MDE 2.0 – Pragmatic formal model verification and other stories

Jordi Cabot

HdR
Index

- Introducing MDE
  - Research in MDE
    - A Research Agenda for MDE 2.0
      - Models & Quality
      - Legacy systems
      - Social aspects
      - Very Large models
  - Dissemination and technology transfer
- Credits
- Conclusions
Software is everywhere
but software development is still a challenge

Challenged = late, overbudget, and/or with less than the required features and functions

Successful = delivered on time, on budget, with required features and functions

Failed = cancelled prior to completion or delivered and never used
Model-driven Engineering

- **MDE** tries to improve this situation by promoting engineering principles in Software Engineering.

- MDE advocates the **rigorous use of software models** as the main artifacts in all **software engineering** activities.

- This in fact is common practice in many other professions.

- Does anybody imagine building a house without plans?
The MDEequation
Many MDE applications

Code Generation

Software Modernization

Systems interoperability
MDE = Model-Driven Everything

MDSD (software development)

MDE (engineering)
What is MDE?

- UML
- Profile
- OCL
- OMG
- Ecore
- Code-Generation
- Marte
- QVT
- Metamodel
- TGG
- Model Evolution
- Metamodel
- EMF
- MOF
- DSLs
- SBVR
- Model-driven Rev. Eng
- Profiles
- MDD
- Model-based Testing
- Multi-modeling
- ATL
- MDA
- M2M
- M2T
- Graph
- BPMN
- T2M
- Model Quality
There’s hope - Order in the Chaos

- Basic principle: *Everything is a model (J. Bézivin)*

- Models are designed/built/generated to be
  – Observed
  – Transformed

- The **MDE Equation:**
  \[ \text{Models} + \text{transformations} = \text{Software} \]

  where of course transformations can also be regarded as models

\[ \text{Models} + \text{transformation models} = \text{Software} \]

\[ \text{Models} + \text{Models} = \text{Software} \]

\[ 2 \times \text{Models} = \text{Software} \]

\[ \text{Models} = 1/2 \text{Software} \]
What is a model

City of Nantes = “system” to be modeled

A map is a model of this system

Its legend (what elements can appear, how they can be combined,...) is the grammar/metamodel

13/09/2012
A model is the result of the observation of a system considering a given metamodel with specific objectives or purposes.
Btw, same approach as other Technical Spaces

MDE
- MOF (metametamodel)
- UML (metamodel)
- ABank.uml

Grammarware
- EBNF.g
- Java.g
- MyProgram.java
MDE Core Technique/Operation: Model Transformation

- Model-to-Model Transformation (M2M)

- Java Metamodel
- ATL Metamodel
- UML2 Metamodel

Conforms to

Model a

Transformation
Java-to-UML2

Model b

© AtlanMod - atlanmod-contact@mines-nantes.fr
Index

- Introducing MDE
- **Research in MDE**
  - A Research Agenda for MDE 2.0
    - Models & Quality
    - Legacy systems
    - Social aspects
    - Very Large models
- Dissemination and technology transfer
- Credits
- Conclusions
Research in MDE
We have advanced a lot on the core techniques

- UML and profiles
- DSLs & Language workbenches
- Model-to-model and model-to-text transformations
- Model management and evolution
- ...
with some contributions: Conceptual Modeling

J Cabot, R Raventós: Conceptual Modelling Patterns for Roles. Journal on Data Semantics
with some contributions: Rule modeling

