Multi-Level Modeling with Melanee

Sara Sali

University of Antwerp

sara.sali@student.uantwerpen.be

Abstract

Multi-level modeling based on deep instantiation has attracted growing interest over the last few years. There is a big number of practical tools that support this approach such as MetaEdit+, GMF, EMFText, XText, MetaDepth, Spoofax, AtomPM, Microsoft Visual Studio DSL and they are divided in two major groups: graphical and textual language based. This report explains some of the basic features and the architecture of the deep modeling tool Melanie, or sometimes referred to as Melanee, MultiLevel Modeling and Ontology Engineering Environment.

Keywords: meta-modeling, melanee, melanie, deep modeling, multi-level, modeling

1. Introduction

Domain Specific modeling tool can be seen as divided in two major groups. One group supporting graphical domain specific modeling (DSL) such as: MetaEdit+, Poseidon for DSLs, Microsoft Visual Studio for DSLs , Atom3/Atompm; and the other group supporting textual domain specific modeling such as: XText, JetBrains MPS, Spoofax, MetaDepth, EMFText.

Melanie is a tool which supports both graphical and textual domain-specific modeling. Although there is no clear study case where this could be useful, in order to solve this problem a software engineer would have to create a mapping between the graphical and textual modeling frameworks. If they were based on the same infrastructure then you could consider it as a lucky case (an example would be Xtext and GMF both based on the

Email address: sara.sali@student.uantwerpen.be (Sara Sali)

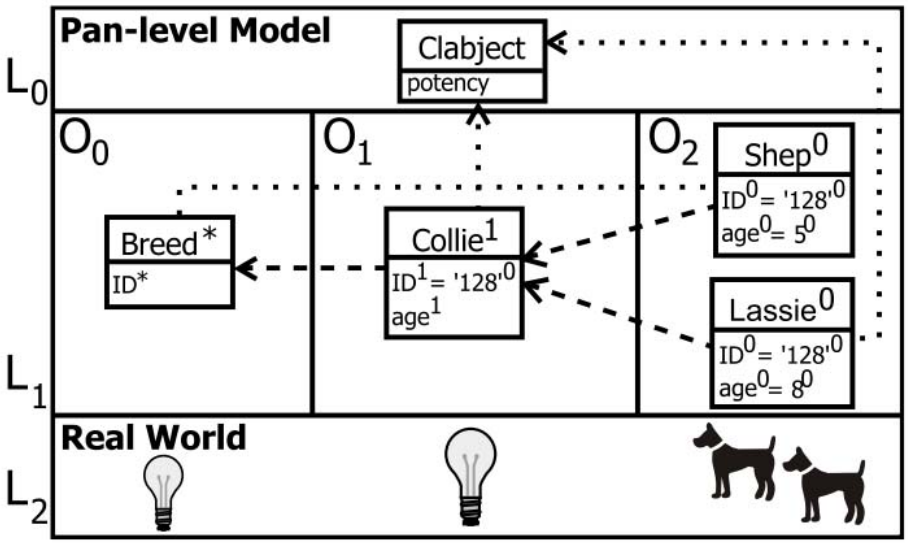
Eclipse Modeling Framework (EMF) and the Eclipse Platform). Creating the mapping would be easier compared to a more complex case where the tools used are based on different platforms. This would increase the software engineering process complexity and probably the cost of it.

In the next section of this paper the architecture, components and features of the tool are discussed. Following, a section describing briefly the Case Study to be modeled. After that a section of implementation and conclusion facts are discussed.

2. Melanie

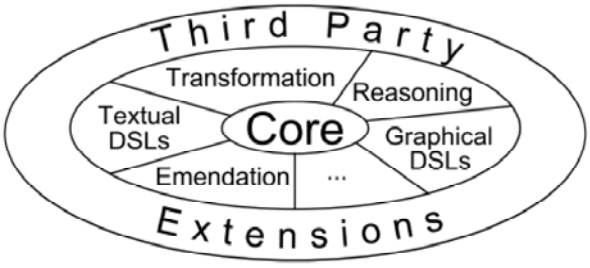
2.1. Melanie Components and Architecture

Multi Level Modeling in Melanie is made possible thanks to the orthoganal classification architecture, which separates the linguistic (horizontally dashed arrows) and ontological classifications (vertically dotted arrows) into two dimensions.



The Pan Level Model (PLM) in the picture above, defines the abstract language of any modeling language created in Melanie as an Ecore metamodel (EMF file). Modelers work on the L1 level, where they can create graphical concrete syntax for the PLM using the LML (Level agnostic Modeling Language).

The following picture shows Melanie and it's components:



The core is in the center and it's responsible for creation and storage in EMF and rendering in LML. The components in the above layer are responsible for Domain Specific Languages support, reasoning and transformation. Melanie's architecture provides symbiotic domain specific/general purpose languages using the concept of Visualizers (how a model element is visually represented). There are two kinds of Visualizers LML and DSL. The DSL Visualizers can be textual or graphical.

