



# **Systemdenken und Gestaltungsmethodik**

## **System-Modellierung**

Prof. Dr.-Ing. Stefan Brunthaler  
TFH Wildau 2008ff  
Master Telematik

# Ausgangsbasis

- Es liegt ein kosten-nutzen-optimales Lösungskonzept vor.
- Die Architektur ist damit auch bekannt.
- Die Details („Feindesign“) sind nun konform zu den Requirements zu beschreiben.
- **Dadurch erhalten wir ein System-Modell!**

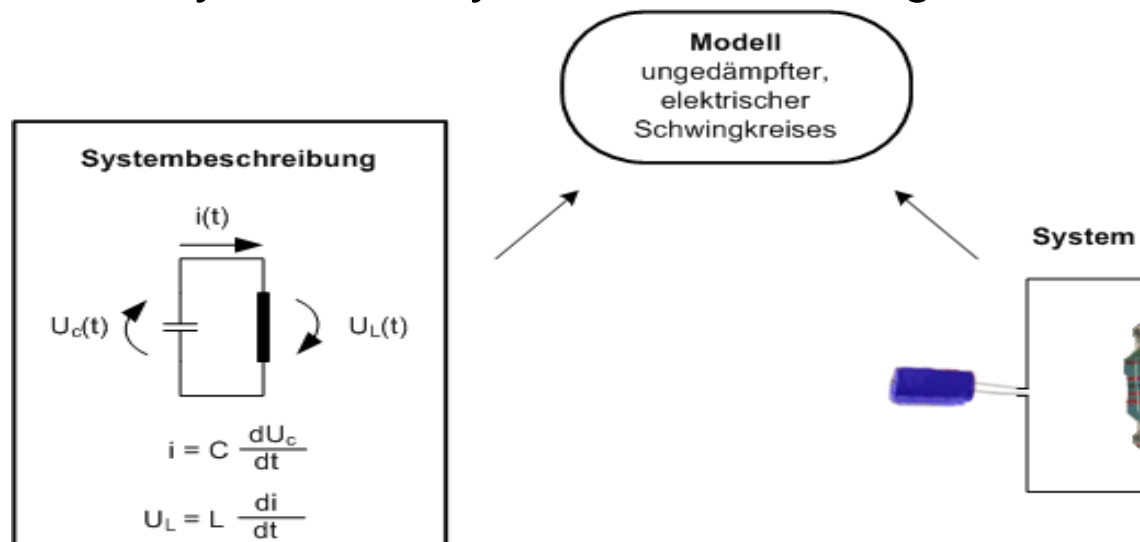
# Was ist ein Modell?

- Abbildung der geplanten „Realität“
- Reduziert auf das Wesentliche (abstrakt)
- Formal beschreibbar (spezifizierbar)
- In verschiedenen Sichten darstellbar ->
- Verifizierbar gegen Requirements

# Definitionsversuch

*Ein Modell ist eine Abstraktion zu einem System, welches nur eine Menge ausgewählter, interessierender Sachverhalte des betrachteten Systems aufweist.*

- Modell vs. System vs. Systembeschreibung/dokumentation



- Häufige Modellausprägungen: Graph, Mengen, Relationen  
--> diskrete Mathematik

# Wozu Modelle?

- Funktion und Architektur prüfen und mit Requirements vergleichen
- Lösungskonzept formal beschreiben
- Anwendungsfälle simulieren
- Akzeptanz bei Nutzern schaffen
- Entscheider überzeugen
- Vorstufe zur Produktion schaffen

# Beispiele für Modelle (1)



# Beispiele für Modelle (2)



Quelle: Kyosho

# Beispiele für Modelle (3)

- Technische Funktionsmodelle
- Lernmodelle (z.B. Medizin)
- Virtual Reality Welten
- Software-Prototypen
- Mathematische Modelle ( $E=MC^2$ )
- Formale Spezifikationen (z.B. UML)



# Unified Modeling Language

- Angesagtes Tool, große Fangemeinde
- Prominente Schöpfer
- Alle UML Bücher beginnen mit:

Es waren einmal 3 Softwareentwickler, die hatten eine Idee. Und wenn sie nicht gestorben ist, dann verwirrt sie uns noch heute!

