Chapter 14
Reusable Modelling Tool Assets:
Deployment of MDA Artefacts

Miguel A. de Miguel
Technical University of Madrid, Spain

Emilio Salazar
Technical University of Madrid, Spain

Juan P. Silva
Technical University of Madrid, Spain

Javier Fernandez-Briones
Technical University of Madrid, Spain

ABSTRACT

Model driven development attempts to resolve some common problems of current software architectures in order to reduce the complexity of software development: i) how to increase the level of abstraction by centring on software models; ii) how to automate the software development process through the use of transformations and generators; and iii) how to separate domain, technology, and technological concerns so as to avoid confusion arising from the combination of different types of concepts. Model driven development uses two basic solutions to resolve these problems: i) description of specialised modelling languages and ii) model transformations and mappings. For each domain and technology, MDSD (Model-Driven Software Development) requires specific MDA (Model Driven Architecture) artefacts for the definition of specialised languages and transformations that address specific modelling languages and platforms. The application of MDSD in a specific domain and technology combines multiple interdependent MDA technologies (e.g. MOF (Meta-Object Facilities), QVT (Query-View and Transformation), MOF2Text, UML (Unified Modelling Language) extensions, and OCL (Object Constraint Language)); MDSD combines these technologies to construct and improve tools that support the model driven development process adapted to specific domains, technologies, and platforms (e.g. e-commerce, safety-critical software systems, and SOA (Service Oriented Architecture)).

DOI: 10.4018/978-1-61350-438-3.ch014

The maintenance and evolution of software models require solutions in order to integrate all these MDA technologies and avoid dependency on their tools. The various kinds of MDA artefacts have interdependencies (e.g. model transformation and modelling language extensions), which complicate their reuse and their adaptation to new development environments. This chapter proposes solutions for the integration of all MDA technologies based on reusable modelling tool assets, and solutions for deploying artefacts so as to provide modelling tool independence. MDSD must address these problems, because in the near future, the migration of these developments to new development platforms will be as complex as current migrations from one specific run-time platform to another.

INTRODUCTION

A basic objective of model-driven software development is to place emphasis on the model when developing software. This is a change from the current situation, in that it shifts the role of models from contemplative to productive. The goal of model-driven engineering is to define a complete life-cycle method based on the use of various models automating a seamless process from analysis to code generation (Frankel 2003). This discipline puts all the software artefacts in the right place (e.g. business models, architectural models and design patterns) and actively uses them in order to produce and deploy applications.

Models provide solutions for different types of problems: i) description of problems and their concepts, ii) validation of descriptions and concepts represented through checking and analysis techniques, iii) model transformation and generation of code, configurations, and documentation.

Separation of concerns avoids the confusion generated by combining different types of concepts. Model-driven approaches introduce solutions for specialising models for specific concerns and for interconnecting concerns based on model transformations. This approach reduces the complexity of models through specialised modelling activities that are separated. It improves communications between stakeholders by using models to support the exchange of information. However, separation of concerns often requires specialised modelling languages to describe specific concerns, and the interoperability of specialised languages requires tools integration.

MDA proposes a set of languages and technologies (Miguel et al. 2002) to construct of modelling tools that adapt MDSD to specific platforms (e.g. EJB (Enterprise Java Beans), RTSJ (Real-Time Java Specification)) and technologies (e.g. transactions, security). Standards defining such languages are: MOF (OMG 2006), QVT (OMG 2011), MOF2Text (OMG 2008a), OCL (OMG 2010b), UML (OMG 2009a), UML profiles and RAS (Reusable Asset Specification) (OMG 2005). MDSD combines these languages to create artefact infrastructures, applicable in modelling tools, to then construct MDSD environments. However the artefact's dependence on tool's infrastructures makes the artefacts tool-dependent and therefore application models are also tool-dependent. The MDA philosophy to avoid platform dependency based on PIM (Platform Independent Model) and PSM (Platform Specific Models) is not reflected in the development of MDA artefacts. For example, UML modelling-tool facilities, such as profile registration and support of stereotype applications based on modelling framework tools (e.g. EMF (Eclipse Modelling Framework)), make the profiles and the models that reuse the profiles tool dependent. When exporting the models, the profiles must be exported too, and manual adaptations must be done within the model, because the profiles installed in the target tool cannot be reused. This process requires extensive experience working with models and is not feasible for complex models.