General purpose MDE tools

Elías Palacios-González, Héctor Fernández-Fernández, Vicente García-Díaz, B. Cristina Pelayo G-Bustelo, Juan Manuel Cueva Lovelle, and Oscar Sanjuán Martínez
Web engineering research group - Oviedo University -

***Abstract* —MDE paradigm promises to release developers from writing code. The basis of this paradigm consists in working at such a level of abstraction that will make it easyer for analysts to detail the project to be undertaken. Using the model described by analysts, software tools will do the rest of the task, generating software that will comply with customer's defined requirements. The purpose of this study is to compare general purpose tools available right now that enable to put in practice the principles of this paradigm and aimed at generating a wide variety of applications composed by interactive multimedia and artificial intelligence components.**

***Keywords*—MDA, MDD, MDE, Software Factories**

XVIII. Introduction

Model Driven Engineering (MDE) is a new paradigm which opens new expectations in software development. The aim of this idea is to release the developer of the boring routine of writing in a specific programming language the software that complies with the requirements specified by the customer. If needs are correctly in a language that machines can interpret, what would happen if we let them translate it to a programming language?

Although its an emerging technology, in the last few years its popularity has caused the creation of a considerable number of tools based on it. The purpose of this study is to compare the general purpose tools that are available right now.

The remaining part of this paper is organized as follows: In section 2, we present a brief review of the main concepts of this paradigm, to understand the technical terms that will be used afterwards. In section 3 we review related researches and set the tools and features to be evaluated. In section 4 we show a comparative study. In section 5 we discuss the results and in section 6 we show our conclusions.

XIX. Background

In the mid 60's "The Software Crisis" [14] came apart. Software development had got to a point where mostly all resources where spent in its maintenance instead of its creation.

The origin of the problem was the fact that software production was intended to create code as fast as possible to deliver it to customers. The result was a poor quality software, which carried out the basic needs of the customer, with many errors and not very reliable. Its exploitation in most cases meaned a long process of debugging and change. This process degraded even more code quality, and made very difficult to maintain by other developers.

Then, Software Engineering appears [17], consolidating methodologies and increasingly sophisticated tools designed to help engineers to work with a high-level overview of systems which development they have to organize and run.

Software engineers design high-level documents supported by graphics languages such as UML [11]. However, in most cases, this initial documentation is not maintained anymore when it comes to the later stages of development in which we must codify and deadlines beset. The result is a rift between high-level documentation and the real state of the project.

All these problems are compounded by the size and complexity of the projects, emphasizing that the solution does not seem likely to be focused on the code despite advances as significant as the object oriented software and design patterns.

Finally, in recent years a new paradigm has appeared which tries to find a solution focusing on models: Model Driven Engineering (MDE). Concerning this concept, several proposals to implement appear, but the most important are: Model Driven Architecture (MDA) and Software Factories, one opposite to another.

A. Model Driven Engineering

The Model Driven Engineering (MDE), proposes to focus software development on models, rather than on code. In this context a good definition of a model is provided by the Object Management Group (OMG) [8]: "A model of a system is a description or specification of that system and its environment for some certain purpose. A model is often presented as a combination of drawings and text. The text may be in a modeling language or in a natural language."

From these models, combining two basic aspects [16]: **Domain-Specific Modeling Languages** (DSML) to formalize the application structure, behavior, and requirements within particular domain, and **transformation engines and generators**, to analyze certain aspects of models and then synthesize various types of artifacts, which can range from source code to alternative model representations.

The aim is to provide developers with the definition and construction of a high level system module, and from then on, create successive models with a lower level of abstraction each time, reaching at last, a model directly executable on a physical machine.

B. Model Driven Architecture

Model Driven Architecture (MDA), is an OMG proposal [8] which seeks to standardize an implementation of MDE. The three primary goals of MDA are portability, interoperability and reusability through architectural separation of concerns.

