Multi-Level Modeling and Language Engineering. Promoting Reuse, Integrity, and Flexibility of Languages, Models and Software Systems (Tutorial)

Ulrich Frank[1,*,†], Tony Clark[2,†]

[1]University of Duisburg-Essen, Germany
[2]Aston University, Birmingham, UK

Abstract

This tutorial offers an introduction to multi-level modeling and corresponding multi-level language architectures. They make it possible to solve serious problems of traditional modeling and programming languages that are limited to two levels. The additional abstraction they provide not only promotes the reusability, integrity and adaptability of languages, models and software systems, but also allows for new software architectures that feature a common representation of models and programs. Users of these systems are empowered to navigate the conceptual foundation of the software they use at runtime, and if needed, adapt it to changing requirements. The multi-level language architecture and corresponding modeling methods that are subject of the tutorial [1, 2] have been developed over the last fifteen years within the project "Language Engineering for Multi-Level Modeling", LE4MM https://www.le4mm.org [2], with roots going back to the ninities of last century.

Keywords

Conceptual Modeling, Modeling Languages, Executable Models, Language Architecture

1. Goals

First of all, we want to clarify the motivation for the use of multi-level languages architectures – why bother anyway? Second, we want to make it clear that this is not a minor extension of traditional language architectures, but a different language paradigm – making it an opportunity and a challenge at the same time. Third, we want to show that the full potential of multi-level modeling, which has been around for some time, can only be exploited if multi-level modeling languages go hand in hand with multi-level programming languages. Fourth, related to the previous, the participants should experience the additional power modelers and developers are given by a multi-level language engineering, modeling and execution environment. Fifth, they should be enabled to describe and understand the foundational language architecture of this environment. Finally, the participants should be able to assess the potential of multi-level language architectures as well as the prerequisites of introducing them.

2. Audience

This tutorial is appropriate for researchers and industrial practitioners who are interested in how multi-level language architectures contribute to significantly increasing the benefits of conceptual models and DS(M)Ls. Participants that are skeptical in this respect are very welcome, too. Participants should have knowledge of conventional information modelling, as exemplified by ERM and UML, and conventional programming.

ER2025: Companion Proceedings of the 44th International Conference on Conceptual Modeling: Industrial Track, ER Forum, 8th SCME, Doctoral Consortium, Tutorials, Project Exhibitions, Posters and Demos, October 20-23, 2025, Poitiers, France
*Corresponding author.
ulrich.frank@uni-due.de (U. Frank); tony.clark@aston.ac.uk (T. Clark)
0000-0002-8057-1836 (U. Frank); 0000-0003-3167-0739 (T. Clark)

3. Teaching Method

As the participant structure is not known in detail, we consider a rigid teaching scheme to be inappropriate. In order to be able to respond better to the participants, we will conduct a short survey at the beginning to get an idea of the participants' knowledge and expectations. Irrespective of this, our idea of teaching is based on the following cornerstones.

3.1. Emphasis on Interactive Mode

Active participation is key to creating a stimulating teaching experience. We use the following measures to promote the involvement of participants. Teaching content is mainly accessed via questions. For this reason, the presentation of content is accompanied by questions that are put up for discussion. Participants are also encouraged to ask questions or make comments. In order to promote a differentiated understanding of the content, it is important to take a critical look at it. We will therefore not only carefully justify the concepts presented, but also put them up for discussion.

Exercises encourage active engagement with problems. A series of tasks is therefore provided for participants to work on alone or in a group. Interaction is also supported by the use of the XModeler$^{ML©}$, a multi-level modeling and execution environment [1], which extends previous work on language engineering and meta-modeling [3, 4]. It features the multi-level modeling (meta) language FMML^{x} [5], which can be used either with a graphical notation (see Fig. 1) or with a textual representation within a browser inspired by Smalltalk. User feedback is, among other things, promoted by constraints that are checked during the design of a model (or a modeling language respectively). In addition, the tool enables particularly motivating feedback, as models can be executed. This enables participants to create small applications. The screenshot of the XModeler$^{ML©}$ in Fig. 1 gives an impression of this feature. For example, the results of operations are shown in the diagram and updated as soon as an operation has been executed, e.g., as a consequents of a state change. Even if it is not mandatory, we recommend that participants download and install it in advance https://www.le4mm.org.

3.2. Problem-Driven and Solution-Oriented

We assume that most participants are already familiar with numerous modeling languages. Therefore, a convincing motivation is essential to arouse their interest in modeling. To achieve this, we will first present and discuss a number of problems that cannot be solved convincingly with common languages. These are not exotic problems, but essential difficulties associated with the design of models, languages and software systems. Against this background, we will introduce the concepts of architectures step by step and show how they allow the problems presented to be solved.

3.3. From Examples to Theoretical Foundation

Examples are helpful to illustrate problems and possible solutions. They also contribute to the understanding of architectures. However, in order to achieve a deeper understanding, the theoretical basis must be explained. This includes basic principles of conceptual modeling on the one hand and the technical foundation of the XModeler$^{ML©}$ on the other.

