Abstract

Transitioning to Model Driven Software Development

Preparing for the Paradigm Shift

Author: Jorn Bettin

Version 1.0

April 2006

SoftMetaWare

Revision History

Version	Author	Description
1.0	Jorn Bettin	April 2006

It is time for a major stock take of model driven software development approaches within software intensive industries. The speed of progress in the last few years in terms of interoperability standards for model driven tooling has not been spectacular. For example it took the OMG five years to develop an industry standard for expressing model transformations, and it remains to be seen if and when the OMG's Query, View, and Transformation (QVT) standard will become relevant. The term "Model Driven" has gone through the usual hype cycle, and the dust is in the process of settling. The outcome does not really come as a surprise - the assessment from the organizers at the OOPSLA'02 workshop "Generative Techniques in the Context of Model Driven Architecture" still holds:

> *The era of expensive software development tools is long gone. The market already discards MDA tool-hype, and does not readily buy into MDA. On the other hand, domain-specific approaches and generator technology have a huge potential. The main problem is not one of tools - as some vendors would like people to think - but one of culture. Unless this is recognized, MDA will become the CASE of this decade.*
> *…*
>
> *The only realistic way to gain widespread acceptance for the use of MDA-type approaches and generation tools is by addressing the related cultural issues, and by providing tools at a very low cost - so low that a CEO or CTO does not have to think twice. How about an open source-based project? While the tool vendors jockey for position and battle amongst themselves, an open source project may actually deliver results and become a de-facto MDA standard.*

Now, in 2006, there is still no market leading MDA tool vendor in sight, and in general the uptake of proprietary commercial tools has fallen significantly short of vendor expectations. In the mean time, Microsoft joined the MD* club in 2003 with their Software Factories initiative, and a large range of Open Source MD* projects have emerged and have had time to mature. For the most part decision-makers and software developers in the software industries have been watching from the sidelines. This is the result of lessons learnt from Computer Aided Software Engineering (CASE) tools and from the productivity drop in the 90s that followed the adoption of object oriented languages such as C++ and Java, and the hype about reuse.

This white paper analyzes the current situation from a tool vendor independent perspective, and it provides practical recommendations for the adoption of a Model Driven Software Development paradigm. The recommendations are consistent with SoftMetaWare's Industrialized Software Asset Development (ISAD) methodology, and are the result of guiding software development organizations in various industries through the paradigm shift to model driven approaches.

1 The Model Driven & Open Source Software Hype Curve

The following picture is a subjective - but possibly quite helpful - interpretation of the MD* adoption rate to date. The diagram intentionally superimposes MD* milestones and OSS milestones, so that it becomes clear how the MD* trend is able to surf on the wave of mainstream acceptance of OSS tools that currently rolls through software development organizations.

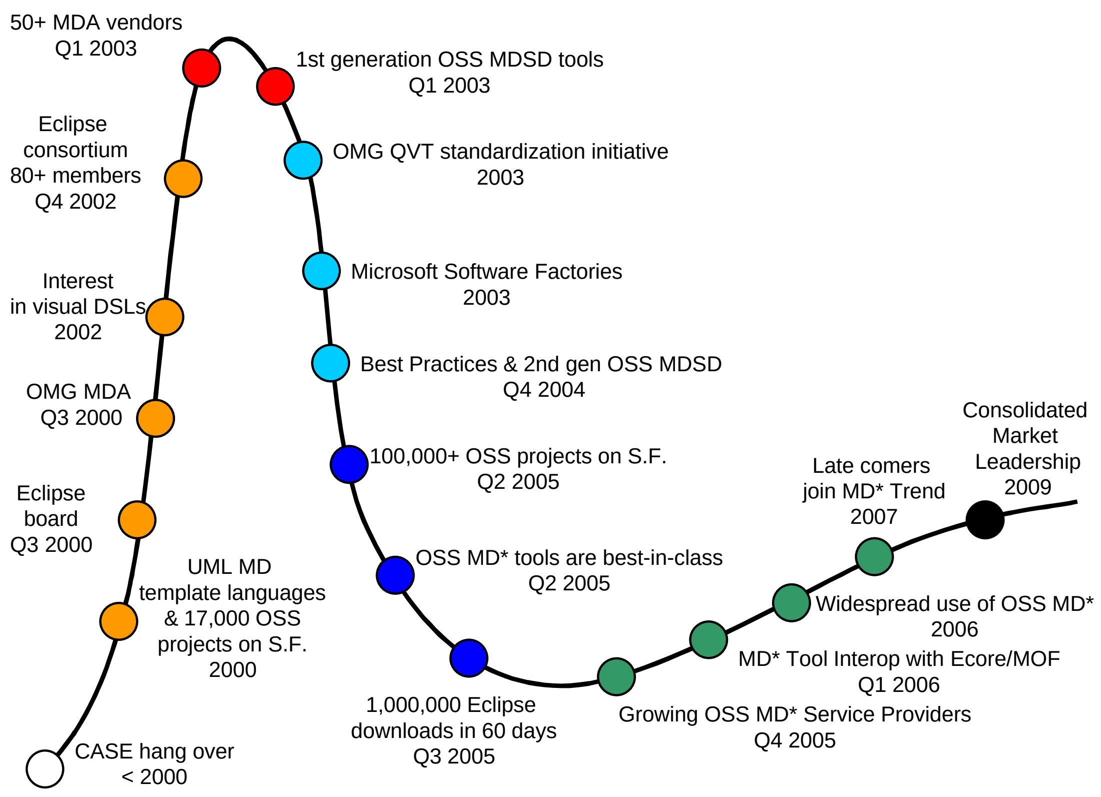
50+ MDA vendors Q1 2003
1st generation OSS MDSD tools Q1 2003
Eclipse consortium 80+ members Q4 2002
OMG QVT standardization initiative 2003
Microsoft Software Factories 2003
Interest in visual DSLs 2002
Best Practices & 2nd gen OSS MDSD Q4 2004
OMG MDA Q3 2000
100,000+ OSS projects on S.F. Q2 2005
Consolidated Market Leadership 2009
Late comers join MD* Trend 2007
Eclipse board Q3 2000
UML MD template languages & 17,000 OSS projects on S.F. 2000
OSS MD* tools are best-in-class Q2 2005
Widespread use of OSS MD* 2006
MD* Tool Interop with Ecore/MOF Q1 2006
1,000,000 Eclipse downloads in 60 days Q3 2005
Growing OSS MD* Service Providers Q4 2005
CASE hang over < 2000