MDA defines three types of models, each with a higher level of abstraction than the next:

The **Computation Independent Model** (CIM) shows the system but does not show details of its structure. It is used to answer the following question: What makes the system?

The **Platform Independent Model** (PIM) shows the systems logic and its interactions with other systems but without detailing what kind of technology it will use or if it adapts itself to a particular platform. It is used to answer the following question: How does the system do what was defined in CIM?

The **Platform Specific Model** (PSM) which combines PIM specifications with the details that specify how to use the system with a particular type of platform.

To define these models, the technology proposed by OMG is MOF [10]. Each of the Domain-Specific Modeling Languages that are used to describe it (UML, SPEM, etc…), will be defined by their corresponding metamodels which in turn will be instances of MOF.

Transformation of models is the key to the process, which consists of converting a model in another model of the same system. The MDA tool must generate one or more PSMs from a PIM along with a set of transformation rules. The same tool or a different one will generate an executable model (source code) of the application from PSM together with additional information.

The standard language that proposes for the definition of OMG transformation models is the Query / View / Transformation (QVT) [10], which in turn is based, on the restrictions language Object Constraint Language (OCL) [12].

C.Software Factories

Software Factories are an alternative to MDA leaded by Microsoft. Both are opposite proposals, but an objective analysis of these [9] leads to the conclusion that both are not as antagonist as the confrontation of OMG vs. Microsoft could make us suppose.

A Software Factory, is defined as [3] “A software product line that provides a production facility for the product family by configuring extensible tools using a software template based on a software schema.”

Software Factories focus on the development of similar systems by promoting reuse of architectures, software components and knowledge. In this case the role of the model is to define the design of the parts of the system that extend the functionality of the patterns already developed.

XX.MDE Tools

To compare the tools, it is necessary to choose which are the appropriate and which are the characteristics that are going to assess for each of them. We have reviewed the related work in recent years and we have selected a battery of features to implement this study.

A.Related Work

There have been various studies characterizing and/or comparing development tools oriented models. Most of these are focused toward MDA tools that are without discussion, the most popular.

Stuart Kent’s work [5] provides a first classification on the basis of the aid that each tool provides the developer.

The study of King's College [6], is interesting but is dedicated to a single tool, OptimalJ. The evaluated features are drawn from the specification of MDA [8].

The work of Czarnecki and Helsen [2] is geared to establish a classification approach of transformation models which use the tools.

Molina el al. [7] conducted a comparison between OptimalJ and ArcStyler based on the characteristics assessed in the study of King's College [6] that added three additional features

Tariq and Akhter in [18] establish a set of characteristics that they understand are essential for a MDA tool. This study was applied to 10 commercial tools.

Wang, W. compared to the extensive work [19], 6 tools focused on the UML models transformation.

The work done by Herrera al. [4] is a compilation of the characterizations made in most of the earlier work and brings some new. However, the comparison focuses exclusively on open-source tools.

Quintero and Anaya in [15] do a categorization of the tools based on the usefulness for the end user. They conducted a compendium and the selection of evaluation criteria used in previous studies. Finally apply the comparative to 10 tools, 2 for each of the categories.

Bollat et al. in [1] conducted a comparison using features drawn from previous work and the specification of MDA [8]. These features are studied in three tools in order to assess whether any of these can be adapted to a particular architecture.

Finally, in other work [13] we had an extensive review of tools and collected many features defined in the other works previously detailed.

B.Reviewed tools

Of all the tools seen in previous work, we have chosen for our study those that meet a series of requirements that we believe are suitable for automating and supporting the entire MDE development process.

The required features were as follows:

2) **Models level**: The tool can handle CIM, PIM and PSM models.
3) **M2M**: Tool performs transformations between models.
4) **Applications type**: It must be generalist, being oriented to implement the widest possible coverage.
5) **Graphical editor**: Integrates a graphical editor of models (typically UML), or a module that allows it.

