Supporting Stakeholders in the MDA Process

Keith Phalp, Sheridan Jeary, Jonathan Vincent, John Mathenge Kanyaru,
& Simon Crowle

Software Systems Modelling Group,
Bournemouth University, Poole, Dorset. BH12 5BB
kphalp;sjeary;jvincent;jkanyaru;scrowle @bournemouth.ac.uk

Abstract

By providing a route from model to architecture, Model Driven Architecture (MDA) promises, among other things, to narrow the gap between business users and software developers. That is, the approach is intended to involve stakeholders in modelling, so that, by transformation from Computationally Independent Models (CIM), through Platform Independent Models (PIM), to Platform Specific Models (PSM), the resultant software systems will be aligned with the client's business needs.

However, there has been little attempt to consider who the stakeholders in an MDA process would be, or whether the models and supporting tools used would be appropriate to their backgrounds, expertise and needs. In contrast, this paper suggests a classification of stakeholder roles and their relationship to MDA phases. The paper then presents a brief review of existing toolsets, and ties the support offered to each of the MDA modelling levels; CIM, PIM and PSM.

In brief, we find that whilst there appears to be good support for what amounts to traditional development, at PIM and PSM, there is still relatively little support for CIM. Since a proportion of key stakeholder roles appear to be working at this level, it would appear that MDA is lacking in providing appropriate support for, a significant, and perhaps the most vital, subset of stakeholders.

1.0 Introduction to Model Driven Architecture and Requirements

Model driven development (often referred to as Model Driven Architecture and referred to in the rest of the document as MDA) provides mechanisms for formalising the abstraction level of models in software development, and for managing transformations among them (that is, moving from the Computationally Independent Models (CIM) through Platform Independent Models (PIM) to Platform Specific Models (PSM). However, MDA initiatives tend to focus on modelling notations which are clearly within the domains of those already versed in software development. For example, class diagrams still feature heavily [1]. Similarly, tools for MDA tend to support traditional software engineering notations, largely those within the UML, and transformations, downstream, from them. While this clearly offers many advantages in providing a systematic development path, it is not clear that there are distinct advantages offered in the earlier requirements phases.

As a first step in understanding this issue, this paper considers who (or what roles) would be found in MDA projects, by suggesting classes of stakeholders. Where other authors have sought to identify key stakeholders in the MDA process, they have, in all of the papers we have found [2,3], taken only a software development perspective, and identified roles within the development team, with the most upstream roles considered being something akin to a systems analyst. Note, however, that whilst we focus on MDA development roles, there are existing taxonomies, e.g., [4,5] which describe the complexities of stakeholder impact, (many of whom are outside the development) in far greater depth than here. Indeed, our rationale for identifying specific MDA user groups was motivated by our intention, as part of a collaborative European Commission Framework 6 project VIDE [6], to produce tools to support model driven development and, specifically, to enhance the involvement of non-technical stakeholders in the development process. Hence, we believe that the explicit identification of such roles, as described within this paper, is an important first step towards understanding their needs, and ultimately supporting development with appropriate models and toolsets. In particular, we attempt to understand the nature of modelling, be it at CIM, PIM or PSM levels.

This paper then considers whether MDA is sympathetic to the needs of those stakeholders, in terms of the models and paradigms offered to them. Given the promise of MDA, it seems particularly important to ensure that all of the people, or at least all of the stakeholder roles, can be identified, such that one may maximise their involvement in requirements determination and validation, and in the production and validation of the models which will drive subsequent development. The remainder of the paper is as follows. Section two examines briefly the importance of stakeholders and of identifying stakeholder roles. Section three then considers those stakeholders specific to MDA and maps these roles to MDA phases. Section four examines existing toolsets, and section five considers the appropriateness, or otherwise, of existing approaches. Section six offers some conclusions.

