

Part 0 – MOST Introduction

1. Modeling

Prof. Dr. rer. nat. Uwe Aßmann
Institut für Software- und
Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
Technische Universität Dresden
<http://st.inf.tu-dresden.de/teaching/most>
Version WS-21-0.2, 20.11.21

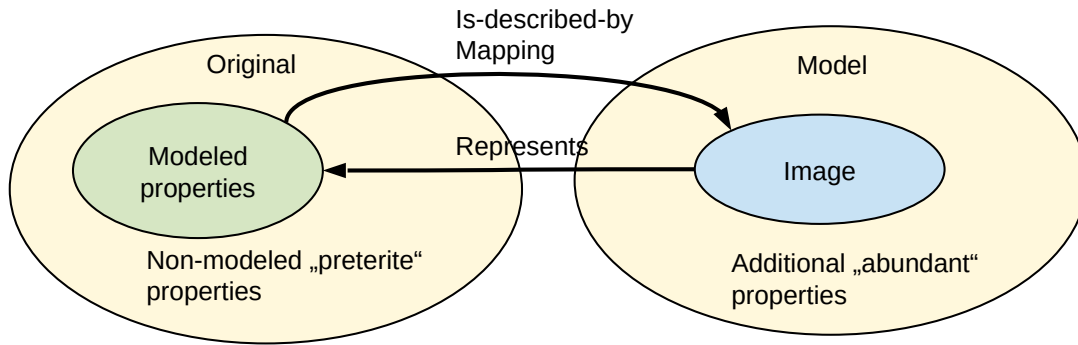


DRESDEN
concept
Exzellenz aus
Wissenschaft
und Kultur

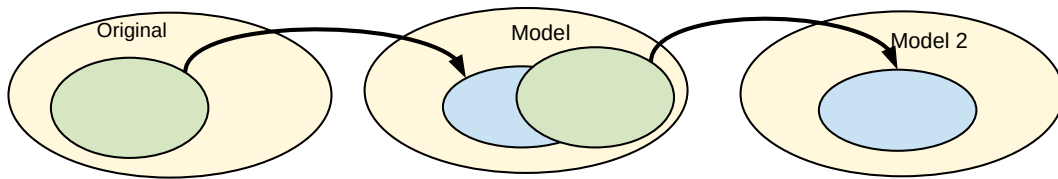
Literature

- ▶ Obligatory:
 - [HesseMayr] Wolfgang Hesse and Heinrich C. Mayr. Modellierung in der Softwaretechnik: eine Bestandsaufnahme. Informatik Spektrum, 31(5):377-393, 2008.
- ▶ References:
 - Stachowiak, Herbert. Allgemeine Modelltheorie. Springer, Wien, 1973

Original and Representing Model



- ▶ [HesseMayr, Stachowiak]
- ▶ Model mappings can be sequenced:



A **model** is an abstraction of an original [Stachowiak]

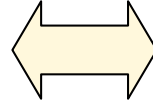
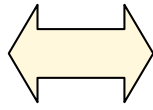
A **system model** is an abstraction of a system