Table 2
Equivalences for collection operators

<table>
<thead>
<tr>
<th>Condition 1</th>
<th>Condition 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-&gt;includes(o) ⇐ X-&gt;count(o)&gt;0</td>
<td>X-&gt;excludes(o) ⇐ X-&gt;count(o)&lt;1</td>
</tr>
<tr>
<td>X-&gt;includesAll(Y)</td>
<td>X-&gt;excludesAll(Y) ⇐ X-&gt;count(o)&lt;1</td>
</tr>
<tr>
<td>Y-&gt;forAll(y1</td>
<td>X-&gt;includes(y1)</td>
</tr>
<tr>
<td>X-&gt;isNotEmpty() ⇐ X-&gt;size()=0</td>
<td>X-&gt;notEmpty() ⇐ X-&gt;size()&gt;0</td>
</tr>
<tr>
<td>not X-&gt;isNotEmpty() ⇐ X-&gt;notEmpty()</td>
<td>not X-&gt;notEmpty() ⇐ X-&gt;size()&gt;0</td>
</tr>
<tr>
<td>X-&gt;excluding(o) ⇐ X-&gt;¬(Set{o})</td>
<td>X-&gt;including(o) ⇐ X-&gt;union(Set{o})</td>
</tr>
<tr>
<td>X-&gt;size()&lt;=0 ⇐ X-&gt;size()==0</td>
<td>not X-&gt;size()==0 ⇐ X-&gt;size()&gt;0</td>
</tr>
<tr>
<td>X-&gt;last() ⇐ X-&gt;at(X-&gt;size())</td>
<td>X-&gt;first() ⇐ X-&gt;at(1)</td>
</tr>
<tr>
<td>X-&gt;union(Y),r1...rn -&gt;forall(z1</td>
<td>X)</td>
</tr>
</tbody>
</table>

Table 3
Equivalences for iterator expressions

<table>
<thead>
<tr>
<th>Condition 1</th>
<th>Condition 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-&gt;exists(Y) ⇐ not X-&gt;forall(not Y)</td>
<td>not X-&gt;exists(Y) ⇐ X-&gt;forall(not Y)</td>
</tr>
<tr>
<td>X-&gt;select(Y)-&gt;size()&gt;=0 ⇐ X-&gt;forall(not Y)</td>
<td>X-&gt;select(Y)-&gt;size()&gt;=0 ⇐ X-&gt;forall(not Y)</td>
</tr>
<tr>
<td>X-&gt;forall(Y implies Z)</td>
<td>X-&gt;select(Y)-&gt;exists(Z) ⇐ X-&gt;forall(Y implies Z)</td>
</tr>
<tr>
<td>X-&gt;reject(Y) ⇐ X-&gt;select(not Y)</td>
<td>X-&gt;any(Y) ⇐ X-&gt;select(Y)-&gt;asSequence()-&gt;first()</td>
</tr>
<tr>
<td>X-&gt;isUnique(Y) ⇐ X-&gt;forall(x1,x2</td>
<td>x1&lt;x2 implies x1,Y&lt;x2,Y)</td>
</tr>
<tr>
<td>X-&gt;select(Y)-&gt;size()==X-&gt;size() ⇐ X-&gt;forall(Y)</td>
<td>X-&gt;forall(Y)</td>
</tr>
</tbody>
</table>
with some contributions: Code Generation


with some personal contributions: Code Generation

with some contributions: UML Validation

with some contributions: API integration

J Cánovas, F Jouault, J Cabot, J García Molina. API2MoL: Automating the building of bridges between APIs and Model-Driven Engineering. Information and Software Technology.
But it’s clearly not enough

- Modeling will be commonplace in 3 years time – S. Mellor
  Though he is giving the same answer for the last 20 years
Index

- Introducing MDE
- Research in MDE
- **A Research Agenda for MDE 2.0**
  - Models & Quality
  - Legacy systems
  - Social aspects
  - Very Large models
- Dissemination and technology transfer
- Credits
- Conclusions
What else do we need?
MDE 2.0
Four main challenges

1. Quality of models
2. Support for legacy systems
3. Social aspects of MDE
4. Very large models / Scalability
Quality

- Quality is a very broad concept

We focus on the verification of models (are we building the models right?) and, partially, on their validation (are we building the right models?)
Importance of Quality in models

Errors in models will lead to errors in the resulting software
A Basic quality property: Satisfiability

- **Satisfiability** is the most basic correctness property for static models. Liveliness, redundancy,... can be expressed in terms of this one.
- A model is satisfiable if it is possible to create a **valid instantiation** of that model. Otherwise it is useless, users won’t be able to work with the model.
- A instantiation is valid if it satisfies all model constraints.

- A model is satisfiable if it is possible to create a valid instantiation of that model. Otherwise it is useless, users won’t be able to work with the model.
- A instantiation is valid if it satisfies all model constraints.

### Example

**Employee**
- name: String

**Dept**
- id: String

**Boss**
- name = “Timothy”

**Manager**

1. WorksIn
   - workers
   - dept
   - manager
   - managed

1. WorksIn
   - e1: Employee
     - name = “Peter”
   - d1: Dept
     - id = “HR”

1. WorksIn
   - e2: Employee
     - name = “Timothy”
Example: Is it satisfiable?

+ constraint: Nobody can be his own ancestor
How models are verified?