2.0 The Importance and Identification of Stakeholders

Much of the software engineering literature recognises that the requirements phase is particularly important to a successful project outcome and that inadequate determination of requirements is the biggest single factor in project failure. For example, Glass [7] notes that many software projects fail due to poor or non-existent requirements processes, and Hall, Beecham et. al. [8] report that 48% of development problems are in the requirements phase. Research in the area agrees that one way to improve the requirements process is to increase the involvement of stakeholders [9] in both the elicitation of requirements and the validation of specifications.

The software engineering community has long understood the importance of stakeholder involvement in validation of requirements and specifications, and development of quality systems (see [10,11,4,12]) and some enlightened authors go so far as to explicitly define requirements as “the effects that stakeholders wish to be brought about in the problem domain” [13,14]. Indeed, [15,5] even argue that perhaps the most common single mistake in development efforts is to leave an essential person [stakeholder] out of the process. There is significant work in requirements engineering (and related fields), over many years, which supports this view. For example, the CORE approach [16] to requirements engineering specifically attempted to make explicit the multiple perspectives of differing individuals (stakeholders). Similarly, Checkland’s Soft Systems approach [17] (and its extension by [18]), has at its core the idea that different stakeholders will have contrasting worldviews and perspectives.

However, although it may be widely recognised that understanding ‘who your users are’ is vitally important in the development of the software product [19], to actually identify those people that may have requirements for any system (that is, to determine who are the stakeholders) is far from trivial. In fact, stakeholders can be identified as anyone that could be materially affected by the implementation or outcome of a new system [20]. Traditionally, the customer and user would be the only people identified as having a requirement on the system. However, it is a mistake to class users as a homogenous group. Two broader groups, containing a selection of roles (as identified by [21]) are involved in any system development. Firstly, people on the development side, including: programmers, systems analysts, business analysts, project managers, senior IT management and the chief information officer. Secondly, there are those people from a business for whom the system is required. Further definitions in [21] classify these users as individuals that utilise output or outcomes of an interaction with the system. These will include business users, business management and business strategy management. In addition, there may be external users, who are outside the boundary of the company, which the system will serve. For example, customers or potential customers, information users, trusted external users, shareholders or other sponsors (even the society at large), that are affected by the system.

The following section attempts to provide an indication of roles of typical stakeholders within model driven development approaches. An added dimension for MDA is that there are various levels (CIM, PIM and PSM) or phases within the development, with models being transformed from one phase to the next. Hence, stakeholders are also placed within this hierarchy and their typical skills, experience and goals are articulated. Note, however, that since we wished to concentrate on actual, or potential, modellers (or tool users), only a single role (Domain User) represents the breadth of stakeholders who are not part of the development team.

3.0 MDA Stakeholders

The stakeholder descriptions below are based on roles in an MDA process as defined by the OMG and applied to the domain of business software. Our efforts at identifying such roles were further informed by the experiences and opinions of all partners within the VIDE project [6]. Clearly, any delineation into categories is somewhat arbitrary. However, since partners cover a range of business domains and a breadth of development activities, we have reasonable confidence in the scope covered. We were most concerned with coverage of the development lifecycle, from business process modelling (of which there was extensive experience in the consortium) through traditional analysis and requirements activities (our main focus) to traditional software development.