A direct **model** is an abstraction of a reality

A **world model** is an abstraction of a world

An indirect **model** is an abstraction of another model

A **domain model** is an abstraction of a domain of the world



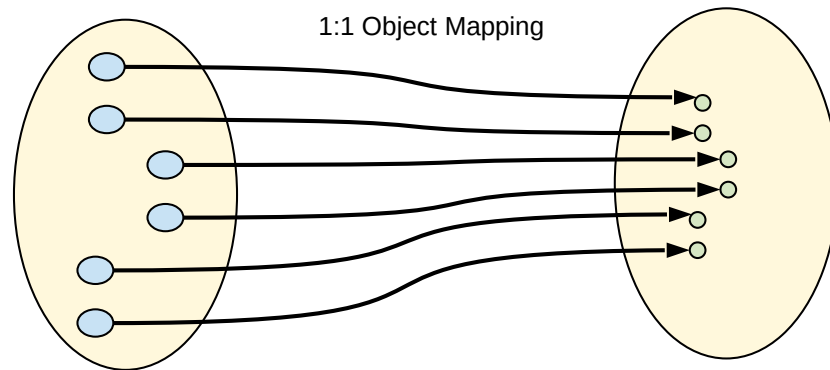
Descriptive
Modeller



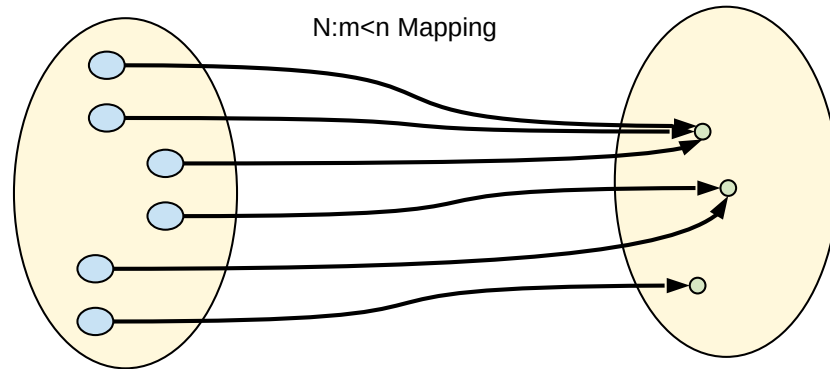
Prescriptive
Modeler;
Specifier;
Implementer

Token Modeling Provides Abstraction of Features of Objects

- ▶ In **Token modeling**, some features of the objects in original domain O are forgotten, but never the objects themselves
 - Abstraction over features
 - Leading to view-based modeling, aspect-oriented modeling

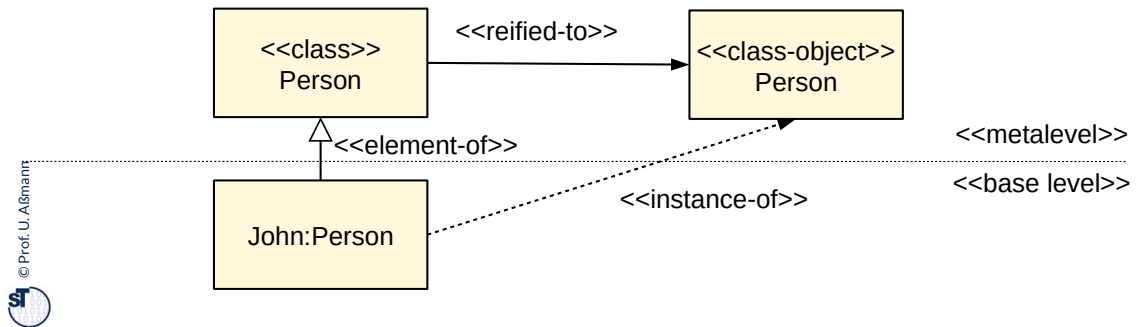


- ▶ In **type modeling**, sets of objects are abstracted



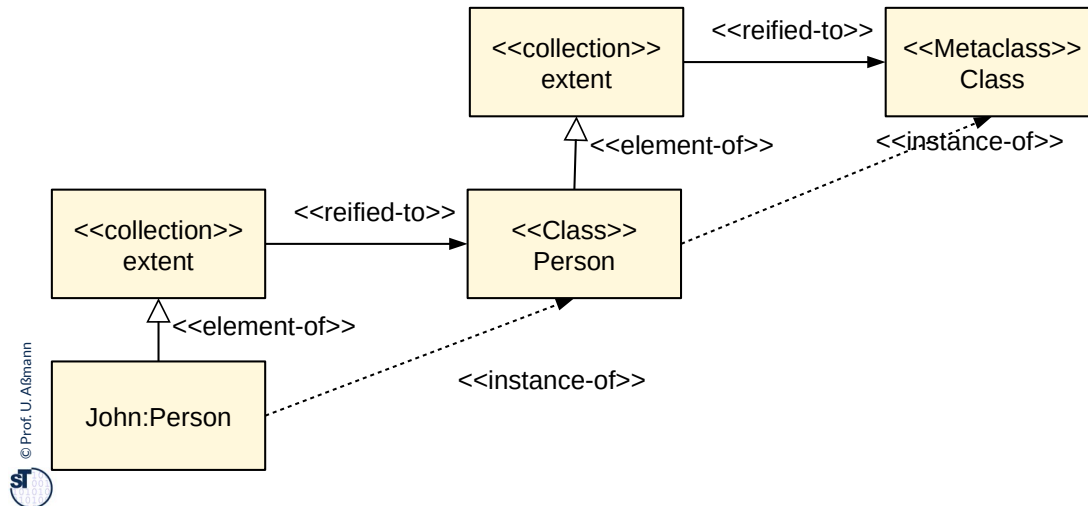
Type Modeling with Reification

- ▶ **Clabjects (class-objects)** are classes reified as *representant objects* on the metalevel.
 - In an object-oriented program, clabjects are objects that represent classes of other objects.
- ▶ Russells Paradox “The set of all sets containing themselves as elements” forbids infinitely many reifications
- ▶ <<instance-of>> is a composition of <<element-of>> with <<reified-to>>