Strong Satisfiability
But Quality classified as a Grand Challenge

1E+18 scenarios for small models (classes=10)

More than cells in the human body!!!
Finding the right trade-off

- No perfect solution exists

Automation

Efficiency

Expressiveness

Complete

Verification
Our “pragmatic” approach

EMF/UML model
1. Class diagram / metamodel
2. OCL constraints

Constraint Satisfaction Problem
1. Variables - basic types + struct/list
2. Domains - finite
3. Constraints - Prolog
4. Property -> Additional Constraint

Translate → Deduce

Property?

Solve
Solution?
Translation of OCL invariants

context Paper inv: self.wordCount < 10000

- OCL invariant = instance of OCL metamodel
- Invariant becomes a constraint of the CSP
Translation of OCL invariants

```
LessThan(Vars, Result):-
    // Evaluate subexpressions
    Attribute(Vars, X),
    Constant(Y),
    // Compute result
    #<(X,Y, Result).
```

Diagram:
- `OperationCallExp` with `Operation` as `<`
- `AttribCallExp` for `wordCount`
- `IntLiteralExp` with value `10000`
- `VariableExp` for `self`
attribute(Vars, X) :-
  // Evaluate subexpressions
  self(Vars, Z),
  arg(Z, "wordCount", X).

self(Vars, Z) :-
  // Z is the only visible var
  nth1(1, Vars, Z).
Translation of OCL invariants

context Paper inv: self.wordCount < 10000

invariant(Papers):-
// Expression must be true
// for each Paper
( for (Paper, Papers)
do
  LessThan([Paper],Result),
  Result #= 1 ) .
OCL Prolog library

- To analyze OCL constraints, it is necessary to provide a translation for all OCL constructs in terms of the Prolog-based language used by the ECLiPSe solver.

- We have extended ECLiPSe with a new library dedicated to the translation of OCL constraints.

- The library makes use of the ECLiPSe support for higher-order predicates:
  
  e.g. `ocl_set_collect(Instances, Vars, Set, Predicate, Result),`

- Removal of symmetries and the suspension mechanism are used to optimize the performance of the library.
Resolution of the CSP

Define cardinality variables

Constraints on cardinalities

Assign cardinalities

Define attribute variables

Constraints on attributes

Assign attributes

Proof
Trade-offs in verification

- **Decidability**: is automation possible?  Yes

- **Completeness**: proof for any input?  No, bounded verification (but Small Scope hypothesis – D. Jackson)

- **Expressiveness**: full OCL support?  Yes

- **Efficiency**: Controlled by the user

- **Validation**: is also possible: we can generate valid instances from partial models
Other applications (1): Operation contracts

- Verification of contracts: Applicability and executability of operations, determinism,...
Other applications(2): Graph Transformation to OCL

context System::work()
pre: Machine::allInstances()→exists(m)
   not ( Operator::allInstances()→exists(op1|m.operator→includes(op1)))
post: Machine::allInstances()@pre→exists(m| not
   ( Operator::allInstances@pre()→exists(op1|m.operator->includes(p1)))
   and op2.oclIsNew() and op2.oclIsTypeOf(Operator) and m.operator→includes(op²)

Other applications(3): TGG / QVT / ATL to OCL

**Invariants:**

**context Cl2Tb Inv:**
self.class.size()=1 and self.table.size()=1 and self.class.name=self.table.name

**context Class Inv:**
self.persistent=true implies self.Cl2Tb.size() >=1

**context Table Inv:**
self.Cl2Tb.size() >=1

---

Cabot, Clariso, Guerra, de Lara: Verification and validation of declarative model-to-model transformations through invariants. Journal of Systems and Software
Challenges / Work in Progress

- Incremental verification
- Combine CSPs with SMT solvers to provide a complete verification approach when possible.
- More meaningful feedback (e.g. the subset of constraints that cause the inconsistency)
- Model slicing for parallel verification
Model-driven Reverse Engineering
MoDISCO: a MDRE framework

