Multilevel Modeling with ConceptBase[⋆]

Manfred A. Jeusfeld[0000−0002−9421−8566]

University of Skövde, IIT, Sweden, manfred.jeusfeld@his.se
http://conceptbase.sourceforge.net/er21demo

Abstract. Multilevel modeling aims at improving the expressiveness and conciseness of conceptual modeling languages by allowing to express domain knowledge at higher abstractions levels. In this demonstration, we go thru two variants of multilevel extensions for the ConceptBase system, which had originally been used more for the design of domain-specific conceptual modeling languages. The demonstration highlights the partial evaluation feature of the deductive rule engine of ConceptBase. It also shows how multilevel modeling is essentially about a better understanding how instantiation, specialization, and attribution relate to each other in conceptual modeling.

Keywords: multilevel modeling · conceptual modeling · powertype · Telos.

1 Significance of Multilevel Modeling

The conceptual modeling field virtually started with the proposal of the Entity-Relationship Model [3]. Entity types and relationship types are simple classes. Its instances are entities and links, respectively. This is an example a two-level approach, as also promoted by the classical relational database theory, and logic. At about the same time, Abrial [1] presented his binary data model, which distinguished three abstraction levels: objects and links, classes and binary relation, and the binary data model itself. The multilevel modeling field started around 2000 with the works by Atkinson and Kühne [2], de Lara and Guerra [8] and others [9,4]. Multilevel modeling essentially utilizes more than a single abstraction level to describe the artefacts of a domain. By doing so, it avoids redundant definitions of attributes and relationships at multiple classes. For example, at the (metaclass) level of "ProductType" one can define an attribute "serialnumber" and designate this attribute to be used two abstraction levels below. Another key factor in multilevel modeling is to regard classes as regular objects ("clabjects") that have properties and links on their own right. Multilevel modeling is strongly rooted in classical conceptual modeling and has contributed to a better understanding of how to utilize abstraction levels beyond the language-driven metamodeling like promoted in UML.

⋆ Copyright © 2021 for this paper by its author. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

This demonstration is showcasing the multilevel modeling capabilities of ConceptBase [5,6]. ConceptBase is a mature conceptual modeling tool based on deductive database technology with a customizable graphical user interface. In the rest of this paper, we shortly introduce to the ConceptBase system and its underlying data model Telos [7]. We demonstrate two different approaches to multilevel modeling that have been implemented between 2014 and 2020 with ConceptBase. Dual-deep instantiation extends the so-called potency based approaches by source potencies. MLT-Telos is a powertype-based approach implementing a subset of the multilevel modeling theory MLT*. All demonstration parts are available via the web page http://conceptbase.sourceforge.net/er21demo and can be replayed by the interested reader with minimal installation effort. The main purpose of this demonstration is to explain how the deductive rule engine of ConceptBase realizes a rather elegant implementation of different approaches to multilevel modeling. The first one is potency-based (assigning level numbers to attributes and relations) and the other is powertype-based.

2 ConceptBase for Conceptual Modeling

ConceptBase was originally developed as a repository for software engineering artefacts. It is technically a deductive database with its own Datalog-neg engine to evaluate recursive deductive rules. The data model is a variant of the Telos language, which itself is based on the three principles instantiation, specialization, and attribution (subsuming relations between objects). Telos has a potentially unlimited instantiation hierarchy (individuals, classes, metaclasses, metametaclasses, etc.) terminated by a so-called omega-level class "Proposition" (standing for all objects), which has any explicit information as instance. This includes explicit attributes, instantiations, and specializations, which all are objects in Telos. ConceptBase also features a customizable graphical user interface CBGraph. It displays views on the database (i.e. models) by representing nodes and links by graphical shapes depending on (derived) properties of the objects. For example, a deductive rule can specify to display instances of the class "RelationshipType" with a diamond shape. The textual editor CBIva allows to define new objects in a frame syntax and query the ConceptBase database.

Traditionally, ConceptBase was mainly used for metamodeling, in particular to define families of interrelated domain-specific modeling languages. Deductive rules and integrity constraints are used to specify well-formed models and the semantics of modeling constraints. For example, the semantics of cardinality constraints can be expressed by integrity constraints (a special form of deductive rules). The multilevel modeling case is a particular challenge for the deductive rule engine of ConceptBase, since it has to cope with objects at more than two abstraction levels. Further, multilevel modeling has an intricate relation to the core principles of conceptual modeling, i.e. instantiation, specialization and attribution. These principles are predefined in the ConceptBase implementation of Telos.