3.4. Sustainable Learning

If learning is limited to the time of the seminar, it will hardly be sustainable. It is therefore important to us that the participants we have been able to interest in the topic are supported in anchoring and expanding what they have learned. To that end, they can draw on extensive resources provided on the LE4MM web portal (https://www.le4mm.org). They inlcude various screencasts, examples, supplementary explanations, and an extensive bibliography, not only of our work, but also of literature on multi-level modeling in general. Our work is documented in many publications that are available at https://www.le4mm.org/LE4MM/publications/.

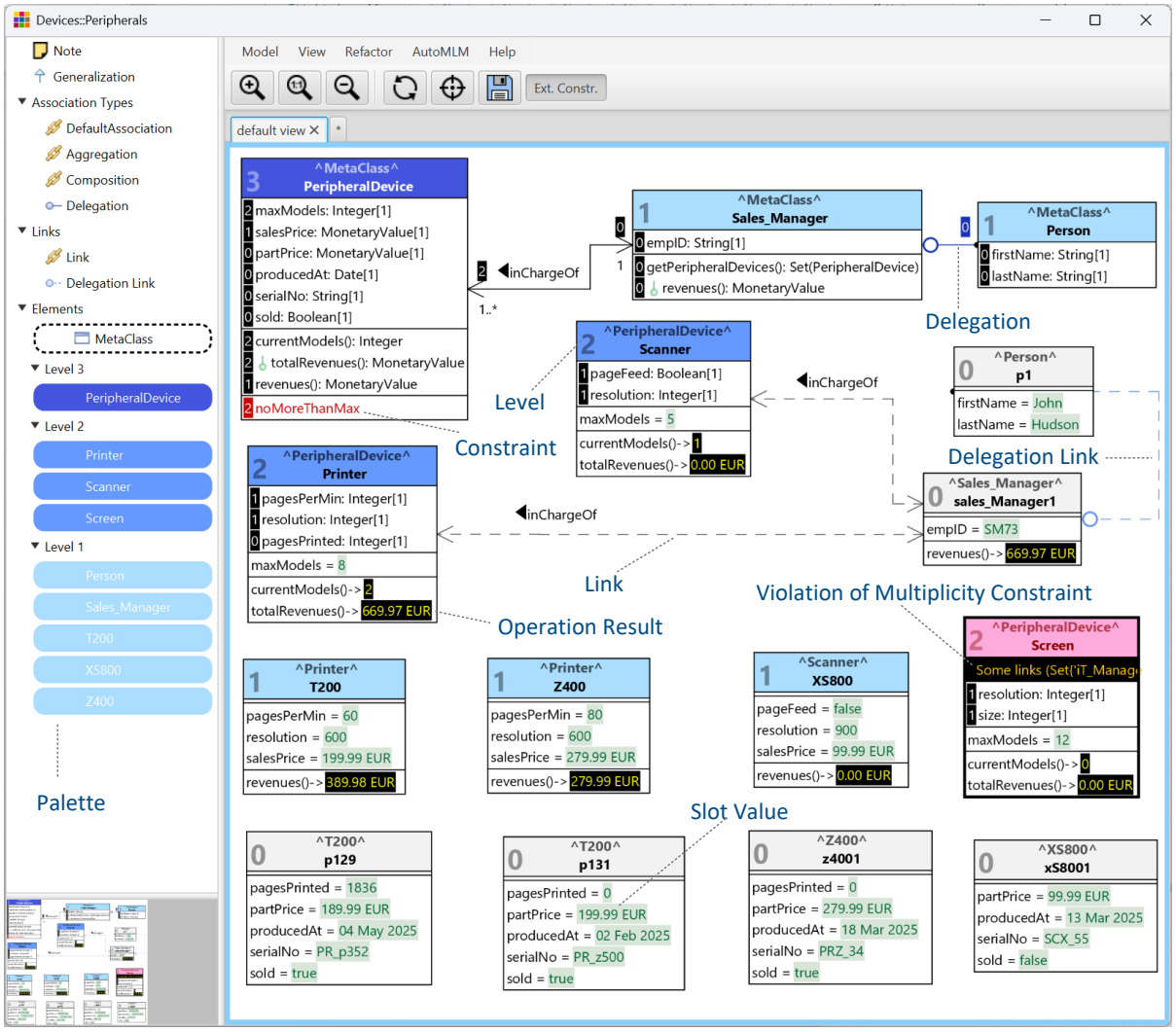

Figure 1: Screenshot of XModeler$^{ML©}$

4. Detailed Outline and Timetable

The following outline describes the content and procedure of the tutorial in more detail. The times given serve as a rough orientation.