Legacy Systems

- Source code
- Databases
- Configuration files

Modernization helpers

- Documentation
- Impact analysis

- Models
- Viewpoints

- Restructured code
- Migrated code

- Metrics
- Norms checking

- Instead of adhoc Rev. Eng. Solutions, we use an intermediate model-based representation of the legacy system
MDRE phases in MoDisco

---

**Legacy artifacts:**
- source code
- configuration files
- tests
- databases
- etc.

**Why?** Models provide an homogeneous and interrelated representation of all legacy components.

No information loss: initial models have a 1:1 correspondence with the code.
Challenges / Work in Progress

- Extraction of business rules
  - Getting a model of the code helps but it’s still too low-level for a stakeholder to look at it
  - Semi-automatic approach with IBM.

- Rev Eng of the whole software system
  - We know how to extract a model from a single component but we don’t take into account its relationship with other components (esp. in other layers)
  - Companies are interested in knowing the enterprise architecture (e.g. TOGAF) of a system
Challenges / Work in Progress

- Reverse Engineering of security policies
Social aspects of MDE
Social aspects of MDE

- We need a better understanding of the needs of users (technical and non-technical) to make sure we solve their actual problems (and not the ones we think they have).

- One example: Huge amount of research on providing methods for the specification, validation, etc of non-functional requirements:
  - Nontechnical constraints (like cost, type of license, specific providers) are as prominent as technical requirements like performance or security
  - Modern technology platforms already cover many applications’ quality requirements so explicit NFR management not so useful
  - NFRs are hardly ever documented and poorly validated

Architecture Quality Revisited. Buschmann, Frank; Ameller, David; Ayala, Claudia P.; Cabot, Jordi; Franch, Xavier. IEEE Software, vol. 29 (4)
Dealing with users is not easy

How users see the programmers

How programmers see the users

Get cheered up on 9GAG.COM
Collaborative development of DSLs

- Participation of end-users is specially relevant when creating DSLs since we are creating a language for them.
Collaborative development of DSLs

Initial version

1. Route \( p_1 \) follows \( p_2 \) = Truck

Collaboration process

2. Change Proposal
   - The language should be able to represent the existing traffic or the routes to facilitate the decision of the best route to follow.
   - Agreement: Developer 1, Developer 2
   - Accepted

3. Solution
   - The Route concept should be enriched to incorporate a new String attribute describing the traffic.
   - Agreement: End-User 1, Developer 2
   - Rejected

4. Comment
   - This attribute should be represented as an enumerated value.
   - Agreement: End-User 1, Developer 3
   - Accepted

5. Solution
   - The Route concept should be enriched to incorporate a new attribute describing the traffic. The possible values of this attribute are defined by the Enumerated value
   - Agreement: End-User 1, Developer 2
   - Accepted

6. New version
   - TrafficType
     - LIGHT
     - NORMAL
     - DENSE
   - Route
     - Traffic : TrafficType
     - follows
     - Truck
Collaborative development of DSLs

- **Collaboro**: process + DSL + tool to enable the collaborative development of DSLs

- Users suggest changes to both abstract and concrete syntax levels

- The community comments and votes changes and solutions

- Once an agreement is reached (based on a given decision policy, e.g. unanimity) the solution is added to the current language version

- We get:
  - Languages that better satisfy the users’ needs
  - Traceability to justify the rational behind the language design decisions
Challenges / Work in progress

- Gamification techniques to promote more user interaction

- What constitutes a good concrete syntax for DSLs?

- Learning from web designers: AB testing for DSLs
  - E.g. evolve the concrete syntax based on which alternative gets more “conversions”).
Very large models
MDE tools fail when dealing with very large models
- E.g. reverse engineering of the Eclipse platform generates a model of more than 5M instances
- EMF just crashes with these volumes

Scalability important both at the model (loading very large models) and model manipulation level (executing complex transformations on large models)

Key problem in industrial scenarios but far from a trivial one
- Very Large DataBases is the main conf. in the database domain with 37 editions and still looking for solutions
Modeling in the cloud

- Can the Cloud be used to handle VLMs?

