Kurs OMSI
im WiSe 2011/12

Objektorientierte Simulation mit ODEMx

Prof. Dr. Joachim Fischer
Dr. Klaus Ahrens
Dipl.-Inf. Ingmar Eveslage

fischer|ahrens|eveslage@informatik.hu-berlin.de

LV-Angebot im WS 2011/12

Modul
OMSI-1: Objektorientierte Modellierungs-,Spezifikations- und Implementationstechniken

2 Vorlesungen pro Woche
plus Praktikum

Modul
Industrielle Workflows

2 Vorlesung pro Woche

plus Praktikum

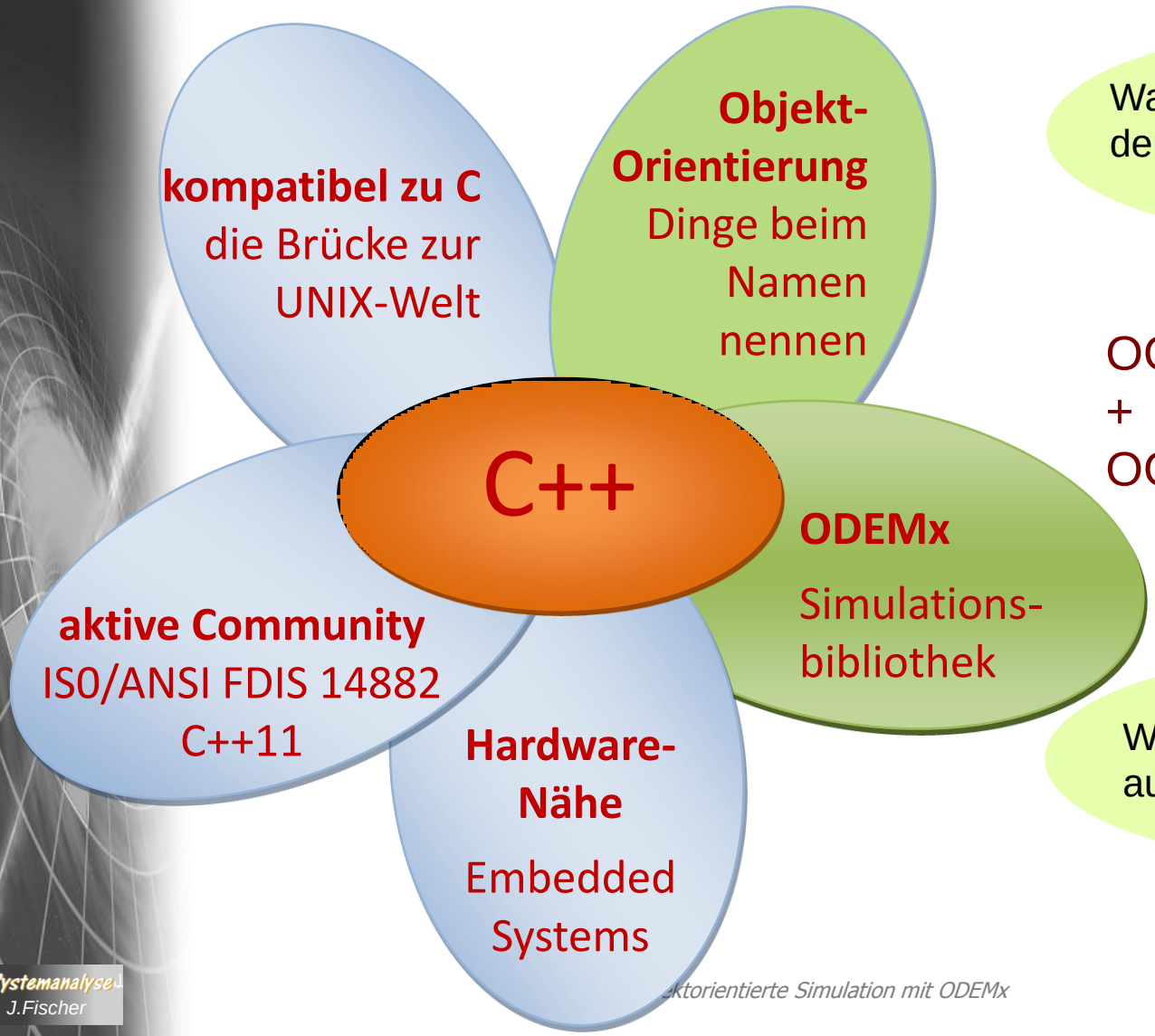
Semester BA-Projekt
Erdbebenfrühwarnung

1 Kompaktvorlesung

plus spezifische Vorlesung
plus Entwicklungsarbeit

Forschungsseminar Modellbasierte Softwareentwicklung
BA-, Studien- Diplom-, Promotionsarbeiten

Sprache, Bibliotheken



Was steckt hinter den Sprachkonzepten?