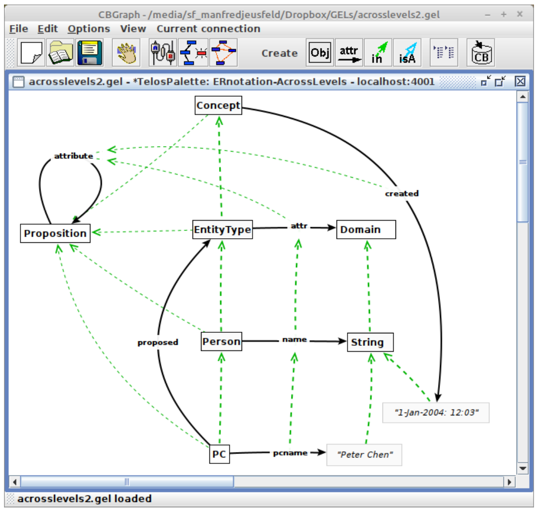
CBGraph - /media/sf_manfredjeusfeld/Dropbox/GELs/acrosslevels2.gel
File Edit Options View Current connection
Create Obj attr in isA CB
acrosslevels2.gel - *TelosPalette: ERnotation-AcrossLevels - localhost:4001
Concept
attribute
created
Proposition
EntityType
attr
Domain
proposed
Person
name
String
"1-Jan-2004: 12:03"
PC
pcname
"Peter Chen"
acrosslevels2.gel loaded

Fig. 1. CBGraph view on the ER Model.

3 Demonstration Part 1: Dual Deep Instantiation

The first part of the demonstration discusses our first attempt to implement constructs for multilevel modeling with ConceptBase. Dual deep instantiation (DDI) is a so-called potency-based approach, where attributes and (binary) relations have a source and a target potency. When both numbers are set to ”1”, then the attribute/relation behaves like a in classical conceptual modeling: the attribute/class is defined at the class level and is instantiated at one level below the current class level. When the numbers are greater than ”1”, then the attribute/relation is instantiated at lower levels depending on the number. For example a metaclass ”ProductType” could have an attribute ”serialnumber” with source potency of 2 and target potency of ”1”. Assume that ”Car” is an instance of ”ProductType”. Then, an instance ”mycar123” of ”Car” can have a serial number, without any need to define it for the class ”Car”. It is defined for all products. The demonstration consists of the following steps:

1. Define the metaclass ”ProductType” and its ”serialnumber” attribute with the desired potencies.
2. Define a class ”Car” with a regular attribute ”numberofdoors”.
3. Define an instance like ”mycar123” that instantiates both attributes.
4. Define another class ”MobilePhone” with attribute ”protocol”. Instantiate ”MobilePhone” and re-use the attribute ”serialnumber”.

We shall also show the formulas defining DDI and how they are compiled to two-level formulas by the partial evaluator of ConceptBase. Dual deep instantiation provides a number of additional capabilities to constrain the use of attributes and relations defined at higher instantiation levels. In particular, attributes/relations with potencies can be restricted at lower instantiation levels.

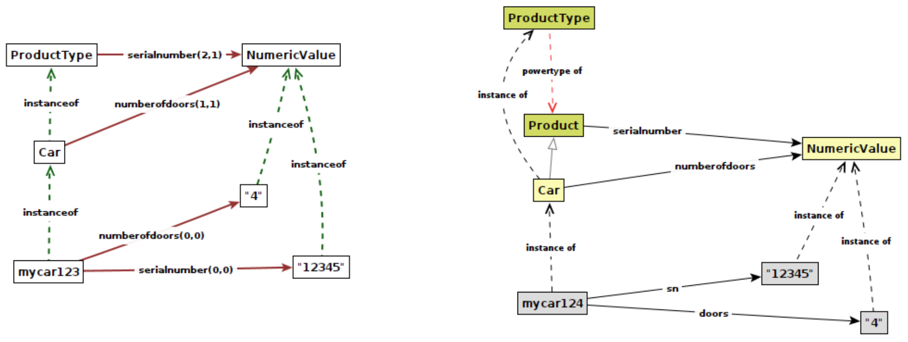

Fig. 2. DDI and MLT-Telos solutions for the modeling task.

4 Demonstration Part 2: MLT-Telos

The second part of the demonstration uses the same example but is based on the powertype construct. Rather than using level numbers, it encodes the multi-level structure of a model by in instantiation hierarchy of powertypes. The attributes/relations are then defined at the suitable level of this hierarchy. Multiple hierarchies can co-exist and be linked via relations. The demonstration consists of the following steps:

1. Define the metaclass ”ProductType” as powertype of ”Product”.
2. Define a class ”Car” as instance of ”ProductType”. Define the attribute ”serialnumber” for ”Product” and ”numberofdoors” for ”Car”.
3. Define an instance like ”mycar123” that instantiates both attributes.

In this simple scenario, the MLT-Telos variant is more concise than DDI but lacks the expressiveness of the dual levels of attributes/relations in DDI. We shall demonstrate also some larger examples from the web site for this demo.

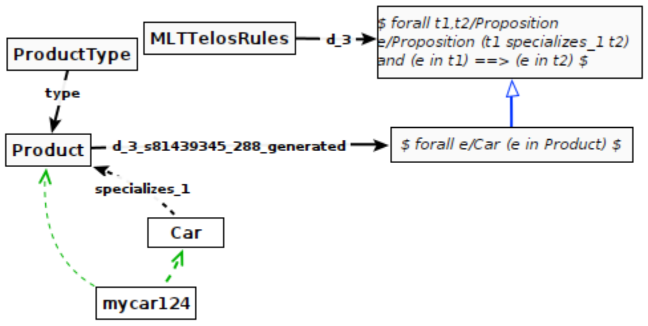
ProductType
MLTTelosRules
d_3
$ forall t1,t2/Proposition
e/Proposition (t1 specializes_1 t2)
and (e in t1) ==> (e in t2) $
type
Product
d_3_s81439345_288_generated
$ forall e/Car (e in Product) $
specializes_1
Car
mycar124

Fig. 3. Partially evaluating a multilevel rule.

