

MDA adoption for a SME: evolution, not revolution – Phase II *

Régis Vogel

Enabler, Avenida da Boavista, 1223 4100-130 Porto, Portugal
regis.vogel@enabler.com

Keith Mantell

IBM UK, Hursley Park, Winchester, Hampshire, SO23 7AT, UK
keith_mantell@uk.ibm.com

June 29, 2006

Abstract

In last year's ECMDA Workshop we presented how Enabler enhanced its development process to take advantage of Model Driven Architecture(MDA)[1]. In this paper we present further findings on the adoption of this enhanced process, but also take a broader view of introducing MDA into this SME(small/medium enterprise).

In particular we review how an analysis of the development process led to: an education programme plan for development stakeholders, the definition and integration of a tool chain incorporating Unified Modeling Language(UML) and Object Constraint Language(OCL) tools, the definition of UML Profiles for both Requirements and User Interfaces together with the relevant transformations (both model-to-model and mode-to-text), and an application of all these to the modification of a software module.

Finally we present how the effectiveness of applying MDA is being measured via a number of industrial experiments and discuss the preliminary results of these measurements.

1 Introduction

Enabler tailors software frameworks for Retailers, applying them to particular customers needs. In so doing, it is always looking to improve quality and

*The work presented here has been developed within the MODELWARE project. MODELWARE is a project co-funded by the European Commission under the "Information Society Technologies" Sixth Framework Programme (2002-2006). Information included in this document reflects only the author's views. The European Community is not liable for any use that may be made of the information contained herein.

efficiency within a commercial environment with an ISO 9000 capability. Whatever new technologies are introduced must ensure that the business controls for Enabler and its customers continue to provide a high quality framework for the control of all commercial aspects of a project. Another aim is to improve communications among teams, some of which are geographically dispersed.

Enabler decided to investigate MDA¹ in order to increase its ability to develop and deploy the Java-based Retail frameworks, and chose to do this in part by joining the Modelware project. Modelware was focused on increasing the deployment of MDA in industry by researching new techniques and processes, developing and integrating new tools, and applying these to industrial scenarios in order to gain documented and measured case studies for future adopters.

A key part of Modelware is the use of industrial experiments. These experiments are in a variety of industries involving software companies of different sizes and approaches. Each of these experiments is designed to show how a set of metrics of importance to each company changes due to the introduction of MDA techniques and tools. Enabler is an SME in the Retail software segment. This paper discusses Enabler's approach to its experiments and the initial results.

At last year's conference, we introduced Enabler's experiences of analysing an MDA Process Framework and applying it to an existing business and software development process [1]. In the current paper, we discuss:

- An overview of the experiments
- Education plan for MDA Stakeholders
- Business and engineering metrics for project management
- Further developments of the MDA Process Framework
- Integration of an MDA tool chain
- UML Profile: Requirement profile
- Transformations from models to both documentation and code
- The initial results
- Conclusions

We summarise the challenges and benefits discovered so far, and hope to publish a formal paper on the full results when the Modelware project completes in July, 2006.

2 The Experiments

As mentioned above, industrial experiments are a key part of the Modelware project. Enabler's early work showed that three areas were impacting the adoption of MDA within the company:

¹MDA is a trademark of the Object Management Group(OMG)

- To shift the mind-set and approach from a code centric- to a model-centric paradigm.
- To change the tools they used to do their work
- To adapt the actual development process and the actors' roles

It was important to choose experiments which focused on areas of importance to Enabler's business, areas where improvements could have a notable impact. So Enabler elected to split the experiment into three:

Software Development Study This experiment takes a full module and develops it from top to bottom using MDA. This experiment is still continuing and will be completed by the end of the Modelware project in July, 2006.

Software Maintenance Study This study applies a change request to a module; the addition is performed with and without MDA. This experiment result will show the effectiveness of MDA in providing fast solutions for requirement changes and bug fixing, which is where most development effort is usually spent.

MDA Education Study This will be performed with a non-random group, identified as the pilot educational group, including subjects with varying technical skills and job functions (e.g., project managers, software developers, software testers). The purpose of the study is to capture the perceived value of using MDA in different job functions and technical areas, as well as providing insight into the subject's view on advantages, constraints, and concerns of MDA. The future use of this information includes incorporation into the change management plan that will be used to manage the adoption of MDA across the entire the organization.