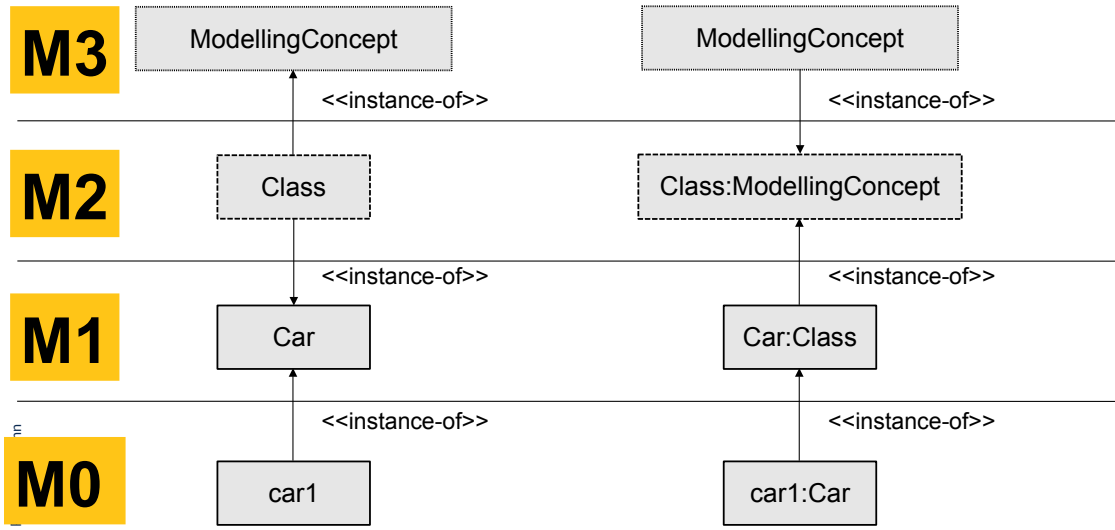


Type Modeling with Reification Works over Several Levels: The Smalltalk Metaclass

- ▶ Smalltalk-80 was the first language to introduce metamodeling
- ▶ It introduced **clabjects** as **class-objects** (and as **metaclasses**).
- ▶ Changing the Smalltalk metaclass changes the semantics of all classes and all objects.
- ▶ In Java, class `Class` is the metaclass, but it is immutable



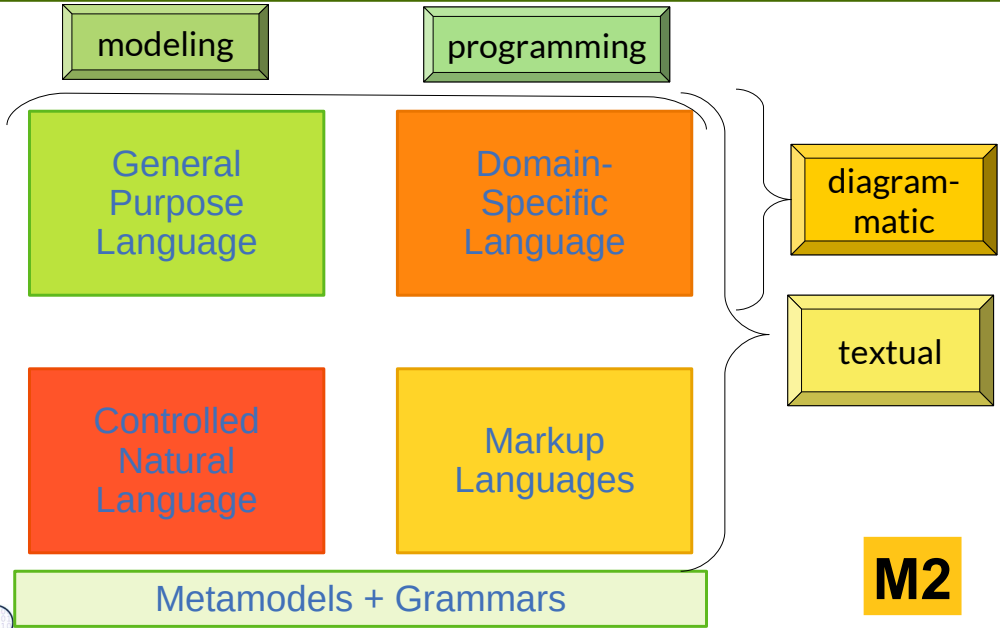
- ▶ We write metaclasses with dashed lines, metametaclasses with dotted lines