1. Introduction (10 min.)
2. Motivation (20 min.)
    - Brief discussion: important objectives of designing models and languages
    - Problems with current language architectures
        - conceptual modeling
        - DSML design
        - Model-driven development
3. Focus on MLM I (60 min)
    - essential characteristics of MLM
    - special features of XModeler$^{ML©}$ and FMML^{x}
    - First MLM - including demo of the XModeler$^{ML©}$

- Assignment 1: Design of a small multi-level model

4. Focus on MLM II (20 min)
    - Top-down vs. bottom-up development of multi-level models
    - Assignment 2: Design of small multi-level DSML
    - Discussion
5. Critical Reflection (20 min.)
    - MLM: substantial improvement ...
    - chance to re-invigorate the idea of reference models
    - contribution to cross-organizational integration of information systems
    - .. but no silver bullet
    - challenges of tight coupling
    - support for process modeling
6. Theoretical/technical Foundation (20 min.)
    - XCore and FMML^{x}
    - outline of a design method
    - MOP
    - constraints ...
7. Concluding Discussion (15 min)

5. Projected Benefits

Multi-level language architectures represent a new paradigm of modeling, language engineering, and software engineering in general. They do not only enable the conjoint development of languages, models and programs without the need for code generation, they also allow for application systems that are integrated with their conceptual models during their entire lifecycle, thus promoting user empowerment. Approaching a new paradigm recommends thoroughly studying its essential characteristics, comparing it with existing paradigms and evaluating its potential. That applies both to software engineering and economic aspects.

To that, the tutorial should encourage participants to adopt a complementary perspective on fundamental aspects of conceptual modeling and thereby re-evaluate the capabilities and limitations of common modeling and programming languages. Participants will be able to assess the potential of multi-level architectures. It includes not only a gain in reusability, integrity, integration and flexibility, but also new architectures of application systems that are characterized by a close integration of models and programs over the entire life cycle.

In particular, participants will benefit from the fact that multi-level language architectures support the design and implementation of significantly more powerful DSMLs. Through a specific design method [6] and the associated quality criteria, the participants receive an orientation for using the supplementary abstraction of multi-level architectures appropriately and reducing design risks.

Finally, those participants who use UML in teaching will benefit from the UML mode offered by the XModeler$^{ML©}$. Unlike other UML modeling tools, it features a close integration of class and instance diagrams and also encourages student motivation by allowing the execution of objects in an instance diagram [7]. This way, students experience a way of modeling and programming they have not seen before.

6. Presenters

Tony Clark is Professor for Software Engineering at Aston University in Birmingham. He is one of the developers of XMF and the XModeler. Tony was co-founder and Technical Director of Xactium Ltd, a software modeling tools company. Tony has been involved in a number of commercial and industrial projects including contributing to the UML 2.0 standard, and consultancies with companies including British Aerospace, BT and CitiGroup.

Ulrich Frank is a Senior Professor at the Department of Business Informatics at the University of Duisburg-Essen. His research interests include enterprise modelling, i.e., the development and evaluation of modelling languages, methods and corresponding tools. During the last 15 years, his research was mainly on the design of multi-level languages and models. To this end, he and his team extended the XModeler to become XModeler©, which implements the FMML^{x} , a multi-level modeling language that integrates modeling and programming.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] U. Frank, Multi-level Modeling: Cornerstones of a Rationale, Software and Systems Modeling 21 (2022) 451–480. URL: https://link.springer.com/content/pdf/10.1007/s10270-021-00955-1.pdf. doi:10.1007/s10270-021-00955-1.

[2] U. Frank, T. Clark, Language Engineering for Multi-Level Modeling (LE4MM): A Long-Term Project to Promote the Integrated Development of Languages, Models and Code, in: J. Font, L. Arcega, J.-F. Reyes-Román, G. Giachetti (Eds.), Proceedings of the Research Projects Exhibition at the 35th International Conference on Advanced Information Systems Engineering (CAiSE 2023), CEUR, 2023, pp. 97–104.

[3] T. Clark, P. Sammut, J. Willans, Applied Metamodelling: A Foundation for Language Driven Development, 2 ed., Ceteva, 2008. URL: https://arxiv.org/pdf/1505.00149.pdf.

[4] T. Clark, P. Sammut, J. Willans, Super-Languages: Developing Languages and Applications with XMF (Second Edition), 2015. URL: http://arxiv.org/pdf/1506.03363.

[5] U. Frank, The Flexible Modelling and Execution Language (fm) – Version 2.0: Analysis of Requirements and Technical Terminology. ICB Research Report, No. 66, University of Duisburg-Essen, 2018.

[6] U. Frank, Prolegomena of a Multi-Level Modeling Method Illustrated with the FMML^{x} , in: Proceedings of the 24th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, IEEE, 2021.

[7] U. Frank, P. Maier, UML-MX: Boosting Power of Object-Oriented Modeling and Enriching User Experience, in: M. Kaczmarek-Heß, K. Rosenthal, M. Suchánek, M. M. Da Silva, Proper, Henderik, A., M. Schnellmann (Eds.), Enterprise Design, Operations, and Computing - CBI & EDOC 2024 Workshops - Case Reports, Forum, Tools & Demos, Doctoral Consortium, iRESEARCH, and MIDas4CS, Vienna, Austria, September 10 - 13, 2024, 2024.