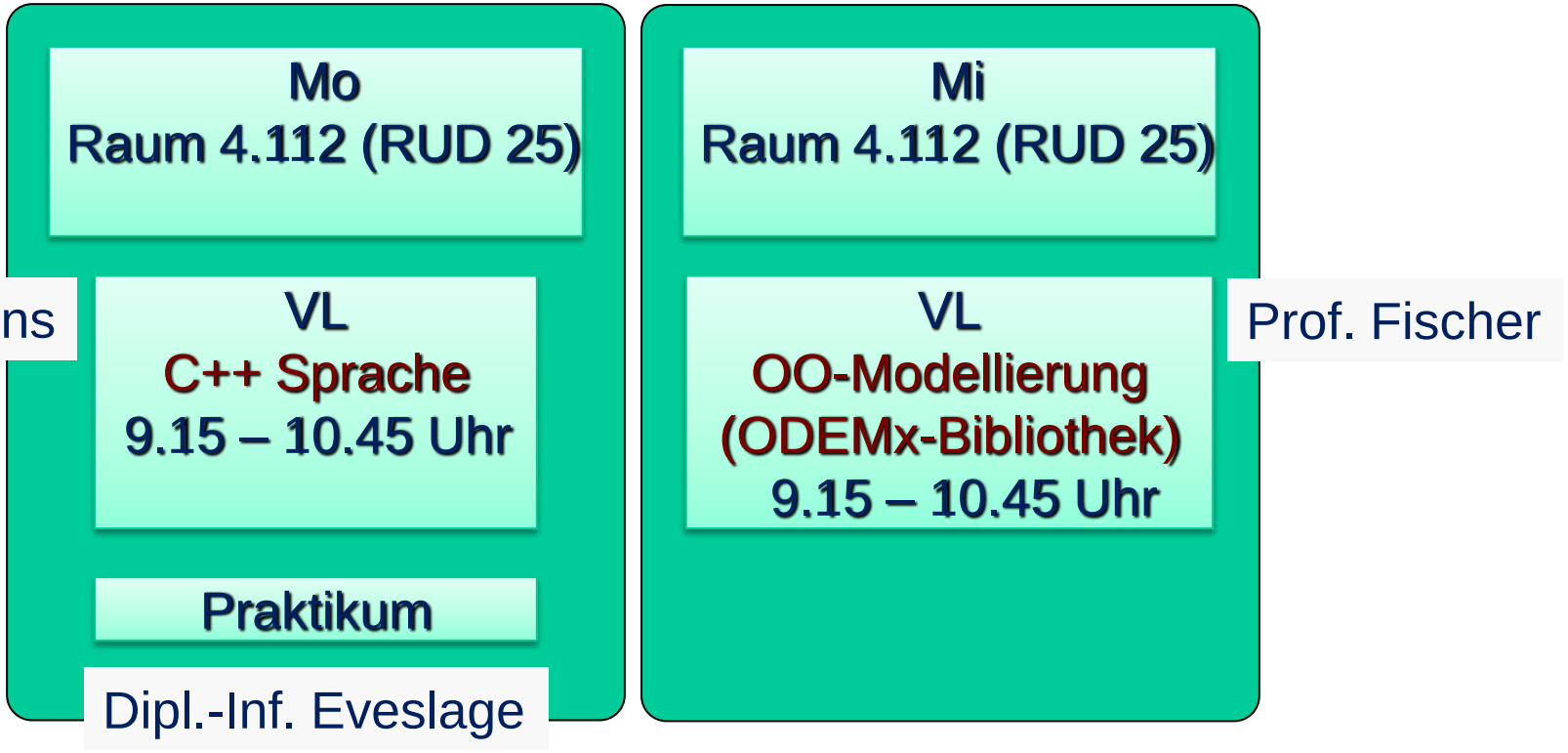
OO-Programmierung
+
OO-Modellierung

Wie kommt man zu ausführbaren Modellen?

OMSI-1: Was, Wann und Wo?

