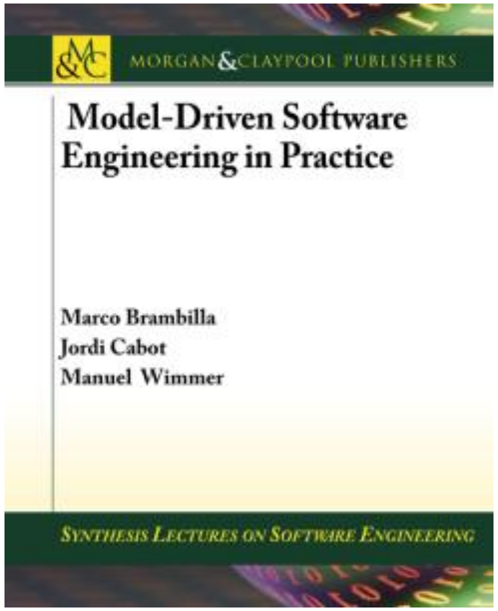
MORGAN & CLAYPOOL PUBLISHERS

Chapter #1

INTRODUCTION

Teaching material for the book
Model-Driven Software Engineering in Practice
by Marco Brambilla, Jordi Cabot, Manuel Wimmer.
Morgan & Claypool, USA, 2012.


MORGAN & CLAYPOOL PUBLISHERS

Model-Driven Software Engineering in Practice

Marco Brambilla
Jordi Cabot
Manuel Wimmer

SYNTHESIS LECTURES ON SOFTWARE ENGINEERING

Introduction

Contents

- Human cognitive processes
- Models
- Structure of the book

Abstraction and human mind

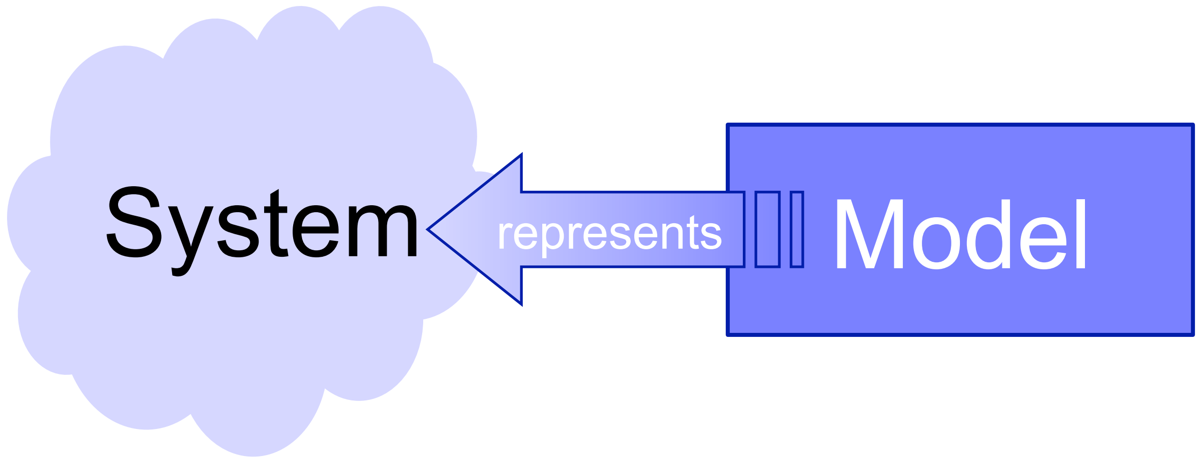
- The human mind continuously re-works reality by applying cognitive processes

- **Abstraction:** capability of finding the commonality in many different observations:
  - generalize specific features of real objects (generalization)
  - classify the objects into coherent clusters (classification)
  - aggregate objects into more complex ones (aggregation)

- **Model:** a simplified or partial representation of reality, defined in order to accomplish a task or to reach an agreement

Models

What is a model?