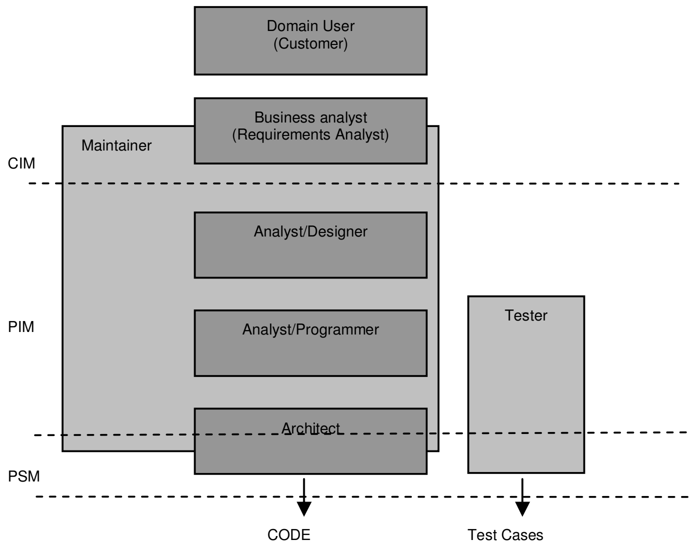

Figure 1. User Roles and MDA levels

Figure 1 (taken from VIDE deliverable 1) depicts the involvement of stakeholders within the MDA levels. Starting from obtaining (informal) requirements from domain users the behavioural model is extended in each step and by the specific role until it can be transformed into coding. The user roles are described in more detail in the following sections.

3.1 Domain User (Customer)

The domain user is the end user of the constructed software solution. They work for the customer and are experts in their special domain. For example, an insurance salesman knows about his company’s offers and legal regulations and is supported by software solutions without any knowledge of technical realisation. The domain user normally has no knowledge about business modelling but can draft the requirements for a software application. In combination with a business consultant the CIM level models can be constructed. The language and the graphical representation should be easy to understand so that domain users can validate the correctness of the models. Since domain users usually use specific vocabulary all tools should support translations into the domain specific language. The domain user serves as a software tester for acceptance tests, i.e., reviews whether a simulated model performs the expected tasks.

A domain user has special skills in his field of work, often knows about business economics and enterprise management but has normally only office application skills. Experience in modelling business processes can not be assumed.

3.2 Business Analyst

Business Analysts advise enterprises on the analysis, conception and implementation of IT solutions. They constitute the connection between the customer and the involved IT specialists and need technical as well as social competences. A business analyst is one of the main user types envisaged for MDA toolsets. They accomplish interviews with domain users and analyse and model the proposed solution on the CIM level. Since they have knowledge of the modelling of business processes, as well as technical architectures, it should be easy for them to use any supporting MDA tools.

Business analysts should have a variety of different skills to fulfil their diverse tasks. They should have knowledge about business processes, modelling, IT concepts and technologies, procedure models, project management and business economics. Beyond these technical skills they require social competences like leadership, team organisation, partner management or knowledge of legal regulations. Typical tools used by a Business Analyst are business rule management systems and business process modelling tools.

3.3 Analyst/Designer

Analysts/Designers are responsible for the conceptual models of business entities and the high level business logic. They use design artefacts and models produced by the business analyst and transform them into designs. The software designer is also responsible for deciding if predefined components may be reused or composed or if they need to be re-implemented. The role of the software designer is often combined with that of the programmer especially in smaller development projects and organisations.

The software designer is a PIM level expert with a strong background in conceptual modelling and UML class diagrams. The software designer defines the first level of behaviour, but leaves the details to the programmer. For reusing or composing new applications from pre-existing components the designer has an understanding of CIM level artefacts.. Typical tools for a software designer are graphical modelling tools.

3.4 Analyst/Programmer

The Analyst Programmer is responsible for completing the behavioural modelling that will allow model simulation (i.e., for testing) and the transformation of the models into code. The Analyst Programmer is a PIM level expert with a strong background in behavioural modelling. The Analyst is one of the main users of tools for detailed behavioural modelling. The Analyst uses the format that is most appropriate for that task.

Analysts will also implement components defined by software designers. Therefore, they will model the behaviour/business logic of the interfaces that have been designed and also provide the documentations for the components.

3.5 Architect

The architect is an expert in the target platform (for example, Struts) and the programming language (for example, Java) but also has a good understanding of UML. An architect works in application or systems development. The architect is the expert for the PSM level. The architect should have knowledge of different target platforms and programming languages. Experience in technical system specification and implementation of the proposed solution is mandatory as well as knowledge about programming concepts like software testing methods for quality assurance.