We have decided to collect two sets of features of the selected tools. The first group of characteristics identified the tool and give idea about its origin and agility in its development and maintenance. They are as follows:

- **Name**: Name of the tool.
- **License**: Type of license under which it is distributed.
- **Release**: Release now available.
- **Updated**: Date has been released the current version. This indicates whether the tool has been recently updated or on the other hand takes time stalled its development.

- **Last year releases**: Number of versions that have been released in the last 12 months (September 2007-September 2008). This indicates whether the tool has been frequently updated or on the other hand takes time stalled its development.
- **URL**: Internet address where people can get more information and eventually download the tool.

The second group is purely technical features and we considered important in deciding whether to choose one or the other. They are as follows:

- **Models language**: Languages used to represent models, both textual and graphical.
- **Platform use**: Platform on which the tool can be used.
- **Platform purpose**: Platform for which the tool is capable of generating code.
- **Language purpose**: Programming languages that is capable of generating.
- **Other features**: Other features that may be considered relevant to the tool.

XXI. Comparison

The tools selected from among those available today [20], [21], [22] and that meet the requirements are ArcStyler, Borland Together, AndroMDA and Eclipse Modelling. OptimalJ, although it meets the requirements has been dismissed for having been discontinued.

A. Identifity features

This set of features (Table I), identify the tool. Also give ideas on the origin of the tool and agility in its development and maintenance.

TABLE I
Identifity features

Name	License	Release	Updated	Last year release	URL
AndroMDA	BSD	3.3	21/04/2008	1	http://www.andromda.org
ArcStyler	Commercial	5.5	11/04/2006	0	http://www.interactive-objects.com/products/arcstyler/arcstyler-overview.html
Borland Together	Commercial	2008	08/06/2008	2 (2007 and 2008)	http://www.borland.com/us/products/together/index.html
Eclipse Modelling					http://www.eclipse.org/modeling/

B. Technical features

This is the group of features (Table II) that are considered important in deciding whether to choose one or another tool.

TABLE II
Technical features

Name	Metamodels	Platform Use	Platform purpouse	Languaje Purpouse
AndroMDA	UML 1.4 (XMI); All EMF-based	JVM	J2EE; .NET; Other	Java; C#; etc..
ArcStyler	UML 1.4	Windows; Linux; IBM RSM	J2EE; .NET	Java; C#
Borland Together	UML 2.0 (XMI 2.0)	JVM	J2EE; .NET; Other	Java, C++, and C#
Eclipse Modelling		JVM		

XXII. Discussion

Now we will comment each of these tools, emphasizing its strong and weak points.

AndroMDA is a powerful and versatile tool with a consolidated cartridges system that allows generate code for different architectures. Although it doesn't have a graphical editor, it supports UML 2.0 and Eclipse EMF based tools. One possible drawback derived from its own versatility and complexity is an abrupt learning curve.

Their development also seems somewhat stalled. After the release of version 3.2 in November 2006, a beta version was released of the expected 4.0 in May 2007. Almost a year later, in April 2008 revision 3.3 was released with some minor changes and corrections. Yet there has been no stable release of version 4.0.

ArcStyler supports multiple platforms and a very friendly interface. Stresses for its ease of use, but it seems that their development is stalled, with its latest version of April 2006.

Borland bet with force by MDE with its tool Together, for which has released two versions in the last year. It presents an excellent support for PIM using BPMN and a user-friendly interface based on Eclipse.

We also take in notice its implementation of OCL 2.0 and the use of QVT for model-to-model transformations.

Eclipse modelling is not exactly a tool, but a framework which includes many of these tools. But being a very active project, with a large amount of free and advanced tools, has moved us to include it in this work. To use this toolkit, we must have a good knowledge of the technologies being employed and dedicate a large amount of time in collecting and preparing the proper plug-ins.

XXIII. Conclusions

From our point of view, the most advanced generalist tool, easy to use and which meets the requirements, is now Borland Together 2008. It specially emphasizes for its support for PIM and implementation of standard MDA.

If you want to use free software, AndroMDA and Eclipse Modelling tools are very nice options, although they need more knowledge and dedication.