In the next section we discuss the key aspects of Education which underpin most of the experiments.

3 Education

In this section we describe how an Education Plan was developed to introduce and prepare team members for their initial usage of MDA; this included explaining the benefits of the approach as well as the new processes, techniques and tools required. Moreover the objective of the education program is to provide Enabler, as a company, with the theoretical and practical knowledge, as well as the corporate confidence, necessary to introduce MDA methodologies and tools in its development processes.

Application of the MDA process to Enabler resulted in identification of the role and responsibility of each participant in the Education plan, ensuring participation of each of the areas impacted by the introduction of MDA.

To capture the learning curve and dissemination, we performed evaluation surveys of the education actions. All feedback has been used as an input for subsequent education actions, in order to increase the efficiency of the training sessions as well as the trainees' satisfaction. Feedback on the "MDA Education Series" has been incorporated, whenever possible, in the subsequent editions.

To facilitate the introduction of the technology, a MDA course with Enabler-specific content has been designed. Additionally, a forum and news group has been created to give visibility of the project to the rest of the company.

We tailored the classroom training and lectures with specific content based on both the profile of the attendees and the objective of the course. We developed a meta model of each course, including the objectives, the duration, agenda, profile of the attendees, pre-requisites etc. There are also newsletters, forums and other means of developing an MDA community.

The education plan includes:

- MDA Education Series: news letters to disseminate to the whole company, covering such topics as MDA Fundamentals, MDA Tools: State-of-the-art and Case Studies of successful projects.
- MDA Training Courses: due to the wide variety of existing knowledge of MDA in Enabler, many courses were provided, including:
 - Introduction to MDA
 - Introduction to MDA for executive
 - UML, focusing on UML 2.0 topics such as MOF, extensions, OCL etc.
 - Tool support: We develop a top to bottom example using a model of the Retail application(Point Of Sale) to provide the trainee with concrete and contextual example
 - * UML tool RSA
 - * Modelware toolchain and language such as OCL, Transformation technologies based on the Modelbus

In addition to educational aspects, there is the need to win "hearts and minds" of current developers to get their "buy-in" to such a change.

4 Metrics

Another key aspect of introducing MDA is providing measurable proof that the introduction has been of benefit to the business. For that reason, a set of business and engineering metrics were investigated and evaluated against a set of criteria: usefulness to the business, accuracy of collection and predictive power.

The following were considered:

effort (man*days) Effort determines the costs of the software development and is the main productivity measure for most IT companies.

cost Cost is typically a function of effort multiplied by resource rate; therefore, it is dependent on effort and will not be considered in this study.

duration Duration is a measure that can be very frequently manipulated by introducing new resources into the scene. In this way, the influence of duration on profitability is actually measured by the effort (sometimes time to market is an important factor). For example, “what effort is required to deliver a component by a given date”.

In fact, the primary measure (recorded via the normal time recording facilities at Enabler) is *effort*, since all the other maybe derived from it.

The following variables are measured:

Tools maturity how mature are the tools being used in the study. This is measured by the time that the user was idle due to bugs in the tool.

Learning curve how fast people learn model driven tools and methodology. Measured by a survey of attendees of each course.

Resistance to change how willing people are to start using model driven development. Measured by a survey of attitudes.

Perceived value of using MDA perceived advantages, constraints, and concerns of using MDA. Measured by a survey of attitudes.

The following demographic data is collected:

Job function (software developer, project/software/quality manager)

Job tenure (number of years in Enabler)

Professional tenure (number of years in software development)

Highest completed education (bachelors degree or equivalent, masters degree or equivalent, doctoral degree, other)

These measures are designed to show how MDA affects the business measures of Enabler - the final result of the studies will be reported in a later paper.

5 MDA Process Framework

As explained in our previous paper[1], it is critical that any new technology be based upon a process which can be added to a pre-existing development process which already satisfies the project management and commercial needs of both Enabler and its customers.