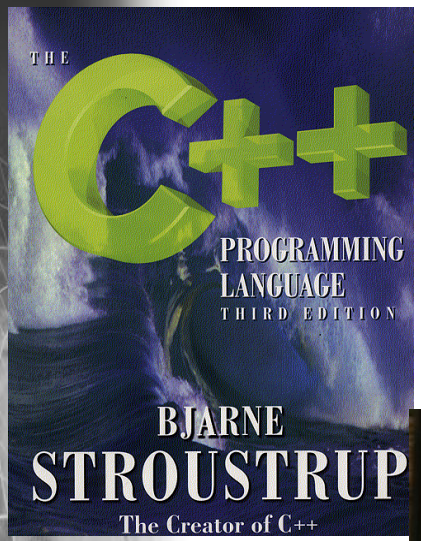
- Eröffnung -

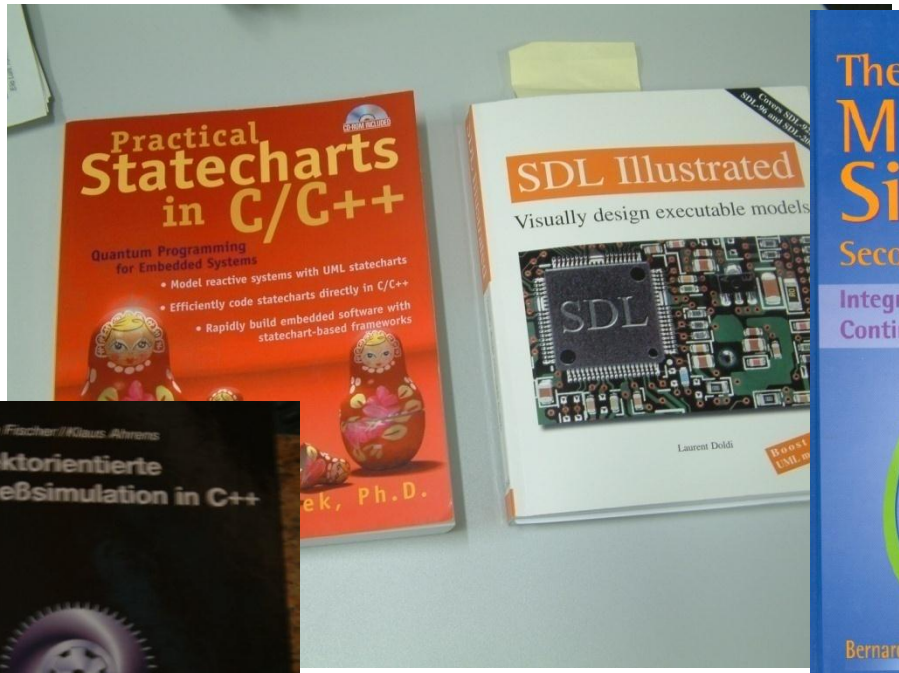
Mittwoch, 19.10.2011

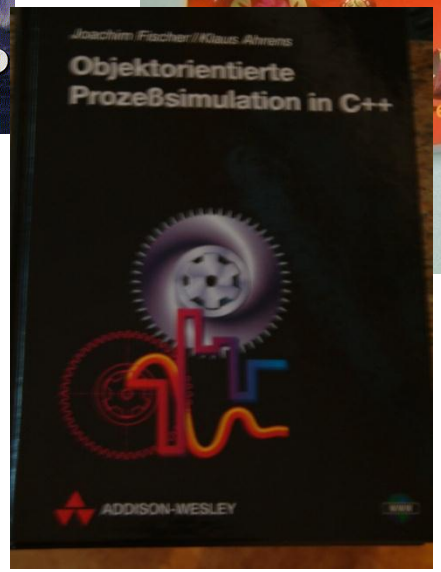


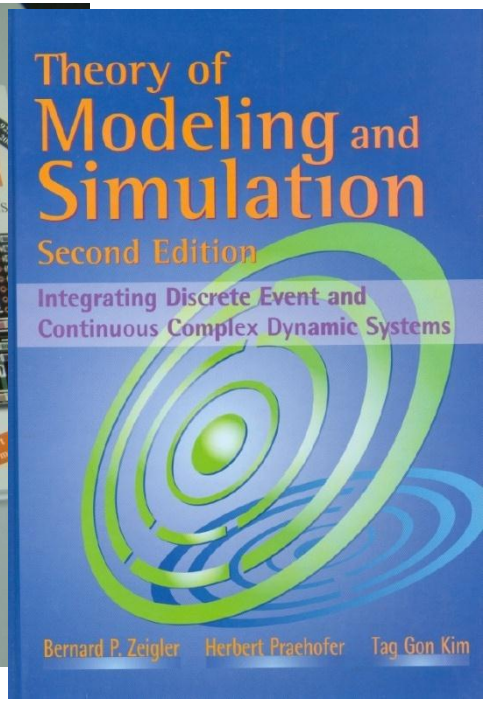
OMSI- Literaturhinweise

Objektorientierte Modellierungs- Spezifikations- und Implementationstechniken









sowie
ODEMx Online-Dokumentation