4.0 Tool Support for Model Driven Development

The principal aim of this study of MDA tools is to determine the extent to which the MDA approach is supported by the tools currently available in industry, the extent to which existing MDA tools adapt a mainly visual development approach (often amenable to those without IT expertise), and the extent to which we view such environments to be accessible to business people. Within this paper, the scope of the review concerns functional support for MDA development.

An initial source of tool vendors is the list of contributors to Object Management Group efforts, including the MDA approach [22].

Given the wide ranging set of tools currently being researched and developed for MDA development, this paper focuses on mainstream tools, that is, those which exhibit a thriving support base or active support consortium. Hence, tools solely developed during academic research are omitted since these more frequently form the basis for further industrial research. Similarly, we omit tools which, whilst they may feed in to some part of the MDA process (e.g., requirements tools), are not principally MDA tools.

As mentioned above, a key MDA development task is model transformation; hence, the following analysis also outlines circumstances where tools support (or do not support) transformation from CIM to PIM to PSM, or PIM to PSM. The authors recognise that the main MDA process model is the transformation PIM to PSM to Code; however, some tools support variants of this process (e.g., PIM to

Code), and these are indicated in our analysis where they occur. As a relatively new development paradigm, MDA continues to enjoy much attention, and tool vendors will inevitably continue to evolve their tools to provide further support for MDA type features, or improve on the already provided ones. Hence, this is not definitive statement of the extent to which tools do (or will) support MDA, but rather a snapshot of the state of the art.

4.1 CASE Tool Support for Model Driven Development

This section provides an outline of tools that exhibit MDA capability, alongside their associated providers. Short names have been provided for tools to enable management and layout within the presentation tables' real estate. The tools considered are OptimalJ [23], ArcStyler [24], Constructor [25], Codagen Architect [26], Objecteering [27], Ameos [28], Together Architect [29], XMF Mosaic [30], Jamda [31], PathMate [32], NetBeans [33], Rational XDE Developer [34], Eclipse Modelling Framework [35],Websphere [36] and Aris [37].

Short word	**Full Name**	**Company**
OJ	OptimalJ	Compuware
AS	Arcstyler	Interactive Objects
CT	Constructor	Dot Net Builders
CA	Codagen Architect	Codagen
OG	Objecteering	Objecteering Software [SOFTEAM]
AM	Ameos	Aonix
TA	Together Architect	Borland [Inprise]
XM	XMF Mosaic	Open source
JD	Jamda	Open source
PT	PathMate	IBM
NB	NetBeans	NetBeans & Sun Microsystems
RX	Rational XDE Developer	IBM
EMF	Eclipse Modelling Framework	IBM
WS	Websphere	IBM
Aris	Aris Toolset	IDS Scheer

In order to be able to compare the tools capabilities, a set of MDA features that each tool would be assessed against was adapted from [38]. For example, whether or not each tool provides functionality for model to model transformation, including support for standards such as MOF, and UML. The overall set of assessment metrics can be found at [39]. Table 1 provides a summary of our analysis.

Feature/Support for	OJ	AS	CT	CA	OG	AM	TA	XM	JD	PT	NB	RX	EMF	WS	Aris
CIM					✓	✓	✓					✓	✓		✓
PIM	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PSM	✓		✓	✓	✓			✓	✓		✓	✓	✓*		
UML 2.0	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOF 2.0	✓	✓		✓	✓	✓		✓		✓	✓	✓	✓	✓	
Action Semantics						✓	✓			✓			✓	✓	
UML profiles		✓		✓	✓	✓									
XMI	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓		
CWM															
QVT								✓							
OCL								✓					✓		
PIM →PSM→Code transform	✓		✓		✓				✓		✓	✓	✓*		
PIM←PSM←Code transform	✓		✓		✓				✓		✓				
PIM→Code transform	✓	✓	✓	✓	✓	✓	✓	✓		✓			✓	✓	
PSM→PSM bridge					✓		✓								
Legacy code→PSM transform	✓	✓	✓		✓	✓	✓					✓			
Transform based on patterns	✓	✓			✓	✓	✓	✓	✓						
Traceability of transforms	✓	✓	✓	✓	✓	✓	✓			✓	✓		✓	✓	
Merging of models	✓				✓					✓			✓	✓	
More than one implementation platforms	✓	✓	✓	✓	✓	✓	✓	✓				✓			