Enabler and IBM UK partnered with the European Software Institute(ESI) to develop the MDD Process Framework (MDD PF) (as outlined in [2]); in turn

the MDD PF was adapted for use within Enabler's existing processes. There has also been a feedback process for topics which Enabler has discovered during the course of their work.

Continuing the MDD PF development, in this section we focus on what Enabler has adapted or contributed to the base Process Framework.

In general, Enabler has not adjusted its own role titles and responsibilities, but rather has mapped the stakeholders, processes and deliverables from the MDA Process Framework to Enabler's terminology where they overlap. One area essential to implementing MDA is understanding who will take on the responsibilities of the meta-team. The role of developing the meta-models that are used to define models at different levels of abstraction, and the role of developing transformations can be performed by people with existing skills (such as analysis and programming) but who need extra education.

The idea of having a "product-line" model, which incorporates a general model of a domain perhaps built up over several projects, and a tailored model for a particular customer is key to Enabler's business. It was essential that the process fully support this way of working. The three partners mentioned above responded to Enabler's requirement and augmented the first version of the process framework.

In a similar manner, the process must include support for modifications or changes to existing systems, rather than just supporting the new development code. As mentioned earlier, this is a key part of Enabler's business, and so has an experiment dedicated to it.

The following areas have been found to be important during the experiments: The use of test cases generated from the use case models, especially for user interface, and Traceability between model artefacts. In both cases, the Enabler process will need to be updated, and the results fed back for potential updates to the MDA Process Framework.

This Process is supported by a toolchain, which we explore in the next section.

6 MDA Toolchain

An early decision of the Modelware project was to use of UML 2.0 for modeling, and UML Profiles to express meta-models.

An important aspect of Modelware has been the development of "ModelBus" [6] to facilitate the interoperability and substitutability of the disparate tools required for MDA. Using the toolsuite via ModelBus proved to be difficult due to many factors. The Modelbus, a new development in Modelware, has continued to evolve throughout the project. This has meant that the co-ordination of the Modelbus with new connections to MDA tools has been difficult to manage. For instance, backward compatibility was broken between versions 0 and 1, and the support for, and transport of, Profiles between tools (improved in version 2) is still not fully supported by attached tools. For these reasons, the architecture used for the experiments was changed, and almost all tools were integrated

within Eclipse. The Modelbus was used for two tools (which were not based on UML, nor used Profiles) and this usage was limited to specific down-level versions of the Modelbus and associated MDA tools.

As reflected in the figure below (which is based on the process framework), the following tools were required: UML tools (RSA), Requirements tools (Mantis), Eclipse, Model Transformations (ATL - Model-to-Model, and MOFScript - Model-to-text) OCL OSLO, Subversion (all integrated using Eclipse), and Maven and Traceability (integrated using Modelbus).

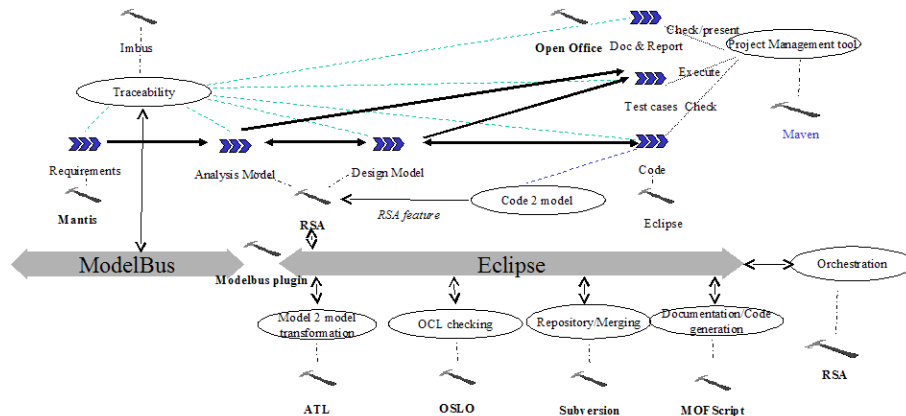


Figure 1: The tool chain for the experiment

The main deliverables include:

Requirements Metamodel A Profile for UML 2 that allows the modeling of stakeholders, requirements, test cases etc.