Mapping Feature	A model is based on an original (=system)
Reduction Feature	A model only reflects a (relevant) selection of the original‘s properties
Pragmatic Feature	A model needs to be usable in place of an original with respect to some purpose

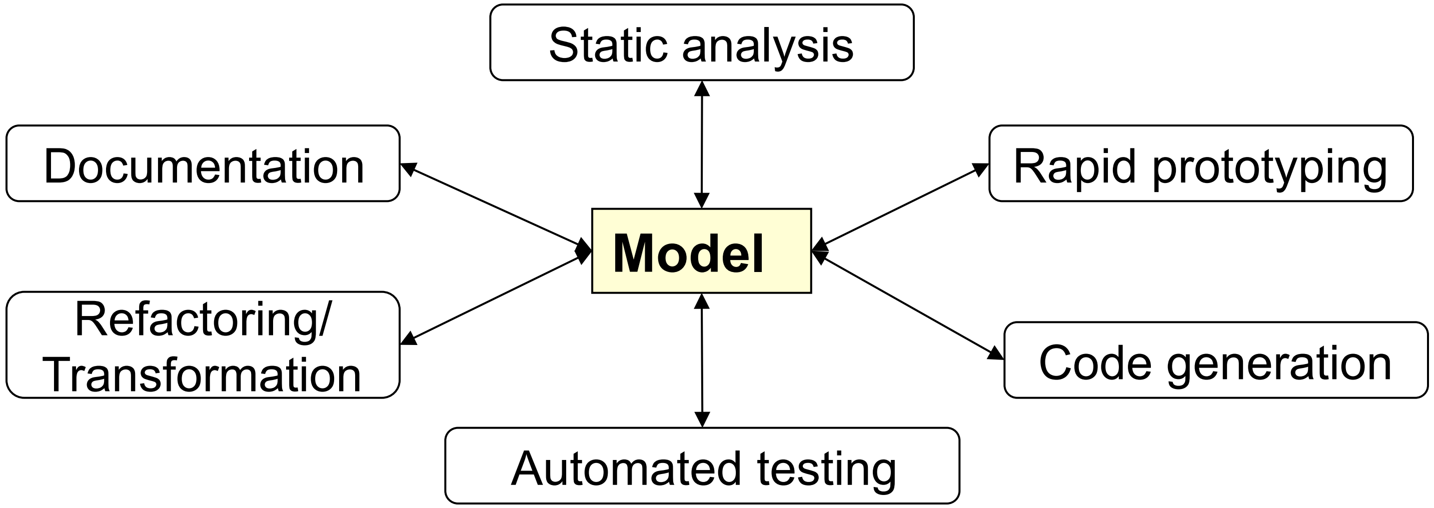
Purposes:

- descriptive purposes
- prescriptive purposes

Motivation

What is Model Engineering?

- Model as the **central artifact** of software development



- Related terms
  - Model Driven Engineering (MDE),
  - Model Driven [Software] Development (MDD/MDSD),
  - Model Driven Architecture (MDA)
  - Model Integrated Computing (MIC)

[Illustration by Bernhard Rumpe]

Motivation

Why Model Engineering?

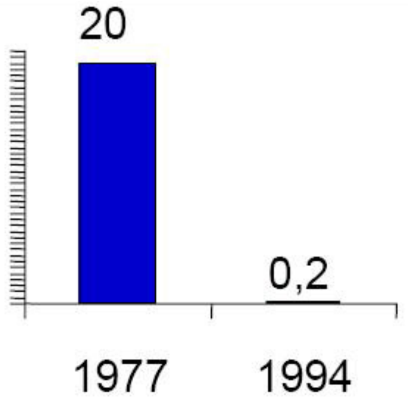
- Increasing **complexity** of software
  - Increasing basic requirements, e.g., adaptable GUIs, security, network capabilities, …
  - Complex infrastructures, e.g., operating system APIs, language libraries, application frameworks
- Software for **specific devices**
  - Web browser, mobile phone, navigation system, video player, etc.
- **Technological progress …**
  - Integration of different technologies and legacy systems, migration to new technologies
- … leads to **problems** with software development
  - Software finished too late
  - Wrong functionality realized
  - Software is poorly documented/commented
  - and can not be further developed, e.g., when the technical environment changes, business model/ requirements change, etc.