5 Demonstration Part 3: Partial Evaluation

We demonstrate the role of partially evaluating rules that range over more than 2 levels to sets of rules that range over two levels, see figure 3. The multilevel rule considered here is

```
forall t1,t2/Proposition e/Proposition
  (t1 specializes_1 t2) and (e in t1) ==> (e in t2)
```

This rule has variables e, t_1 and t_2, where e is an instance of t_2, if t_1 specializes t_2 and e is and instance of t_1. The "specializes" relation is defined as the level of the class "Proposition". So we have three levels of objects here: Proposition is at the highest level, t_1 and t_2 are at the middle level, and the variable e ranges over objects at the lowest level. Now, when we have a (derived) fact like `(Car specializes_1 Product)`, then we can substitute t_1 with "Car" and t_2 with "Product" and we can generate a partially evaluated 2-level formula:

```
forall e/Car (e in Product)
```

ConceptBase automatically generates the 2-level formulas, which are much more efficient to evaluate due to the lower number of variables and thus reduced number of join operations for the Datalog engine.

6 Summary

This paper provided the background for the two multi-level modeling variants DDI and MLT-Telos, both being implemented by the ConceptBase system via its metamodeling and deductive capabilities. The two demonstration examples highlight the differences between the two approaches. In short, MLT-Telos is better integrated with the Telos axioms by reusing its notions for instantiation and specialization. DDI has an elaborate system of rules and constraints expressing how attributes and relations can be specialized and instantiated. The main purpose of the demonstration is to explain how a collection of metaclass definitions plus a set of deductive rules and integrity constraints can capture most of the desired semantics of multilevel modeling paradigms. The web page http://conceptbase.sourceforge.net/er21demo contains further instructions to run the demonstration on your own computer, as well as links to videos of the demonstration.

References

1. Abrial, J.: Data semantics. In: Klimbie, J.W., Koffeman, K.L. (eds.) Data Base Management, Proceeding of the IFIP Working Conference Data Base Management, Cargèse, Corsica, France, April 1-5, 1974. pp. 1–60. North-Holland (1974)
2. Atkinson, C., Kühne, T.: The essence of multilevel metamodeling. In: UML 2001 - The Unified Modeling Language, Modeling Languages, Concepts, and Tools, 4th International Conference, Toronto, Canada, October 1-5, 2001, Proceedings. pp. 19–33 (2001), https://doi.org/10.1007/3-540-45441-1_3

3. Chen, P.P.: The Entity-Relationship Model: Toward a unified view of data. In: Kerr, D.S. (ed.) Proceedings of the International Conference on Very Large Data Bases, September 22-24, 1975, Framingham, Massachusetts, USA. p. 173. ACM (1975), https://doi.org/10.1145/1282480.1282492
4. Frank, U.: Multilevel modeling - toward a new paradigm of conceptual modeling and information systems design. Business & Information Systems Engineering **6**(6), 319–337 (2014), https://doi.org/10.1007/s12599-014-0350-4
5. Jarke, M., Gallersdörfer, R., Jeusfeld, M.A., Staudt, M., Eherer, S.: ConceptBase - a deductive object base for meta data management. J. Intell. Inf. Syst. **4**(2), 167–192 (1995), http://dx.doi.org/10.1007/BF00961873
6. Jeusfeld, M.A.: ConceptBase.cc user manual. Tech. rep., University of Skövde, Sweden (2021), http://conceptbase.sourceforge.net/userManual82/CB-Manual.pdf
7. Koubarakis, M., Borgida, A., Constantopoulos, P., Doerr, M., Jarke, M., Jeusfeld, M.A., Mylopoulos, J., Plexousakis, D.: A retrospective on Telos as a metamodeling language for requirements engineering. Requir. Eng. **26**(1), 1–23 (2021), https://doi.org/10.1007/s00766-020-00329-x
8. de Lara, J., Guerra, E.: Deep meta-modelling with metadepth. In: Vitek, J. (ed.) Objects, Models, Components, Patterns, 48th International Conference, TOOLS 2010, Málaga, Spain, June 28 - July 2, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6141, pp. 1–20. Springer (2010), https://doi.org/10.1007/978-3-642-13953-6_1
9. Neumayr, B., Grün, K., Schrefl, M.: Multi-level domain modeling with m-objects and m-relationships. In: Kirchberg, M., Link, S. (eds.) Sixth Asia-Pacific Conference on Conceptual Modelling (APCCM 2009), Wellington, New Zealand, January 20-23 2009. CRPIT, vol. 96, pp. 107–116. Australian Computer Society (2009), http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV96Neumayr.html