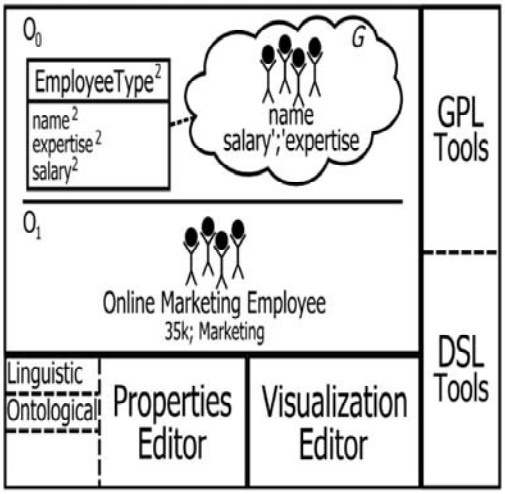
1. **Reasoning and checking** in Melanie is composed by ontological validation which checks correctness at different levels of granularity and ontology population services which checks if there is any inheritance connection between two elements in the model.
2. **Graphical DSL** to add Domain Specific Language Graphical representation. If this is not done then the LML (General Purpose Language) is used. The relationship between the GPL and DSL is called symbiotic since they help reinforce each others strengths and may be used at the same time.
3. **Textual DSL** allows the default textual rendering of multi-level models to be augmented with a custom notation. The textual and graphical renderings of models are fully interchangeable and useable simultaneously.
4. **The emendation module** continually monitors the ontology and as soon as a change occurs that effects more than the currently edited

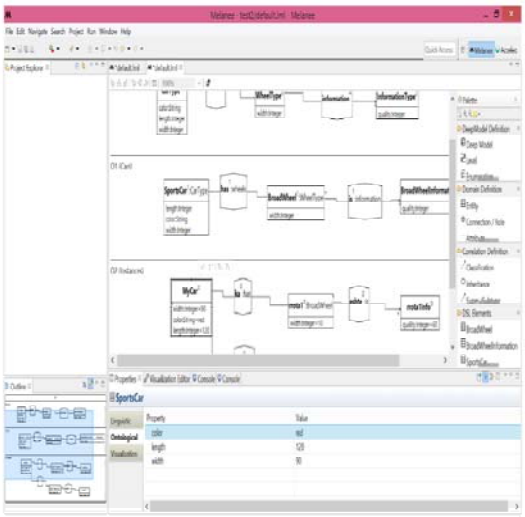
model element it suggests additional changes to keep the ontology consistent.

5. **Multi-level aware Transformation Support.** Melanie incorporates an ATL adapter that supports multi-level aware transformations.

Below is shown a mock up of the tool editor and a picture of the real tool.

Melanie





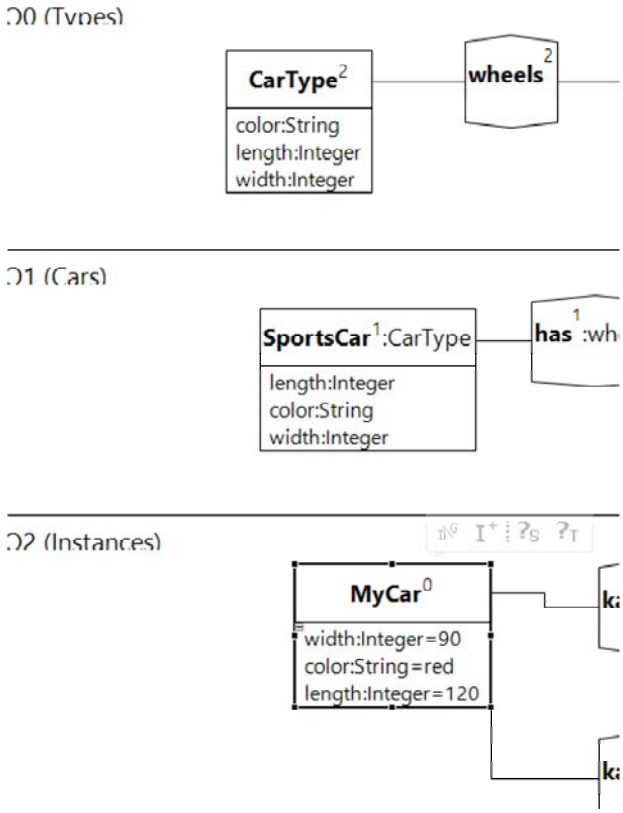
2.2. Melanie Features

In the paper "Rearchitecting the UML Infrastructure" of Thomas Kuhne and Colin Atkins (4) the new UML Infrastructure would have the features mentioned below. Next to each feature is written a short description of the implementation in Melanie.

- **Two or more Dimensional Modeling Framework.** (There are two dimensions: The physical, the dominant classification dimension from the view point of tool builders and the logical one from the viewpoint of the modelers.)

- **Unified Modeling Elements** (CLAss oBJECT (2)). The term clabject is introduced to reflect the type (class) / instance (object) duality of an ontological model-element. In the picture below the element

SportsCar in the middle ontological level is a claobject since it's an instance of the CarType element and a class for MyCar.



- **Deep Instantiation** (Potency) allows information to be carried over more than one instantiation level. In Melanie this is done by assigning a potency value to each element at the lowest ontological level (O0 in the example above). When the value of potency of CarType is set to two this means that instances of it can be created until the O2 level. This would not be the case if it was set to 1. Setting it to one means that at the ontological level O1 the SportsCar is an abstract class and as such we do not need to create instances of SportsCar in the O2 Level. Changing the potency at level O0 will automatically change the value in other levels.

  Moreover, Melanie implements the potency even to the attributes and their values with durability (how many instantiation steps an attribute

can be handed over to instances) and mutability (how many instantiation steps the value can be changed).

3. Case Study

4. Implementation in Melanie

5. Conclusions

6. References

[1] Melanie HomePage *www.melanee.org*

[2] Colin Atkinson, Ralph Gerbig *Harmonizing Textual and Graphical Visualizations of Domain Specic Models*. 2013.

[3] Colin Atkinson, Ralph Gerbig *Melanie Multi-level Modeling and Ontology Engineering Environment*.

[4] Colin Atkinson, Thoman Kuhne *Rearchitecting the UML Infrastructure*. 2002.