Innovations Syst Softw Eng (2009) 5:1–3
DOI 10.1007/s11334-009-0076-5

ORIGINAL PAPER

Introduction to special issue: model-based development methodologies

Ricardo J. Machado · Flávio R. Wagner · Rick Kazman

Received: 1 September 2008 / Accepted: 12 January 2009 / Published online: 10 February 2009

Developing software solutions is not an easy task. Brown [1] names the following difficulties arising in the development of software solutions: (1) understanding highly complex business domains and management of the huge development effort; (2) time-to-market pressures; (3) complexity of target software platforms involving not only new hot technologies, but also a diverse and complex assortment of legacy technology infrastructure frequently kept with poor documentation.

The effective development of today's software applications requires consistent effort to research better approaches, languages, techniques and tools that allow us to face the continuously increasing of complexity. Today, when building large software systems, the main challenge for software developers is to "handle complexity and to adapt quickly to changes" [2]. Model-based development methodologies can be a response to this challenge, as they can "increase productivity and reduce time-to-market", by developing concepts closer to the problem domain than "those offered by programming languages" [3].

A focus on models in the history of software development is not new. An emphasis on models arose when software systems became so big and complex that such projects frequently failed. As a consequence, structured approaches (function-oriented methods and object-oriented methods) to software development appeared to facilitate the production of high-quality and cost-effective systems.

An interest and focus on models has arisen again today due to recent developments that have resulted in the establishment of widely known and accepted standards, particularly those originating from the Object Management Group (OMG), such as the Unified Modeling Language (UML) and model-driven architecture (MDA).

These standards (as well as others) represent, through common agreement and acceptance, the community's best efforts in terms of practices, and have provided the basis for further innovations and developments. These standards also enable reuse of knowledge and artifacts as well as tools' specialization and interoperation, thus providing a significant impetus for further progress. Another key enabler of the movement focusing on models in software development is the availability of more powerful computer-aided software engineering (CASE) tools (currently called IDEs—integrated development environments) supporting the development and management of models and the generation of code.

Model-based development is an important research topic in the software engineering field, and there are many theoretical and practical issues yet to be solved. Additionally, embedded and pervasive software systems are increasing in their impact, due to the numerous applications and services that they provide to the society. Model-based development comprises approaches to software development, which rely on modeling and the systematic transition from models to executable code.

For some, model-based development is considered "the first true generational shift in programming technology since the introduction of compilers" [4], and it can profoundly

R. J. Machado (✉)
Dept. Sistemas Informação, Universidade do Minho,
Campus de Azurém, 4805-058 Guimarães, Portugal
e-mail: rmac@dsi.uminho.pt

F. R. Wagner
Instituto de Informática,
Universidade Federal do Rio Grande do Sul,
Caixa Postal 15064, 91501-970 Porto Alegre, RS, Brazil
e-mail: flavio@inf.ufrgs.br

R. Kazman
Software Engineering Institute, CMU,
4500 Fifth Avenue, Pittsburgh, PA, USA
e-mail: kazman@sei.cmu.edu

change the way applications are developed [5]. Automating many of the complex and routine programming tasks, model-based development allows developers to focus on the functionality that the system needs to deliver and on its general architecture, instead of worrying about every technical detail inherent in the use of a programming language or platform.

The articles published in this issue correspond to extended versions of a selection of papers originally presented in the 2006, 2007 and 2008 editions of the International Workshop on Model-based Methodologies for Pervasive and Embedded Software (MOMPES).

Since its first edition in 2004, this annual workshop series has been collocated with important international events, serving as a forum for disseminating scientific and practical aspects related with the adoption of model-based development methodologies (notation, process, methods, and tools) for supporting the construction of software for pervasive and embedded systems:

- MOMPES 2004 in cooperation with the 4th IEEE International Conference on Application of Concurrency to System Design (ACSD 2004), Hamilton, Canada
- MOMPES 2005 in cooperation with the 5th IEEE/ACM International Conference on Application of Concurrency to System Design (ACSD 2005), Rennes, France
- MOMPES 2006 in cooperation with the 13th IEEE International Conference on the Engineering of Computer Based Systems (ECBS 2006), Potsdam, Germany
- MOMPES 2007 in cooperation with the 10th European Joint Conferences on Theory and Practice of Software (ETAPS 2007), Braga, Portugal
- MOMPES 2008 in cooperation with the 11th European Joint Conferences on Theory and Practice of Software (ETAPS 2008), Budapest, Hungary
- MOMPES 2009 will be held in cooperation with the 31st International Conference on Software Engineering (ICSE 2009), Vancouver, Canada.

The different collocations permit the discussion of the topics embraced by the workshop within distinct research communities. This broad scope of the MOMPES workshop series allows different perspectives, cultures, and approaches (both practical and theoretical) from different knowledge areas to be discussed and reported. This is one of the original objectives of the MOMPES workshops: to gather researchers from industry and academia, and from diverse backgrounds, to advance knowledge about the application of model-based development methodologies to the development of software for pervasive and embedded systems.