# Was ist UML?

- UML ist eine **Beschreibungssprache**.
- UML dient der **Modellierung, Dokumentation, Spezifikation und Visualisierung** komplexer Software-Systeme.
- Eignet sich zur Notation von **Analyse, Design und Architektur**.
- Man kann u.a. **Grafiken** daraus erzeugen.

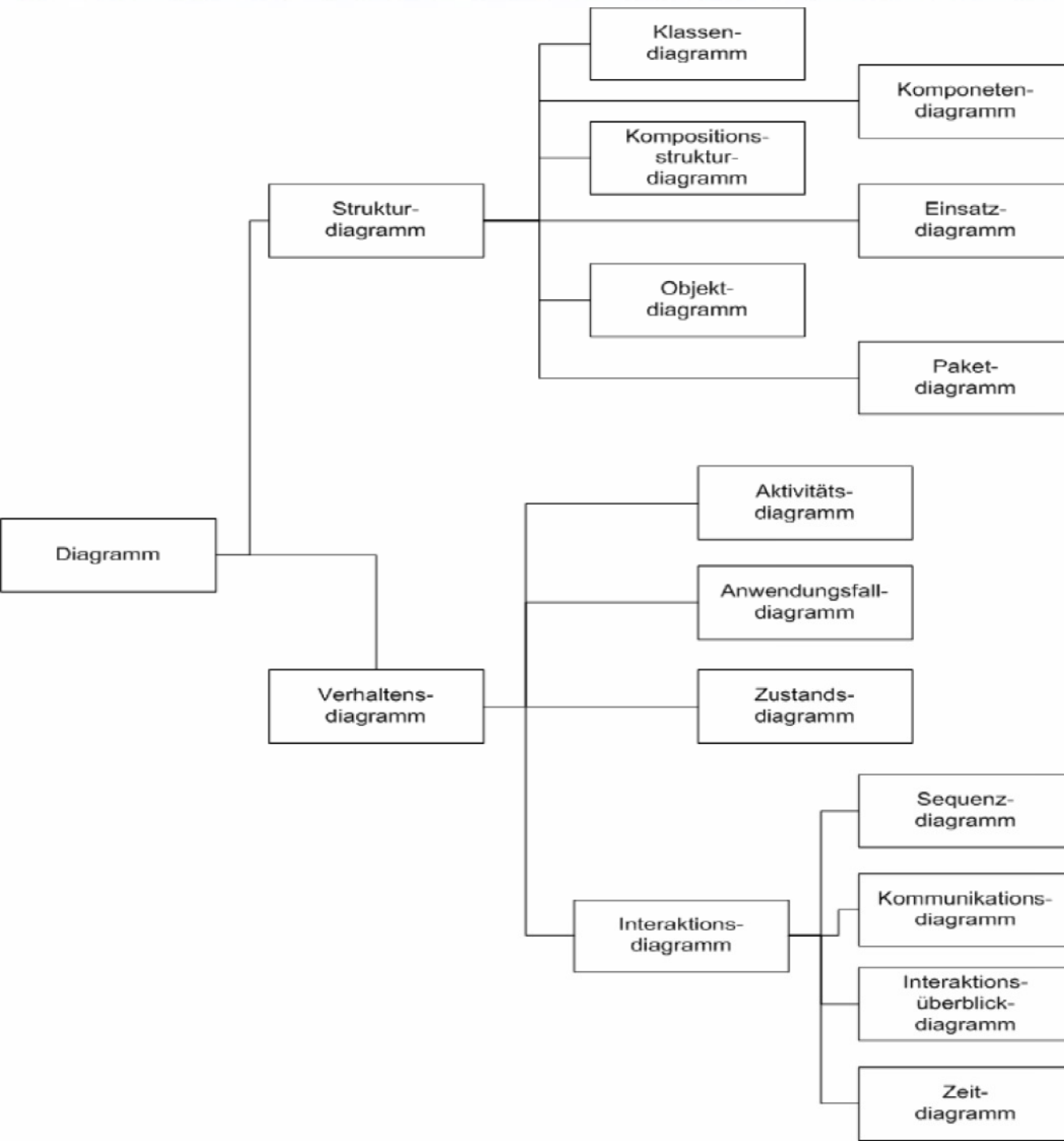
# Marktposition von UML

- DeFacto Standard für
  - OOA (Object Oriented Analysis)
  - OOD (Object Oriented Design)
- Wird in den meisten Informatik-Studiengängen ab dem 1. Semester benutzt (auch bei uns!!!)
- Also nicht wegzudiskutieren.

# Modell-Sichten (Software)

- Funktionssicht (Funktionsstrukturen)
- Datensicht (ERM-Darstellung)
- Dynamik-Sicht (Transitions-Diagramme)

Diese Sichten heißen „orthogonal“, weil sie jeweils disjunkte Aspekte des Modells wiedergeben (überschneidungsfrei)



# Übersicht: Diagramme in UML 2

(Quelle: Kistel 2006)

# Und außer Bildchen?

- Die Idee hinter CASE Tools:  
**Automatisierte Erzeugung, Verifizierung, Tests und Pflege von Programmen.**
- Probleme: Quellcode ist **nicht** ...
  - Lesbar,
  - Änderbar,
  - performant.

# Ansätze zur Code-Optimierung

- Templates, disziplinierte Entwickler
- Programm-Generatoren
- CASE-Tools (klassisch)
- Code-Pflege auf Basis UML ->
- Modellgetriebene Entwicklung (MDA) ->

# UML praktisch – meine Sicht

- Für die Arbeit mit UML gibt es Tools.
- Diese Tools sind „Malprogramme“,  
Beispiel: *MagicDraw* von No Magic, Inc.