Motivation

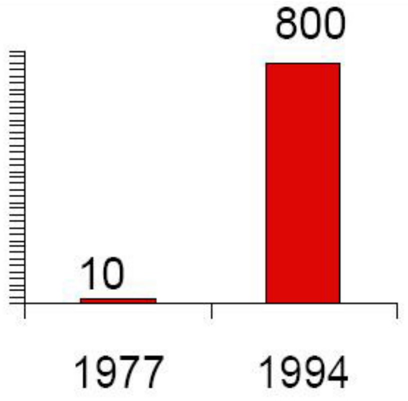
Why Model Engineering?

[Balzert, H.: Lehrbuch der Softwaretechnik: Software-Entwicklung, Spektrum, Akad. Verlag, 1996]

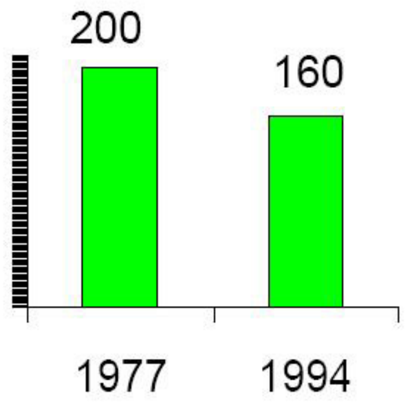
- **Quality problems** in software development



Number of bugs per 1000 LOC



Program size (1000 LOC)



Resulting absolute bug count

Real quality improvements are only possible if the increase in program complexity is **overcompensated** !

(Average values, from Balzert 96)

[Slide by Bernhard Rumpe]

Motivation

Why Model Engineering?

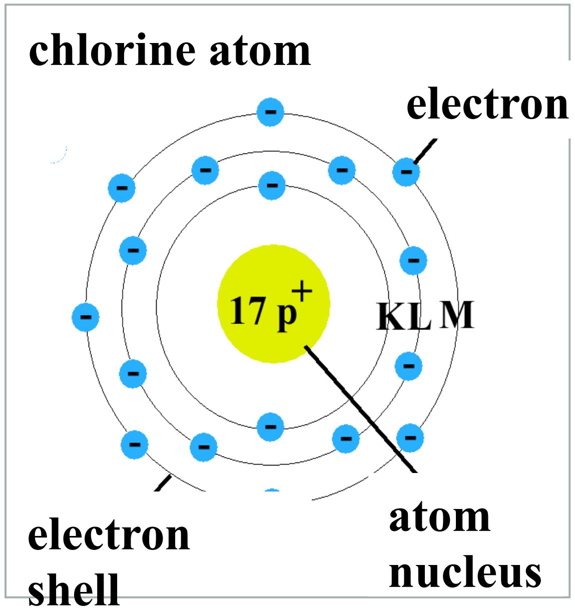
- **Traditional** usage of models in software development
  - **Communication** with customers and users (requirement specification, prototypes)
  - Support for software design, capturing of the **intention**
  - **Task specification** for programming
  - **Code visualization** for understanding

- What is the **difference** to Model Engineering?