Table 1.Overview of tool support for modelling standards

(✓ = Direct support ✓* = Support through 3rd party plug-in)

It is clear from the table that most tools provide MDA modelling based on UML models, and that key requirements for MDA development such as model transformation are also supported by most tools. What is also apparent is that some MDA environments diverge from a PIM to PSM to Code development process to a PIM to Code development pattern. There is no explicit reason given for this deviation from the main MDA process, but our inferences are that the PIM to Code development process provides quicker turn around of application code (akin to agile development).

Many tools do not yet provide any CIM modelling capability. The version of TogetherArchitect evaluated during our study supports CIM modelling with UML activity charts, but our recent assessment of Together 2006 for Eclipse indicates that this version supports Business Process Modelling Notation (BPMN) as well. The Eclipse Modelling Framework 2.2 provides support for CIM modelling with UML activity diagrams. A general point to be made here is that the choice of CIM modelling notation seems to be either UML activity charts or BPMN. Regarding model transformation, our recent study indicates that there is no support for transformation between CIM to PIM. In fact, even within standard UML modelling, there is no way in which a modeller can automatically (or semi-automatically) move from an activity model to say a class model or vice versa.

The main PIM modelling construct is the UML class diagram. The EMF framework provides an interesting feature whereby EMF class diagrams can be constructed (with the Omondo UML modeller), such that Java source code can be generated automatically from those EMF class models. This process is automated, and no other tool provides similar automation. A similar concept for CIM modelling might be useful where the aim would be to generate an executable business model for simulation of business logic as a means of validation with stakeholders.

5.0 Findings: Support for Stakeholders in MDA

In order to understand the ramifications of support offered by tools, it is perhaps worth brief consideration of the somewhat complex relationships among stakeholders. It seems clear that developers have clear understanding of their own (internal stakeholders) and some understanding of the direct contacts with their (business customers). Likewise tool support seems to have a clear understanding of the needs of these (internal) software development roles. However, the business also has a number of stakeholders who, although not directly involved in the development process, may be vital to requirements determination. One of the aims of model driven approaches is to bring together these two groups of stakeholders, and to provide tool support which is both accessible and useful to both. However, it is not always clear who (what roles) will use the tool, at what stage or phase and for what sort of model. Hence, again there is a pressing need to understand the range of stakeholders, and any additional insights into their modelling needs may be a significant advantage in the production of appropriate support mechanisms.

The previous section described support offered by a variety of MDA toolsets. As suggested above, it is also worth bearing in mind that in supporting Model Driven Development (or MDA) complications may arise, both in terms of tool users, and the abstract levels at which the users are envisaged to be working. For example, one would not usually be sure whether stakeholders are to be producing computationally independent, platform independent, or platform specific models, hence, the need to understand the relationship between the stakeholder role and background and the level(s) at which they may, typically, be working.

The analysis of roles, given in section three, reveals that at least two key stakeholders (the customer and the business analyst) are typically concerned only with CIM models. Whilst we have focussed only on those stakeholders directly linked to (or part of) an MDA development process there is (as noted in sections one and two) a much greater variety of stakeholders within the requirements phase than our simple taxonomy suggests. Hence, whilst it might appear that only or two roles (or groups) suffer the wider requirements context is that many varied stakeholders, (beneficiaries, purchasers, users etc.) are involved, and this further strengthens the need to have stronger support for CIM.

It is instructive, therefore, to consider the extent of tool support for CIM. While all tools investigated supported PIM, and the majority PSM, only a minority support CIM, and this is often somewhat limited. Furthermore, whilst PIM to PSM transformations are common, moving from CIM to PIM is often far from trivial, perhaps reflecting the hidden paradigm shift that this often requires.

