University of Isfahan
Faculty of Computer Engineering
Department of Software Engineering
Model-Driven Software Engineering (MDSE) Research Group

Workshop on Model-Based Design, Test, and Verification of Software

July 20 & 21, 2019

http://mdse.ui.ac.ir/
Model-Driven Software Engineering (MDSE)

Concepts & Principles
Adoption in industry

By: Dr. Bahman Zamani
zamani@eng.ui.ac.ir
History of MDE

* Roots:
  * Modeling in traditional software engineering
* Lessons learned:
  * 1980’s Computer-Aided Software Engineering (CASE) Tools
  * 1997 UML standardization by OMG
  * 2003 Selic’s paper “Pragmatics of Model Driven Development”
  * 2003 Sendall and Kozaczynski’s paper “Model transformation: the heart and soul of model-driven software development”
  * 2006 Schmidt’s paper “Model Driven Engineering”
  * 2006 Jean Bézivin’s paper “Model Driven Engineering: An Emerging Technical Space”
What is a model?

- A model is a representation of a system
  - that hides some of the properties and highlights others
- A model is an abstraction
Model Forms

* Mathematical, eg Linear Programming
* Physical, eg Bridge
* Diagrammatic, eg Use case model

\[
\begin{align*}
\text{minimize} & \quad \sum_{j=1}^{n} c_j x_j \\
\text{subject to} & \quad \sum_{i=1}^{m} a_{ij} x_j \leq b_i \quad (i = 1, 2, \ldots, m) \\
& \quad x_j \geq 0 \quad (j = 1, 2, \ldots, n)
\end{align*}
\]

http://www.cise.ufl.edu/~davis/Morgan/Image13.gif

Despite the processes that are code-centric, in these approaches, models are the main artifacts which drive the development → Model-Driven

The ultimate goal is to automatically generate programs from the corresponding models
A metaphor for MDE

http://www.theenterprisearchitect.eu/archive/2009/08/05/a-metaphor-for-model-driven-engineering
**Modes of transformation**
- Manual
- Automatic

**Types of transformation**
- Model-to-Model
- Model-to-Code = Forward Engineering
- Code-to-Model = Backward Engineering
- Code-to-Code
Technology hype cycle

Technology adoption life cycle

**MD* JUNGLE**

* MDE: Model-Driven Engineering
  * MDSE: Model-Driven Software Engineering
* MDD: Model-Driven Development
  * MDSD: Model-Driven Software Development
* MDA: Model-Driven Architecture

Overview of the MDSE methodology

Four layers of meta-ness

Modelware vs. Grammarware

Scope of MDSE code-generation

Model-Driven Development Challenges and Solutions
Experiences with Domain-Specific Modelling in Industry

Juha-Pekka Tolvanen and Steven Kelly
MetaCase, Ylistömäentie 31, FI–40500 Jyväskylä, Finland
{jpt, stevek}@metacase.com

MODELSWARD 2016
4th International Conference on Model-Driven Engineering and Software Development
Rome, Italy 19 - 21 February, 2016
Where to use MDD?

* MDD is less applied in project-based development
  * Eg, consulting, outsourced work, or IT as an internal support function
* MDD is common in product development
  * Eg, automotive, telecom, banking
We focus on modeling that is based on domain specific languages.

Experience has shown us that DSM enables better MDD than general-purpose modeling languages.

The benefits of DSM do not come for free

- The language abstractions and tools to automate development need to be first developed and later maintained.

Our experiences are partly reported in cases over the last 25 years.

Unsurprisingly, most of our experience is with MetaEdit+.
Productivity by DSM & MetaEdit+

Figure 2: Productivity gains reported with DSM.

Model-Driven Development Challenges and Solutions

Experiences with Domain-Specific Modelling in Industry
DSM development effort

![Diagram showing days to define modelling languages and code generators along with tooling support in MetaEdit+](image)

Figure 3: Days to define modelling languages and code generators along with tooling support in MetaEdit+

Model-Driven Development Challenges and Solutions
Experiences with Domain-Specific Modelling in Industry
Metaedit+ vs. GMF

Figure 4: Effort to define the same BPMN language with different tools.