Motivation

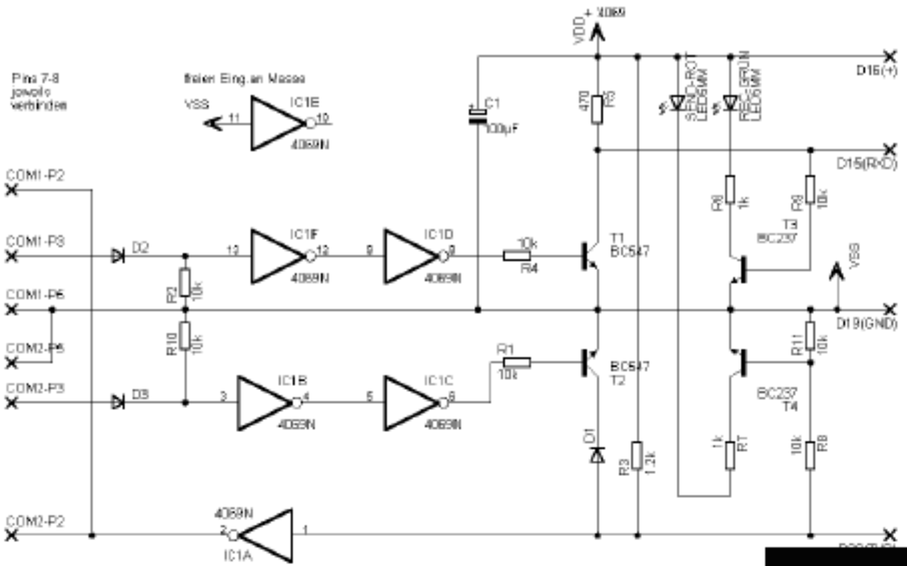
Usage of models

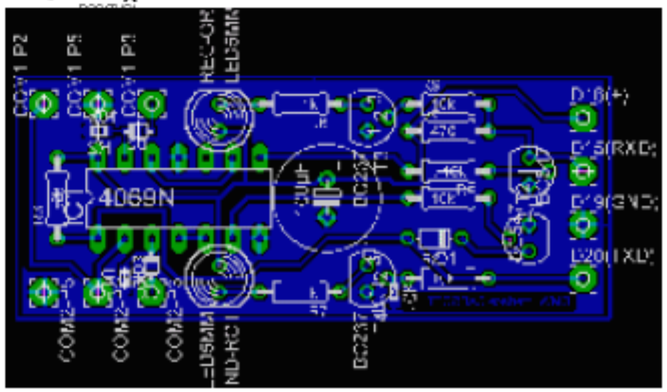
- Do not apply models as long as you have not checked the underlying **simplifications** and evaluated its **practicability**.
- Never mistake the **model** for the **reality**.
  - Attention: abstraction, abbreviation, approximation, visualization, …



Motivation

Constructive models (Example: Electrical Engineering)



