling languages, processes and tools. Briefly, the results confirmed that model-driven engineering, and more in general software modelling, are very relevant phenomena. Approximately 68% of the sample use models during software development; among them, 44% generate code starting from models and 16% execute them directly. The preferred language for modelling is UML (76%), for this reason we start to focus our attention on UML.

– ***A series of controlled experiments*** (in progress) with bachelor [4], master (both at UniGE), and PhD (at PoliTO) students aimed at investigating the effect of model-driven development during software maintenance and evolution activities. In the experiments we used the UniMod tool, a specific implementation of executable UML, and two applications (Svetofor and Telepay). Results indicate a relevant shortening of time (see Figure 1) with no significant impact on correctness, gained through the use of UniMod instead of conventional programming (i.e., code-centric programming). However, the benefits of UniMod in terms of time reduction vary in a substantial way depending on the developers' experience (–33.3% for BSc students and –48.3% for PhD students).

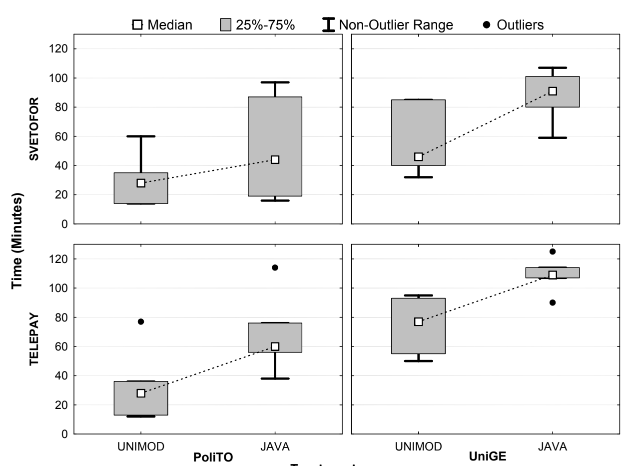

Fig. 1. Boxplots of Time

[1]We use the term MD* to indicate MDE/MDD/MDA

[2]between braces means optional

II. Research Plan

In the experiments mentioned above, we have considered only desktop applications. For generalization purposes, it is recommended to extend the experiments to other kind of applications such as Web or embedded applications. To this end, we have analysed several specific MD* tools and we have selected for the future experiments: WebRatio[3] (focused on production and maintenance of Web-based applications), BridgePoint[4] (focused on real-time embedded software applications) and AndroMDA[5] (focused on J2EE applications). Maintainability, the first characteristic considered in our experiments, is only one of the software characteristics possibly impacted by MD* [2]. In the next future, we will continue our work on maintainability and we will extend it considering also productivity. Our research plan on these two characteristics is summarized in Table I where works in progress are indicated with the check symbol (✓) while the tasks planned to start this year (2013) are marked with a green semaphore (🚦). Later, we intend to broaden our investigation to the other three characteristics reported in [2] (i.e., portability, interoperability and reusability).

Maintainability and Productivity: work in progress and planned activities. First, we have planned to carry out non-exact replications of our experiments on maintainability to obtain results more representative of the underlying reality [1]. In this context, we have already planned a new execution of our experiment on maintainability replacing UniMod (that is focused on desktop Java application) with BridgePoint (research path 1 in Table I) to verify whether the obtained findings hold also in the context of embedded C++ applications. In this case, we will employ more experienced subjects (i.e., professional developers) with respect to the previous experiments. In addition, to complete the picture, we have organized another exact replication of our experiment on maintainability (2) with professional developers. These experiments will allow us to understand: (1) whether the benefits of MD* vary changing the application type (and consequently the selected tool) and (2) whether the maintenance time reduction deriving from the use of MD* is preserved also for more skilled subjects.

Second, we have planned to employ other empirical methods, such as industrial case studies and action research[6] [5], to test on the field the two claimed benefits of MD*. To this end, we started a collaboration with a local company developing e-learning solutions. As first step, we planned to compute the difference in productivity (before) and maintainability (after) to respectively build and modify Web e-learning applications; in this case, the WebRatio tool will be used (3). Our idea is re-implementing from scratch, using WebRatio, a Web application already produced by the company with traditional techniques and comparing the production time (and software quality) with the real time spent by the company to produce that application[7]. Next, we plan to compare the time required to perform real maintenance tasks on both versions of the application (the original and the WebRatio-based).

[3] http://www.webratio.com/

[4] http://www.mentor.com/products/sm/model_development/

[5] http://www.andromda.org/

[6] in which the researchers attempt to solve a real-world problem while simultaneously studying the experience of solving the problem

[7] the company has kept the historical data of the original development

		Research Paths				
		1	2	3	4	5
Char.	Maintainability	X	X	X	-	-
Char.	Productivity	-	-	X	X	X
Software	Desktop applications	-	X	-	-	-
Software	Web applications	-	-	X	X	-
Software	Enterprise and client-server applications	-	-	-	-	X
Software	Embedded systems	X	-	-	-	-
Methods	Controlled experiments	X	X	-	X	X
Methods	Case studies	-	-	X	-	-
Methods	Action research	-	-	X	-	-
Subjects	BSc students	-	X	-	X	-
Subjects	MSc students	-	X	-	X	-
Subjects	PhD students	-	X	-	-	X
Subjects	Professionals	X	X	X	-	X
MD* Tools	UniMod	-	X	-	-	-
MD* Tools	WebRatio	-	-	X	X	-
MD* Tools	BridgePoint	X	-	-	-	-
MD* Tools	AndroMDA	-	-	-	-	X
	Status	✓	✓	🚦	🚦	🚦

TABLE I. Research Plan for Productivity and Maintainability

Finally, we have planned a series of experiments to compare the difference in productivity (if any) between "traditional" development and MD* development in the context of Web applications (4). We have already selected: (1) two classes of BSc and MSc students with specific knowledge in Web application development and (2) a well-known MD* tool (WebRatio). As a next step (5), we will perform an experiment in the context of the enterprise, client-server J2EE applications using AndroMDA and employing more skilled subjects (PhD students and professionals).

III. Conclusion

In this work, we have sketched our research plan and our work in progress about a "massive" evaluation of model-driven techniques, summarizing the results of our first two steps and reporting about future intentions. To the best of our knowledge, this is the first work that tries to give a "massive" empirical foundation to the main claimed benefits of MD*.

References

[1] N. Juristo and S. Vegas. The role of non-exact replications in software engineering experiments. *EMSE*, 16:295–324, 2011.

[2] A. G. Kleppe, J. Warmer, and et al. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley, 2003.

[3] P. Mohagheghi and V. Dehlen. Where is the proof? - a review of experiences from applying MDE in industry. In *Proc. of ECMDA-FA 2008*, pages 432–443, Berlin, Heidelberg, 2008. Springer-Verlag.

[4] F. Ricca, M. Leotta, G. Reggio, A. Tiso, G. Guerrini, and M. Torchiano. Using UniMod for Maintenance Tasks: An Experimental Assessment in the Context of Model Driven Development. In *Proc. of MiSE 2012*, pages 77–83. IEEE, 2012.

[5] F. Shull, J. Singer, and D. I. Sjøberg. *Guide to Advanced Empirical Software Engineering*. Springer-Verlag, Secaucus, NJ, USA, 2007.

[6] F. Tomassetti, M. Torchiano, A. Tiso, F. Ricca, and G. Reggio. Maturity of Software Modelling and Model Driven Engineering: a Survey in the Italian Industry. In *Proc. of EASE 2012*, pages 91–100. IEE, 2012.