- Two key aspects:
  - Model storage in the Cloud for efficiently storing and loading VLMs
  - Model transformation in the Cloud for distributing the computation of the transformation
    - Each virtual node executes the full transformation on a subset of the model
    - Each virtual node executes part of the transformation on the full model
Cloud Virtual Model

Model-Driven Tool

EMF Model Interfaces (Resource)

XMI Model (XMIResource)  DB Model (CDOResource)  Cloud Virtual Model (CloudResource)

FileSystem  Database  Cloud

accesses  implements  implements  accesses
Cloud Virtual Model

Distributed Model

Partial Model 1

Partial Model 3

Partial Model 2

Cloud Virtual Model

Cloud

Local
Challenges / Work in Progress

- Optimal way to split the models and transformations
  - And when not to split them

- Best technology for the backend?
  - Hadoop for storing the models plus MapReduce for the transformations?

- Reactive transformation engine:
  - Automatically activate only the strictly needed computation in response to updates or requests of model elements.
  - Incremental, minimal set of recalculations
Index

- Introducing MDE
- Research in MDE
  - A Research Agenda for MDE 2.0
    - Models & Quality
    - Legacy systems
    - Social aspects
    - Very Large models
- Dissemination and technology transfer
- Credits
- Conclusions
Always with technology transfer in mind
How do we bridge the gap?

- Traditional direct approach

Direct technology transfer doesn’t work!: Our tools are not good and they won’t magically become great
We need a new business model

- Three-entity approach

- **Introduction of a third entity in the process**
  - A SME as Technology Provider
  - Play the role of Interface between researchers and users
Dissemination
The Modeling Languages portal

- Dissemination of MDE news, tutorials, tools,… for people:
  - Researchers and practitioners
  - Novices or experts
- Goal: to explore and discuss when, where, and how MDE can improve current software engineering practices

Welcome to this MODELING Lenguajes Portal – All you wanted to know about software modeling and model-driven engineering

Modeling is supposed to be one of the most important activities in any software development process. At least this is the general understanding within the software engineering research community. However, in the day-to-day practice, modeling is usually regarded as, basically, a waste of time. ... More

Gartner’s view of MDA position in the hype cycle (I disagree)

According to this InfoQ article, Gartner’s believes that the technology “Model Driven Architecture” is still Sliding into the Trough viewed from the perspective of the hype cycle. I beg to disagree. Instead I agree with Stephen J. Mellor that believes MDA is progressing through the Slope of enlightenment. I think MDE hit rock bottom a couple of years ago. Read More

MDE making its way into stackoverflow

I've been in StackOverflow for a while now (to be precise, and according to my user profile, there are 3 years and 4 months) and in the last months I've started to see MDE-related questions in the site. UML has always been a popular topic (with almost 1000 questions)
Model-Driven Software Engineering: MDE in Practice

Marco Brambilla, Politecnico di Milano
Jordi Cabot, Ecole des Mines de Nantes
Manuel Wimmer, Vienna University of Technology

Foreword by Richard M. Soley, OMG

Series: Synthesis Lectures on Software Engineering

This book discusses Model Driven Engineering (MDE), which is the use of model-based approaches to improve the daily practice of software professionals. MDE practices have proved to increase efficiency and effectiveness in software development, as demonstrated by various quantitative and qualitative studies. MDE adoption in the software industry is foreseen to grow exponentially in the near future, e.g. due to the convergence of software development and business analysis.

The aim of this book is to provide the reader with an easy and
Credits
Credits

- Research is always a team activity
  - Works on conceptual modeling and code-generation together with colleagues from Technical University of Catalonia and Politecnico di Milano
  - Quality line started together with R. Clariso (Open University of Catalonia) and J. de Lara and E. Guerra (Autonomous university of Madrid)
  - Univ. of Toronto taught me the importance of social and organizational aspects of Software Engineering
  - ... thanks also to and many more co-authors that I can’t list here

- Current research lines possible thanks to the members of the AtlanMod team (EMN / INRIA / LINA )

- AtlanMod was created in 2008 by Jean Bézivin. He is the “father” of MoDisco and of the technology transfer model we are following.
Wrapping up
Conclusions

- MDE is changing the way we build software
  - Though this “we” is still limited

- We have explored some of the (IMHO) promising research directions to improve MDE (and its adoption):
  - Quality, scalability, human aspects, legacy systems

- MDE as a means to an end → Better Software Engineering

- It is ok to renounce to perfect solutions in exchange of useful ones
  - The good-enough revolution (Wired – 09/2009)