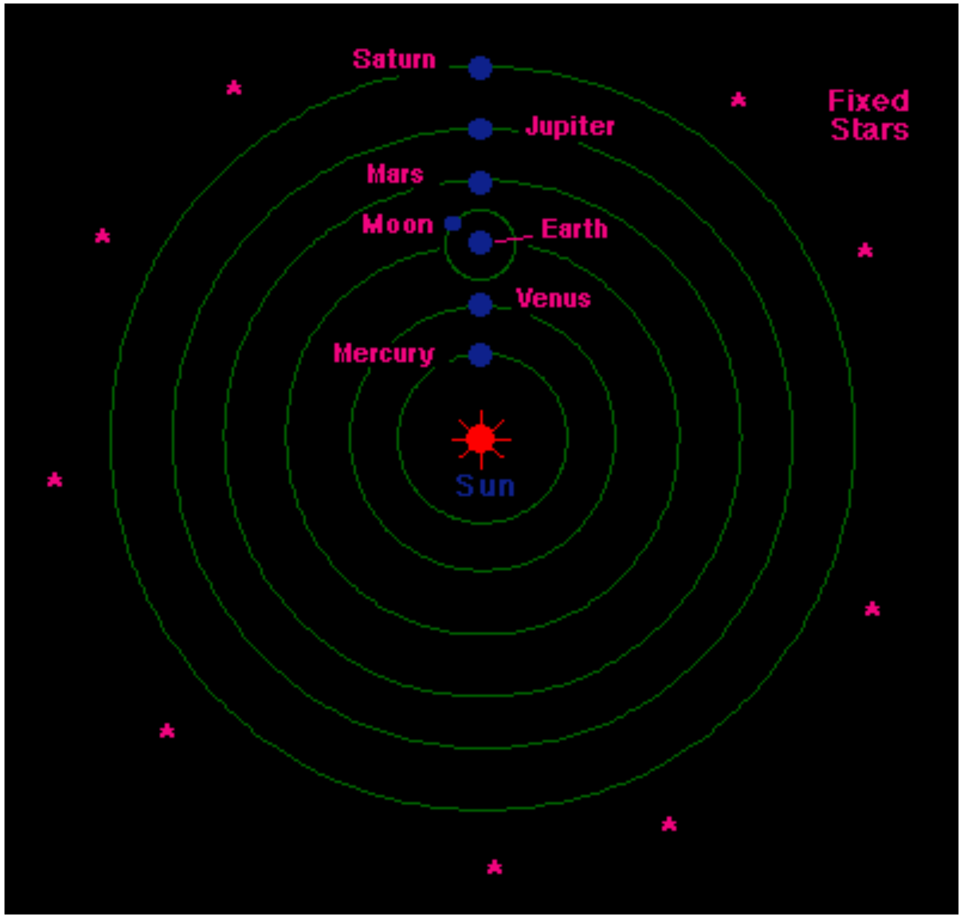


[Slide by Bernhard Rumpe]

Motivation

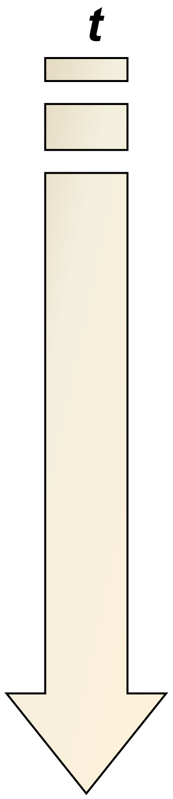
Declarative models (Example: Astronomy)