Q16: Languages in Software Factories are Built on Metamodels and Grammars

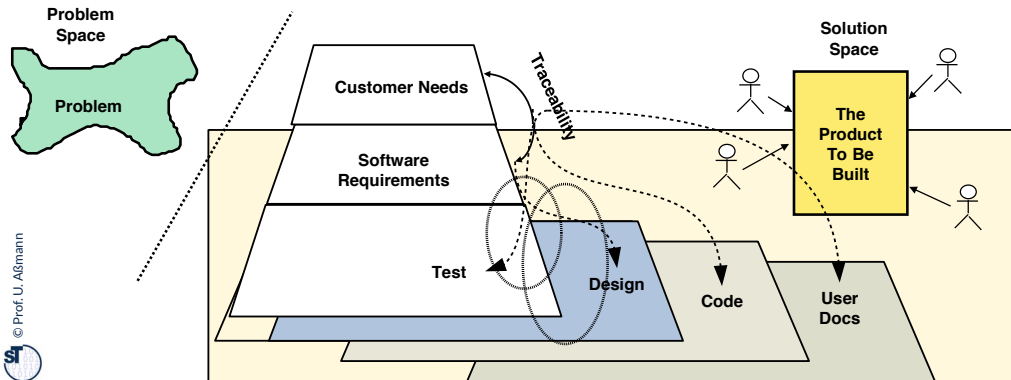
12

Model-Driven Software Development in Technical Spaces (MOST)

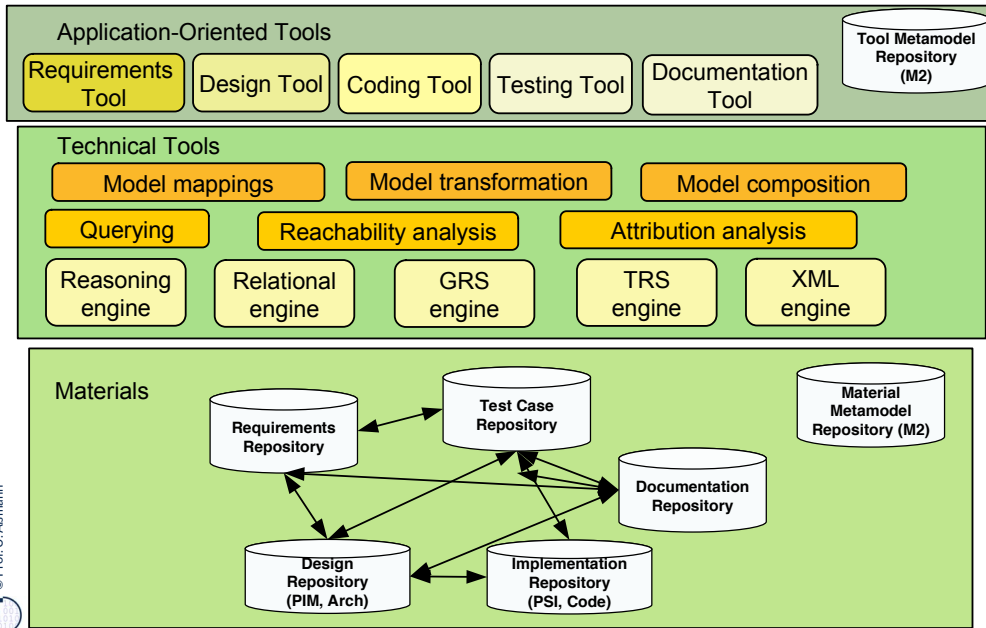


Q1: IDE and Model-Driven Software Development

- ▶ MDSD systematically connects the customer's problems, the system's requirements, testing, design, coding, and documentation and develops these models in coordination
- ▶ MDSD relies on model mappings between requirements, test cases, design, and code
- ▶ **Integrated Development Environments (IDE)** provide tools for all singular aspects, as well as model mappings



Q2: Tool-Objects and Materials in an Integrated Development Environment (IDE, SEU) for MDSD



The End