6.0 Conclusions

This paper considers the support offered to stakeholders within model driven development. As a first step in this endeavour, the paper introduces a description of MDA stakeholder roles, giving typical goals, experience and tasks for these stakeholders. Existing work of this nature tends only to consider stakeholders from a development perspective and overlooks the importance of the business or customer. In addition, this paper attempts to tie these stakeholder roles to their relevant MDA models, CIM, PIM or PSM.

The paper then examines a variety of MDA tools, and summarises the support offered by such tools. A further contribution is the attempt to understand the support offered by tools specific to CIM, PIM or PSM, and thereby to consider the support offered to the various stakeholder roles.

In particular, the findings are that whilst there is excellent support for PIM, and good support for PSM, support for CIM is somewhat lacking in the majority of tools. Hence, at least two classes of stakeholders, customers (and associated customer representatives) and business analysts are not catered for. In addition, whereas transformations from PIM to PSM are supported, moving from CIM to PIM (typically another analyst / designer task) relies heavily on analysts having understanding of both. Hence, whilst MDA tools would appear to offer strong support for software development activities from design onwards, the major issue with software development, that is of facilitating requirements, of moving from domain and business models, to software specification and design is largely overlooked.

Acknowledgements

This work is funded partially by the EC 6th Framework project VIDE (IST-033606).

7.0 References

1. Kleppe, A., Warmer, J. and Bast, W. *MDA Explained: The Model Driven Architecture--Practice and Promise* Boston, Addison-Wesley Professional, 2003.
2. Mellor, S. J. and Watson, A. "Roles in the MDA Process:MDA will make developers more productive not redundant." Retrieved 26 April, from www.omg.org/registration/registration-roles_mda.htm
3. Aagedal, J. and Solheim, I. New Roles in Model-Driven Development. in Second European Workshop on Model Driven Architecture, 2004. Canterbury, UK.
4. Siakas, K. V., Georgiadou, E. and Sadler, C. "Software Quality Management from a Cross-Cultural Viewpoint." in *Software Quality Journal*, 1999, 8: 85-95.
5. Alexander, I. F. "A Taxonomy of Stakeholders : Human Roles in System Development." in *International Journal of Technology and Human Interaction*, 2005, 1(1): 23-59.
6. VIDE. from http://www.vide-ist.eu/index.html.
7. Glass, R. *Software Runaways*. Harlow, Prentice Hall, 1998.
8. Hall, T., S. Beecham and Rainer, A. Requirements problems in twelve software companies: an empirical analysis. in 6th International Conference on Empirical Assessment in Software Engineering, 2002. Keele, Keele University.
9. Nuseibeh, B., J. Kramer and Finkelstein, A. ViewPoints: Meaningful Relationships Are Difficult! in Proceedings of International Conference on Software Engineering(ICSE'03), 2003. Portland, Oregon, USA, IEEE CS Press.
10. Sutcliffe, A. and Maiden, N. Use of Domain Knowledge for Requirements Validation. in Proceedings of IFIP WG 8.1 Conference on Information System Development Process, 1993. Elsevier Science Publishers.
11. Leonhardt, U. Decentralised process enactment in a multi-perspective development environment. in 17th international conference on Software engineering, 1995. Seattle, Washington, United States, ACM Press.
12. Pfleeger, S. *Software engineering: theory and practice*. Prentice Hall., 2005.
13. Jackson, M. *Software Requirements & Specifications:a lexicon of practice, principles and prejudices*. Addison-Wesley, 1995.
14. Bray, I. *An Introduction to Requirements Engineering*. Harlow, UK, Pearson Education Limited, 2002.
15. Gause, D. C. and Weinberg, G. M. *Exploring Requirements: Quality Before Design*. New York, US, 1989.
16. Easterbrook, S. M. 1991. Elicitation of Requirements from Multiple Perspectives. London, University of London.
17. Checkland, P. B. *Soft Systems Methodology in Action*. John Wiley and Sons Ltd, 1999.
18. Avison, D. E. and A.T.Wood-Harper. *Multiview - an exploration in information systems development*. Maidenhead,UK, McGraw-Hill, 1990.
19. Preece, J., Rogers, Y. and Sharp, H. *Interaction Design*. New York, John Wiley & Sons, Inc., 2002.
20. Leffingwell, D. and Widrig, D. *Managing Software Requirements: A Use Case Approach*. Boston, US, Addison-Wesley, 2003.
21. Avison, D. and Fitzgerald, G. *Information Systems Development: Methodologies, Techniques and Tools*. London, UK, McGraw-Hill., 2006.
22. OMG. "OMG MDA Vendor Driectory (http://mda-directory.omg.org/)." Retrieved 7/2006, from http://mda-directory.omg.org/.
23. Compuware. "OptimalJ MDA tool (http://www.compuware.com/products/optimalj/)." Retrieved 08/2006, from http://www.compuware.com/products/optimalj/.