- Den Bildelementen kann man Attribute mitgeben, aber das ist individuell
- Die Attribute *können* für die Erzeugung von Code benutzt werden

# UML Diagramme (1)

- Funktion + Daten = Objektmodell
- (User-) Interaktion = Use-Cases
- Abläufe = Sequenzmodell
- Repräsentiert in UML durch verschiedene Diagramm-Typen

# UML Diagramme (2)

- Es gibt in UML 2 ...
  - Struktur-Diagramme,
  - Verhaltens-Diagramme,
  - Interaktions-Diagramme.
- Bekannte Vertreter:
  - Klassendiagramm
  - Use-Case-Diagramm

# Model Driven Architecture (MDA)

- MDA oder MDSD (model driven software development)
- Basiert auf speziellen Grundkonstrukten
- Diese sind fachspezifisch (Fachdomänen)
- Abwandlung für den jeweiligen Zweck

# MDA und UML

- MDA nutzt UML als Basismodelliersprache
- Applikation als plattformunabhängiges Modell (Platform Independent Model, PIM)
  - ist ein UML-Modell
  - hängt von keiner Technologie ab
- PIM umwandeln in Platform Specific Model, PSM
  - Modell eines Systems für konkrete Anwendungsumgebung
  - Kann in UML verfasst sein, muss aber nicht
  - Aus PSM kann dann Code erzeugt werden

# Quellen

- Rupp, Queins, Zengler: UML2 Glasklar  
Hanser-Verlag 2007, 978-3-446-41118-0
- Jablonski, Hendrik: Ingenieurung und MDA  
Referat im WS 2007/2008 in SDGM (TM/06)
- Kistel, Thomas: Modellierung mit UML  
Referat im WS 2006/2007 in SDGM (TM/05)
- <http://www.cragssystem.co.uk/ITMUML/index.htm>