Anyway we should be alert for the of a stable version of AndroMDA 4.0 with better support of PIM.

References

[1] Bollat, V.; Vara, J. M.; Vela, B. & Marcos, E. (2007), Una revisión de herramientas MDA, in 'Actas del II Taller sobre Desarrollo Dirigido por Modelos, MDA y Aplicaciones (DSDM'05)'.

[2] Czarnecki, K. & Helsen, S. (2003), Classification of Model Transformation Approaches, in 'Workshop on Generative Techniques in the Context of Model-Driven Architecture (OOPSLA'03)'.

[3] Greenfield, J.; Short, K.; Cook, S. & Kent, S. (2004), Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools, Wiley.

[4] Herrera, J. C.; Matteo, A. & Díaz, I. (2005), Una Caracterización de Herramientas MDA de Código Abierto, in '8º Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software (IDEAS´05)'.

[5] Kent, S. (2002), Model Driven Engineering, in 'IFM '02: Proceedings of the Third International Conference on Integrated Formal Methods', Springer-Verlag, London, UK, pp. 286--298.

[6] London, K. C. (2003), 'An Evaluation of Compuware OptimalJ Professional Edition as an MDA Tool.', Technical report, University of York.

[7] Molina, J. G.; Rodríguez, J.; Menárguez, M.; Ortín, M. & Sánchez, J. (2004), Un estudio comparativo de dos herramientas MDA: OptimalJ y ArcStyler, in 'I Taller sobre Desarrollo de Software Dirigido por Modelos, MDA y Aplicaciones (DSDM)'.

[8] Mukerji, J. & Miller, J. (2003), 'MDA Guide V1.0.1'(omg/03-06-01), Technical report, OMG.

[9] Muñoz, J. & Pelechano, V. (2005), MDA vs Factorías Software, in 'II Taller sobre Desarrollo de Software Dirigido por Modelos, MDA y Aplicaciones (DSDM)'.

[10] OMG (2005), 'Meta Object Facitity (MOF) 2.0 Query/View/Transformation Specification. Final Adopted Specification.', Technical report, OMG.

[11] OMG (2003), 'UML 2.0 Superstructure Specification'.

[12] OMG (2003), 'UML 2.0 OCL Specification. Final Adopted Specification (ptc/03-10-14)'.

[13] Palacios-González, E.; Fernández-Fernández, H. ; García-Díaz, V. ; Pelayo G-Bustelo, B.Cristina & Cueva Lovelle, J. M. (2008), A review of Intelligent Software development tools, in 'The 2008 International Conference on Artificial Intelligence (ICAI'08). The 2008 World Congress in Computer Science, Computer Engineering, and Applied Computing (WORLDCOMP'08) ', Las Vegas, Nevada, USA.

[14] Pressman, R. S. (2002), Software engineering (5rd ed.): a practitioner's approach, McGraw-Hill, Inc., New York, NY, USA.

[15] Quintero, J. & Anaya, R. (2007), Marco de Referencia para la Evaluación de Herramientas Basadas en MDA, in 'X Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software'.

[16] Schmidt, D. C. (2006), 'Guest Editor's Introduction: Model-Driven Engineering', Computer 39(2), 25.

[17] Sommerville, I. (1989), Software Engineering, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[18] Tariq, N. & Akhter, N. (2005), 'Comparison of Model Driven Architecture (MDA) based tools', Technical report, Institute of Technology y Karolinska University Hospital. Stockholm.

[19] Wang, W. (2005), 'Evaluation of UML Model Transformation Tools', Technical report, Business Informatics Group, Vienna University of Technology.

[20] (Recovered in 02/2008), 'http://www.omg.org/mda/committed-products.htm'.

[21] (Recovered in 02/2008), 'http://www.codegeneration.net/generators-by-standard.php?standard=1'.

[22] (Recovered in 02/2008), 'http://www.lcc.uma.es/~av/MDD-MDA/'.