The MOMPES workshop series considers several areas of interest: meta-modeling and tools; feature modeling; modeling of non-functional requirements; model-based architectures; model transformation and code generation; software process, product lines and frameworks; model-based analysis, testing and verification; model-based optimization, design spaces and design rules; models for adaptable and predictable system quality; models for adaptive system infrastructure; models for ultra-large scale systems; issues in socio-technical ecosystems; case studies and demonstration cases.

The seven papers that have been selected to appear in this issue cover several topics related to model-based development methodologies.

The first paper is "Towards an advanced model driven engineering toolbox" by Jouault, Bézivin, and Barbero. Model-driven engineering is frequently presented as an important change in software development practices. However, behind this new trend, one may recognize a lot of different objectives and solutions. The authors describe the MetaBoxed modeling toolbox that is capable of fulfilling the requirements of forward and reverse engineering as well as of models at runtime.

Next, we hear from Jørgensen, Tjell, and Fernandes on "Formal requirements modeling with executable use cases and coloured Petri nets". Model driven development can offer significant support for the path from user-level requirements, often based on observations of the real world and informal descriptions, via specifications to implementations of software systems. This paper presents the concept of executable use case, which supports the requirements engineering activities by means of a three-tier model-based approach.

The third paper is "Bridging the requirements-implementation modeling gap with object-process methodology" by Soffer and Dori. The software development processes are supported by a variety of modeling approaches, based on the premise that modeling results in better systems. However, due to the significant conceptual and semantic differences between the requirements and implementation models, islands of representation lead to abrupt, harmful transitions between these in the lifecycle. In this paper, the authors propose the integrated modeling paradigm as a solution for bridging the semantic gap between the requirements specification and the possible architecture, design, and implementation models without loosing information captured in these models.

Then, Riccobene and Scandurra write about "Model transformations in the UPES/UPSoC development process for embedded systems". Model-based development emphasizes that software systems should be designed at a high abstraction level and then incrementally refined to contain more specific and detailed information, ending with the implementation of the system in a given platform. Model-based development principles propose the automation of parts of these refinements through automatic model transformations in order to increase both the productivity and the quality of

the developed systems. In the UPES/UPSoC process, reusable model-to-model and model-to-code transformations for embedded system development are supported.

The fifth paper is "MDE for SoC design" by Truscan, Lundkvist, Alanen, Sandström, Porres, and Lilius. Custom architectures combining the modularity of design with programmability and dedicated hardware accelerators for optimal performance have become increasingly popular in the attempt to cope with the complex requirements of the applications. The authors examine the use of model-driven engineering principles to provide support for designing system-on-chip (SoC) architectures. They also test their approach on a custom SoC architecture, called MICAS.

The sixth paper is "A model driven approach for the derivation of architectural requirements of software product lines" by Bragança and Machado. Aligning a software architecture and its functional requirements is a demanding task because of the difficulty in tracing design elements to requirements. This paper explores how a model-driven approach can be applied to derive the architectural functional requirements of a software product line from its requirements.

Finally, Fuentes, Gámez, and Sánchez provide "Aspect-oriented design and implementation of context-aware pervasive applications". Pervasive computing refers to the ability of a computer-based system to be aware of the environment and to process this information and act accordingly. Thus, context-awareness is a recurring requirement in pervasive computing. In this paper, the authors propose the use of the aspect-oriented executable modeling UML profile for designing and simulating pervasive applications, which constitutes the basis for debugging these models at design time, before moving into an implementation.

We expect that the snapshot of cutting edge research in the field of model-based development methodologies that this issue presents is of interest to you and that you enjoy reading all the papers.

We would like to express our gratitude to the reviewers who very carefully and professionally screened all the submissions, and through their insightful comments and suggestions ensured the very high quality of the papers.

We would also like to thank all the authors of submitted papers for their great efforts. They made considerable improvements with respect to the papers originally published in the MOMPES workshops.

Finally, we would like to acknowledge the Editor-in-Chief, Michael G. Hinchey, for generously supporting this special issue.

References

1. Brown AW (2004) Model driven architecture: Principles and practice. Softw Syst Model 3:314–327
2. Schmoelzer G, Teiniker E, Mitterdorfer S, Kreiner C, Kovacs Z, Weiss R (2004) Model-driven development of recursive CORBA component assemblies. In: Proceedings of 30th Euromicro conference, pp 170–175
3. Sendall S, Kozaczynski W (2003) Model transformation: the heart and soul of model-driven software development. IEEE Softw 20(5):42–45
4. Selic B (2003) The pragmatics of model-driven development. IEEE Softw 20(5):19–25
5. Atkinson C, Kuhne T (2003) Model-driven development: a metamodeling foundation. IEEE Softw 20(5):36–41