Requirements document transformation Transformation from a Requirements Model to an Open Document Format (ODF) document. In addition to text, diagrams were also extracted using an RSA Pluglet.

Analysis Profile Metamodel An analysis Profile that allows the identification of the parts of the model that are used to generate the Functional Requirements Document(FRD)

GUI Profile Metamodel Describe logical screen flow, data elements etc.

Model to GUI transformation A transformation from a UML Model with a GUI Profile to a Java framework used to manage the User Interface. Unit tests for the user interface is also generated.

OCL [8]Used for validation of the model and quality control. We believe that constraints on quality control do not have to be implemented in the meta-model but can rather be in an external file because their life-cycle may

be different from that of the meta model. Further, the quality level rules may differ for different projects, but the meta model remains the same. So we have developed an RSA pluglet for quality validation using two files containing OCL Rules, one file for “Warnings” and another for “Errors”. The pluglet allows the user to navigate through any errors and “jump” directly to the element that doesn’t respect the rule, optimizing the validation and correction process.

SVN Subversion supports the merging of models and version management. The Subclipse plugin for Eclipse provides us with graphical and textual views of model differences. We are using this tool to merge manual modifications of a model and wished to use it for generated modifications as well. For instance, if a generated model has been modified by hand, and then regenerated. At the time of writing we encountered a limitation which we are trying to solve - the problem is that our regenerated model seems too different for the tool, even if only a single new class was added.

Orchestration This was done by RSA extensions (pluglets) to execute MDD tasks. This facility was mainly used to invoke transformation processes, which typically consist of calling OCL to validate the quality of the model, then invoking a transformation. By doing so we hide the complexity from the user who doesn't have to know which transformation file to call; for instance the user will call the Generate BRD in a contextual menu.

Traceability This was used to maintain links (both dependency and traceability links) between artifacts created or generated during the MDD process and did so independently of the tools that produced them. This tool/feature enabled us to perform some queries to get, for instance, the impact analysis of a change or identify if a requirement was tested by a test case. The traceability client integrated into RSA allowed us to trace any kind of artifact to any other in the RSA workspace. Automatic update events were managed by the client.

7 UML Profile

In this section we discuss Requirement Modeling Support, a key profile used in the experiments.

7.1 Profile for Requirements

The following summary description (to be elaborated in another paper) gives a perspective on the critical Requirements Profile.

For Enabler it was important to formalise requirements, which were previously gathered in spreadsheets and documents, improve their quality and ease their validation and to allow traceability to other development artifacts. In fact, experience gathering requirements early in the ModelWare project indicated the

power of modelling requirements, giving the advantages of a structured approach and the chance of a more rigorous definition. Requirements could be related to one another and to other artifacts. The meta-model could be implemented in different tools. For instance, in the Mantis (usually for bug tracking) and in IBM's Rational Software Architect(RSA); Mantis allows for easy input of requirements and a workflow to control their status, whilst RSA allows checking of requirement model constraints.

A survey was made of various existing profiles and other works concerning requirements, including SysML and User Engineering (from IBM). With respect to UML 2 (with its Actors and Use Cases) we adopted:

Actors: The notion of Actor was broadened to include not just users of the system, but Stakeholders in general, such as Customers and Suppliers. Actors have attributes for description, definition of responsibilities, skills etc. See Figure 2.

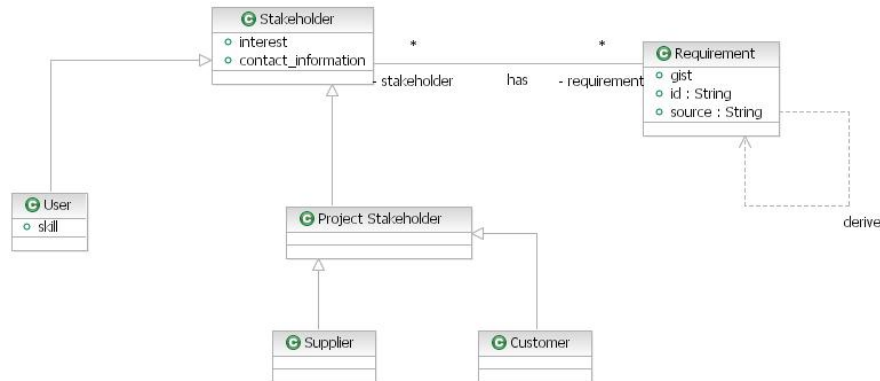


Figure 2: Metamodel of Stakeholders

Goals and requirements: These can be classified - these were inspired by the work of Gilb[4], Architecture Tradeoff Analysis Method(ATAM)[5] etc. Categories include Functional, Performance, Quality etc. By using the ATAM architectural qualities for requirements categorisation, this facilitates subsequent design reviews. See Figure 3