Model-Driven Development Challenges and Solutions
Experiences with Domain-Specific Modelling in Industry
Empirical assessment of MDE in industry

Authors:
- John Hutchinson
  Lancaster University, Lancaster, United Kingdom
- Jon Whittle
  Lancaster University, Lancaster, United Kingdom
- Mark Rouncefield
  Lancaster University, Lancaster, United Kingdom
- Steinar Kristoffersen
  Østfold University College and Møre forskning Molde AS, Halden, Norway

Published in:
Proceedings of the 33rd International Conference on Software Engineering
ICSE '11
Waikiki, Honolulu, HI, USA — May 21 - 28, 2011
ACM New York, NY, USA ©2011
ISBN: 978-1-4503-0445-0
doi:10.1145/1985793.1985858

Bibliometrics
- Citation Count: 60
- Downloads (cumulative): 1,277
- Downloads (12 Months): 120
- Downloads (6 Weeks): 7

Get this Article
Results of a 12 month empirical study
Finding factors of how and why companies succeed or fail with MDE
“The study is also deliberately agnostic with respect to the usefulness of MDE; that is, it is as much interested in MDE failure as it is in success.”
Success/failure more depends on social than technical issues
### Table 1: Illustrative influences of MDE.

<table>
<thead>
<tr>
<th>Impact Factor</th>
<th>Positive Influences</th>
<th>Negative Influences</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Productivity</strong></td>
<td><em>Time to develop code</em>&lt;br&gt;Reduced by: automatic code generation.</td>
<td><em>Increased</em> by: time to develop computer-readable models; implement model transformations, etc.</td>
</tr>
<tr>
<td></td>
<td><em>Time to test code</em>&lt;br&gt;Reduced by: fewer silly mistakes in generated code; model-based testing methods, etc.</td>
<td><em>Increased</em> by: effort needed to test model transformations and validate models, etc.</td>
</tr>
<tr>
<td></td>
<td><em>ROI on modeling effort</em>&lt;br&gt;Positive influences of modeling: more creative solutions; developers see the “bigger picture”.</td>
<td><em>Negative</em> influences of modeling: “model paralysis”; distracting influence of models.</td>
</tr>
<tr>
<td><strong>Portability</strong></td>
<td><em>Time to migrate to a new platform</em>&lt;br&gt;Reduced by: simply applying a new set of transformations.</td>
<td><em>Increased</em> by: effort required to develop new transformations or customize existing ones.</td>
</tr>
<tr>
<td><strong>Maintenance</strong></td>
<td><em>Time for stakeholders to understand each other</em>&lt;br&gt;Reduced since: easier for new staff to understand existing systems; code is “self-documenting”.</td>
<td><em>Increased</em> since: generated code may be difficult to understand.</td>
</tr>
<tr>
<td></td>
<td><em>Time needed to maintain software</em>&lt;br&gt;Reduced since: maintenance done at the modeling level; traceability links automatically generated.</td>
<td><em>Increased</em> since: need to keep models/code in sync, etc.</td>
</tr>
</tbody>
</table>
Table 3. The impact of MDE activities on productivity and maintainability.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Productivity</th>
<th>Maintainability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Increased</td>
<td>Not Used</td>
</tr>
<tr>
<td>Use of models for team communication</td>
<td>73.7%</td>
<td>7.0%</td>
</tr>
<tr>
<td>Use of models for understanding a problem at an abstract level</td>
<td>73.4%</td>
<td>4.8%</td>
</tr>
<tr>
<td>Use of models to capture and document designs</td>
<td>65.0%</td>
<td>9.3%</td>
</tr>
<tr>
<td>Use of domain-specific languages (DSLs)</td>
<td>47.5%</td>
<td>32.6%</td>
</tr>
<tr>
<td>Use of model-to-model transformations</td>
<td>50.8%</td>
<td>24.6%</td>
</tr>
<tr>
<td>Use of models in testing</td>
<td>37.8%</td>
<td>33.9%</td>
</tr>
<tr>
<td>Code generation</td>
<td>67.8%</td>
<td>12.0%</td>
</tr>
<tr>
<td>Model simulation/Executable models</td>
<td>41.7%</td>
<td>38.3%</td>
</tr>
</tbody>
</table>
MDE Gains

Figure 2 (a) “Is your use of code generation an important aspect of your MDE productivity gains?” (b) “Is integrating generated code into your existing projects a significant problem?”

Figure 3 (a) “Do organizations adopt MDE for its technical merits?” (b) “Do organizations adopt MDE to "jump through hoops" or appear to do so?”
Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure

John Hutchinson *, Jon Whittle, Mark Rouncefield

School of Computing and Communications, Lancaster University, UK

HIGHLIGHTS

• We present extensive results from a survey of MDE practices in industry.
• We present case studies of the adoption of model driven engineering (MDE) by four companies.
• We identify important factors that can affect the success or failure of MDE use from both the survey and case studies.
• MDE provides genuine benefits to those companies who use its appropriate contexts.
• Success/failure appears to be more dependent on organizational factors than technical.
Thank You!

Model-Driven Software Engineering Research Group

http://mdse.ui.ac.ir/