- Heliocentric model by Kopernikus



Motivation

Application area of modeling

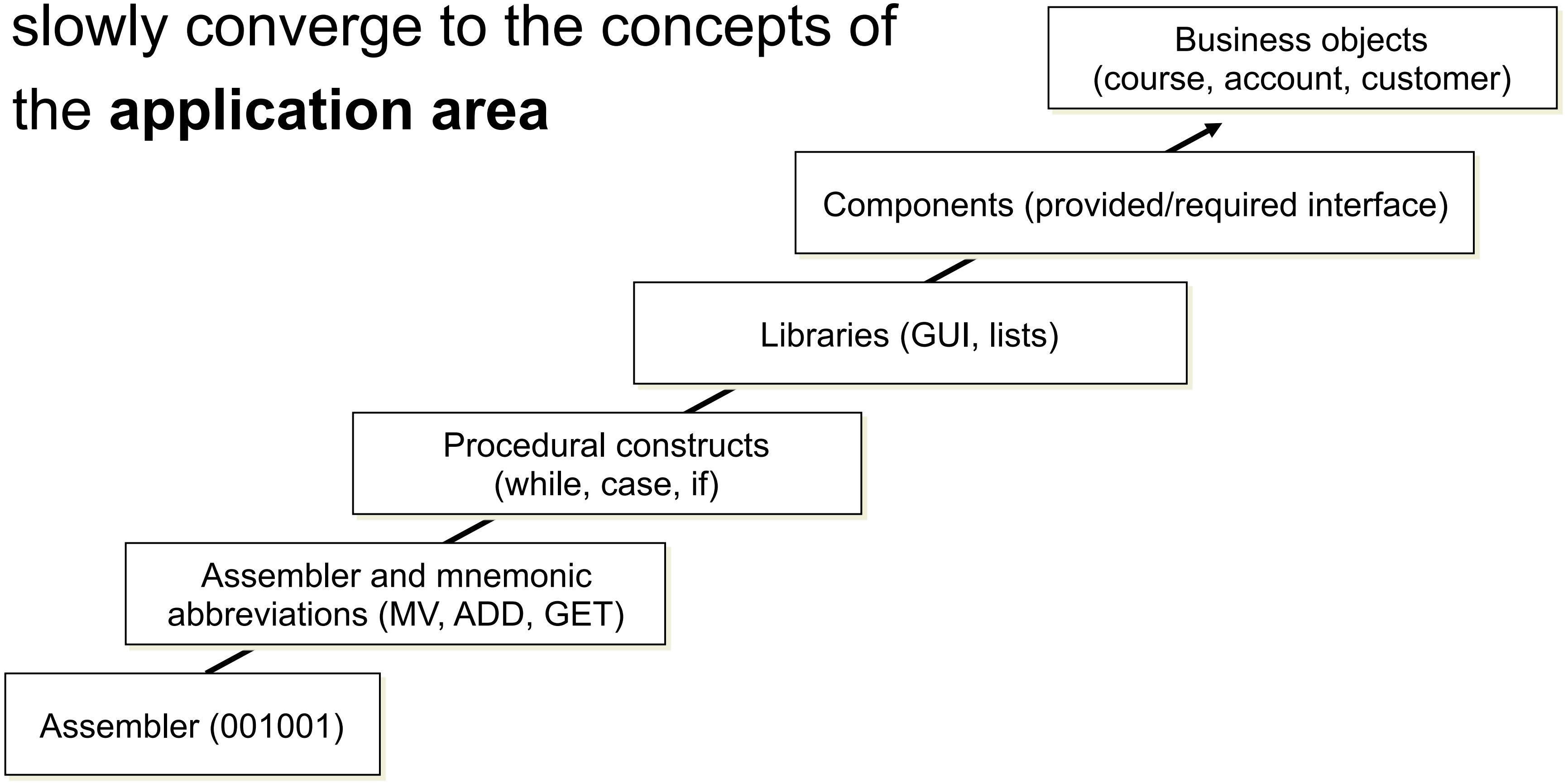


- ***Models as drafts***
  - Communication of ideas and alternatives
  - Objective: modeling per se
- ***Models as guidelines***
  - Design decisions are documented
  - Objective: instructions for implementation
- ***Models as programs***
  - Applications are generated automatically
  - Objective: models are source code and vice versa

Motivation

Increasing abstraction in software development

- The **used artifacts of software development** slowly converge to the concepts of the **application area**


Business objects (course, account, customer)
Components (provided/required interface)
Libraries (GUI, lists)
Procedural constructs (while, case, if)
Assembler and mnemonic abbreviations (MV, ADD, GET)
Assembler (001001)

[Illustration by Volker Gruhn]

Structure of the book

PART 1: MDSE Foundations

Structure of the book

PART 1: MDSE Foundations (continued)

Structure of the book

PART 1: MDSE Foundations (continued)

Structure of the book

PART 2: MDSE Technologies (continued)

Structure of the book

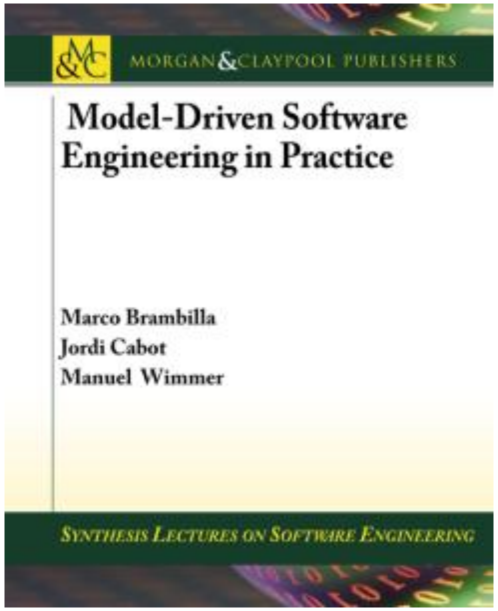
PART 2: MDSE Technologies (continued)

MORGAN & CLAYPOOL PUBLISHERS

MODEL-DRIVEN SOFTWARE ENGINEERING IN PRACTICE

Marco Brambilla,
Jordi Cabot,
Manuel Wimmer.
Morgan & Claypool, USA, 2012.

www.mdse-book.com
www.morganclaypool.com
or buy it at: www.amazon.com


MORGAN & CLAYPOOL PUBLISHERS
Model-Driven Software Engineering in Practice
Marco Brambilla
Jordi Cabot
Manuel Wimmer
SYNTHESIS LECTURES ON SOFTWARE ENGINEERING