The first version of the export facility for requirements from Mantis to a UML 2.0 model was developed in PHP. The profiles weren't used, due to the profile limitation of the toolchain explained previously. Keywords were used instead and the categories were represented as packages. Currently the export is using a variation of the presented profile that corresponds to the actual classification of the requirements defined in Enabler's methodology. Our goal here was, in the first iteration of the MDA tool chain, to avoid changing too much of the process (evolution not revolution). We hope to evolve to the profile that we proposed in the previous paragraph.

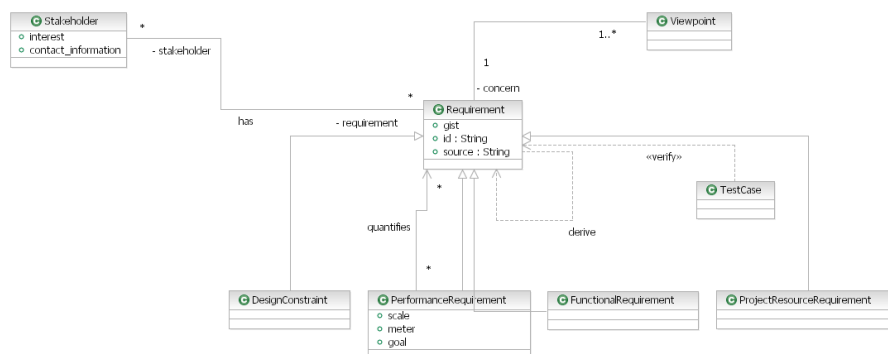


Figure 3: Metamodel of Requirements

8 Transformations

8.1 Model to Text

This section discusses producing Open Document Format (ODF) files as part of the Business Requirements Document used with Enabler’s customers. The tool used is MOFScript[9].

Within the Enabler experiment, model-to-text transformations are used to generate documentation based on the company’s standard templates. One major requirement for the generated documentation is to follow this template as far as possible. Besides maintaining the template, the problem of merging into the document had to be solved, because part of the document is maintained by hand and the other part is generated from the model. To solve part of the problem we adopted Open Office instead of Microsoft, because Open Office uses the Open Document Format(ODF)and provides the master document facility. The advantage of the ODF is that it’s an ISO standard and also ”cleaner” than RTF. ODF is XML-based and easier to generate. Also the master document concept of Open Office allows users to divide the BRD structure in different sub-documents that are autonomous. That is how we solved the problem of integrating generated and handcrafted parts into one master document. Another benefit is providing a more readable final document by generating complete documents and avoiding empty sections.

In addition to documentation, model-to-text code generation is used to generate Java as well as the configuration files needed in the architecture. At the end of the experiment Java (including unit-testing), configuration files, SQL files, and perl files will be generated.

8.2 Model-to-model

The model-to-model tool selected is ATL[7]. We use it to generate analyses used to design models especially for the GUI, with the profile for user interface. However, the limitations and bugs of the tools for managing models with profiles meant we changed our focus to other tools (such as model-to-text). Also, the merging problem caused a loss of information. So we decide that, for the Maintenance experiment, the models will already be present and will only be updated manually. We hope to revisit this before the end of the project.