Figure 1 *MD* & OSS Hype Curve*

Without a reasonable understanding of how OSS shapes the development of software infrastructure [Bettin 2005], including MD* software development tools, it is easy to miss the significance of the illustrated OSS milestones.

From the OMG's perspective, the CASE hang over seemed to be over by 2000, and hence the Model Driven Architecture [OMG MDA] initiative was launched. At that point the market already included several vendors of UML model driven template languages, but customers were few and far between. At the same time the Open Source phenomenon was steaming ahead at grass roots level, but was only poorly understood by most software industry executives and within end user organizations.

Undoubtedly, one of the best decisions in the last five years by IBM was the decision to Open Source the Eclipse Java IDE [Eclipse] platform. Firstly this forced IBM staff at all levels within the organization to get familiar with the Open Source phenomenon. Secondly, it provided a very strong signal to the market, and paved the way for incremental and gradual acceptance of OSS in conservative sectors such as banking and government.

A key element that simply does not fit on the timeline in figure 1 is the long track record of Software Product Line Engineering [SEI SPLE] approaches, which can be traced back to the mid 70s. Successful OSS MD* components such as OpenArchitectureWare [Eclipse GMT-OAW] mainly build on Software Product Line Engineering principles, and only to a lesser degree on emerging OMG MDA standards.

The one MDA standard that is really worth taking note of is the Meta Object Facility [OMG MOF], as this standard can be used as a the foundation for defining domain specific languages (DSLs). See section 2 for a detailed discussion of the relevance of DSLs. Since it is a result of the OMG consortium, MOF suffers from design by committee, and it contains a number of elements that are irrelevant in practice. The good news is that the Eclipse Modeling Facility OSS project includes Ecore, a pragmatic, simplified implementation along the lines of MOF. The practical applicability of Ecore is outlined in section 5.

It took the OMG until 2003 to launch an initiative to develop a standard for model transformations (Query, Views, Transformations or QVT). Without such a standard today's MDA vendors make use of various proprietary languages to express model transformations. Hence it still is somewhat premature to talk about MDA as an established "standard". The initial version of the QVT standard [OMG QVT] is in the process of being published in 2006.

Around the time when the OMG started to think about QVT, Microsoft joined the model driven bandwagon - or rather officially opted out of the QVT standard - by announcing a "Software Factories" strategy. The goal of Software Factories is the development of tools that enable software developers to define and subsequently use DSLs. See section 6 for a discussion of Software Factories and Microsoft's motivation.

2 Domain Specific Languages

The majority of software systems today are implemented in general-purpose languages that address a broad domain. Classical programming languages for business software development such as COBOL or RPG survive alongside modern object oriented languages such as Java or C#. Additionally, modeling languages such as UML provide a formal visual syntax to describe the structural aspect and part of the behavioral aspect of object oriented software systems.

In contrast, all advanced applications of model driven approaches involve domain specific languages (DSLs), i.e. formal specification languages that incorporate domain concepts as first class language elements. Familiar DSLs are end user programming tools such as spreadsheets and symbolic mathematics systems.

The limitations of the predominant one-fits-all philosophy became obvious during the attempts to replace languages such as COBOL with a new generation of general purpose tools for business software development that were marketed in the 80s and 90s under the heading of Computer Aided Software Engineering (CASE) tools. Although some of the better CASE tools definitely raised the level of abstraction - and thereby productivity - their output was always tied to a specific technology stack, which ran counter the trend towards more and more distributed and heterogeneous