24. Objects, I. "ArcStyler MDA tool (http://www.arcstyler.com/)." Retrieved 08/2006, from http://www.arcstyler.com/.

25. DotNetBuilders. "Constructor toolset for MDA (http://www.dotnetbuilders.com/constructor.aspx)." Retrieved 07/2006, from http://www.dotnetbuilders.com/constructor.aspx.

26. Codagen. "Codagen Architect for MDA (http://www.codagen.com/)." Retrieved 07/2006, from http://www.codagen.com/.

27. Objecteering. "Objecteering/UML (http://www.objecteering.com/)." Retrieved 07/2006, from http://www.objecteering.com/.

28. Aonix. "Ameos toolset for MDA (http://www.aonix.com/ameos.html)." Retrieved 08/2006, from http://www.aonix.com/ameos.html.

29. Borland. "Together Architect (http://www.borland.com/us/products/together/index.html)." Retrieved 07/2006, from http://www.borland.com/us/products/together/index.html.

30. Xactium. "XMF Mosaic (http://albini.xactium.com/web/index.php?option=com_content&task=blogcategory&id=27&Itemid=46)." Retrieved 07/2006, from http://albini.xactium.com/web/index.php?option=com_content&task=blogcategory&id=27&Itemid=46.

31. Boocock, P. "The Jamda project (http://jamda.sourceforge.net/)." Retrieved 08/2006, from http://jamda.sourceforge.net/.

32. PathFinderSolutions. "PathMate MDA transformation environment (http://www.pathfindermda.com/products/index.php)." Retrieved 07/2006, from http://www.pathfindermda.com/products/index.php.

33. NetBeans. "NetBeans (http://www.netbeans.org/; http://www.netbeans.org/about/press/8.html)." Retrieved 07/2006.

34. IBM. "Rational XDE Developer (http://www-306.ibm.com/software/uk/rational/awdtools/swdeveloper.html)." Retrieved 08/2006, from http://www-306.ibm.com/software/uk/rational/awdtools/swdeveloper.html.

35. Eclipse.Org. "Eclipse project (http://www.eclipse.org/)." Retrieved 08/2006.

36. IBM. "Websphere MDA tool (http://www-306.ibm.com/software/websphere/)." Retrieved 08/2006, from http://www-306.ibm.com/software/websphere/.

37. Scheer. "Aris Toolset (http://www.ids-scheer.com/)." Retrieved 7/2006, from http://www.ids-scheer.com/.

38. Tariq, N. A. and Akhter, N. 2005. Comparison of Model Driven Architecture(MDA) based Tools. Computer Science Department. Stockholm, Sweden, Royal Institute of Technology (KTH). Masters**:** 74.

39. SoSym. "MDA - Metrics used for tool comparison (http://www.sosym.co.uk/Sections/Projects/VIDE/projectDocs/mdaToolMetrics.html)" Retrieved 04/2007, from http://www.sosym.co.uk/Sections/Projects/VIDE/projectDocs/mdaToolMetrics.html.