9 Initial results

The experiments were encouraging:

- Interest in MDA has been raised and maintained throughout the project - this was in part due to tailoring the education to specific features of importance to Enabler. Some early non-tailored education had made some participants doubt its relevance to their situation.
- The development teams saw the value of the techniques and processes - this is important in the winning of the hearts and minds of the development community and will ease future acceptance by the broader community
- By applying the process with the models and the traceability facility, the users could focus on just those parts impacted by the change request. Using these techniques gradually presents complexity to the user, allowing greater efficiency and confidence in the implementation. The MDA tools and techniques helped separate concerns and provided guidance in performing the tasks.
- The structure of the models (e.g. Requirements Model, Use Case Model etc.) helped the team cope with the complexity of the application and also to structure their work; where a team member had not worked on that aspect of the framework previously, they also allowed faster understanding and therefore an earlier contribution to the team.
- The Enabler-expanded MDA Process Framework allowed team members to adopt MDA quickly.
- Many obvious tasks can be automated by transformations and by validations, allowing the developer to focus on higher value tasks and ultimately boosting morale.
- The traceability aspect was felt to be very valuable as it focuses attention on key points of value to the stakeholders. This was felt to be a clear advantage over previous development cycles. The traceability tool was integrated with the modeling tool, allowing the navigation of traceability links and the impact analysis of changing model elements. However, the

quality of these links is currently only as good as the manual entry of the information

- The requirements tool (Mantis) was appreciated for gathering and refining requirements. The transformation to a requirement model is key with automatic validation and verification of the requirement data. Automatic production of the Business Requirement Document is important but also the fact that using the traceability feature, the requirement model is the entry point to navigate to the down-stream models. The experiment team felt this to be a great improvement over their current approach.
- The OCL Validation helped in the review of the quality of the work done, but as soon as the users understand the mechanism they use it like the results of a compiler: to show problems in order to be able to solve them.
- Feedback on Subversion(SVN) was positive since it allowed the merging of models both at the level of the model element and also the diagram
- The Mofscript editors autocompletion in function of the metamodel was appreciated as it relieves the user of memorising the structure of the meta-model.
- Whilst the toolset shows great promise, it is still somewhat fragile as might be expected at this stage in its life-cycle. For instance, there are still issues of interoperability among tools which makes assembling a toolchain quite labour intensive and limits the functions available. Similarly, after two years some tools are still only alpha quality.
- Some documentation and education materials were incomplete, which caused a steep learning curve and/or required the special creation of supplementary materials.

In addition, more functions and better performance will be required in some areas. In terms of functionality, the ability to merge models allows the preservation of information that had previously been added. For instance, for traceability, none of the transformation facilities produce traceability links automatically; these links had to be added manually to the traceability database for these experiments. Tool support for analysing patterns of traceability links would be useful as well. The administration of the Modelbus and connected tools are necessary points to improved, especially errors reporting, and visibility of exchange status between tools.

10 Conclusion

This paper has discussed the progress made on introducing Model Driven Development to an SME. There are many facets to consider, including business goals, project management controls, personnel, tool support etc. The final results are not known yet, but so far the technique and process results have been positive.

However, the stability and compatibility of the toolchain need improvement before putting it into a production environment.

References

- [1] Vogel, R., Mantell, K.: MDD process in an SME: evolution, not revolution Phase I. ECMDA-FA 2005, Nuremberg, Germany, November 7-10, 2005, (http://www.enabler.com/en/skills/ecmda/PAPER_Enabler.pdf) (document), 1, 5
- [2] Mansell, J., et al.: A Process Framework for the successful adoption of Model Driven Development, ECMDA 2006. (To appear) 5
- [3] Aagedal, J., Solheim, I: New Roles in Model-Driven Development. Second European Workshop on Model Driven Architecture, 2004 (<http://www.cs.kent.ac.uk/projects/kmf/mdaworkshop/submissions/Aagedal.pdf>)
- [4] Gilb, T.: Competitive Engineering. Butterworth-Heinemann, 2005 7.1
- [5] Clements, P., Kazman, R., Klien, M.: Evaluating Software Architectures. Addison Wesley, 2001 7.1
- [6] MDDI - Model Driven Development integration (<http://www.eclipse.org/mddi/>) 6
- [7] ATL : Atlas Transformation Language (<http://www.sciences.univ-nantes.fr/lina/at1/>) 8.2
- [8] OSLO - Open Source Library for OCL (<http://developer.berlios.de/projects/oslo-project/>) 6
- [9] MOFScript (<http://www.eclipse.org/gmt/mofscript/>) 8.1