Skripte und Foliensätze zur Vorlesung

Homepage

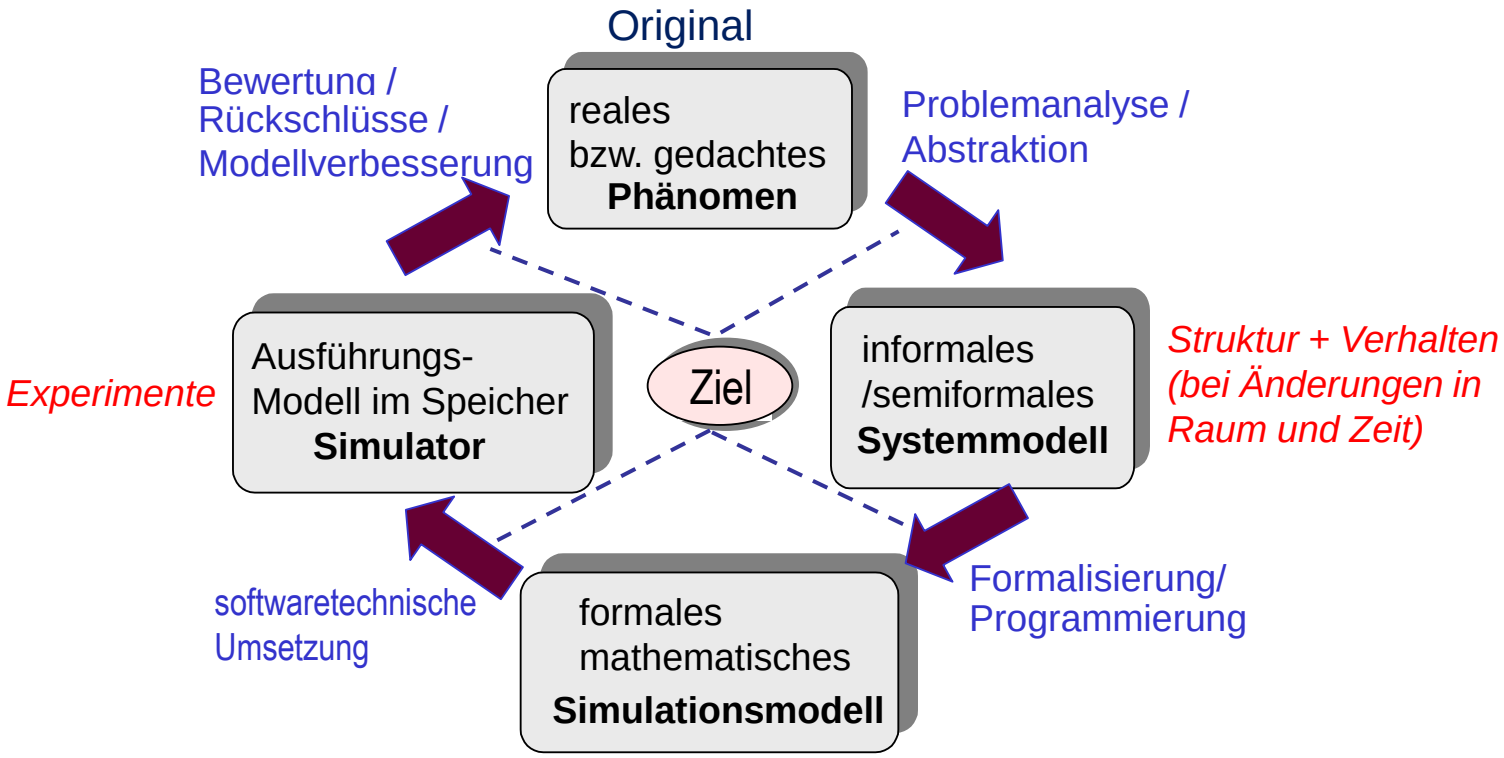
- http://www.informatik.hu-berlin.de/sam/...

1. Einführung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Bespiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Klassifikation dynamischer Systeme
8. M&S eines Niedertemperaturofens

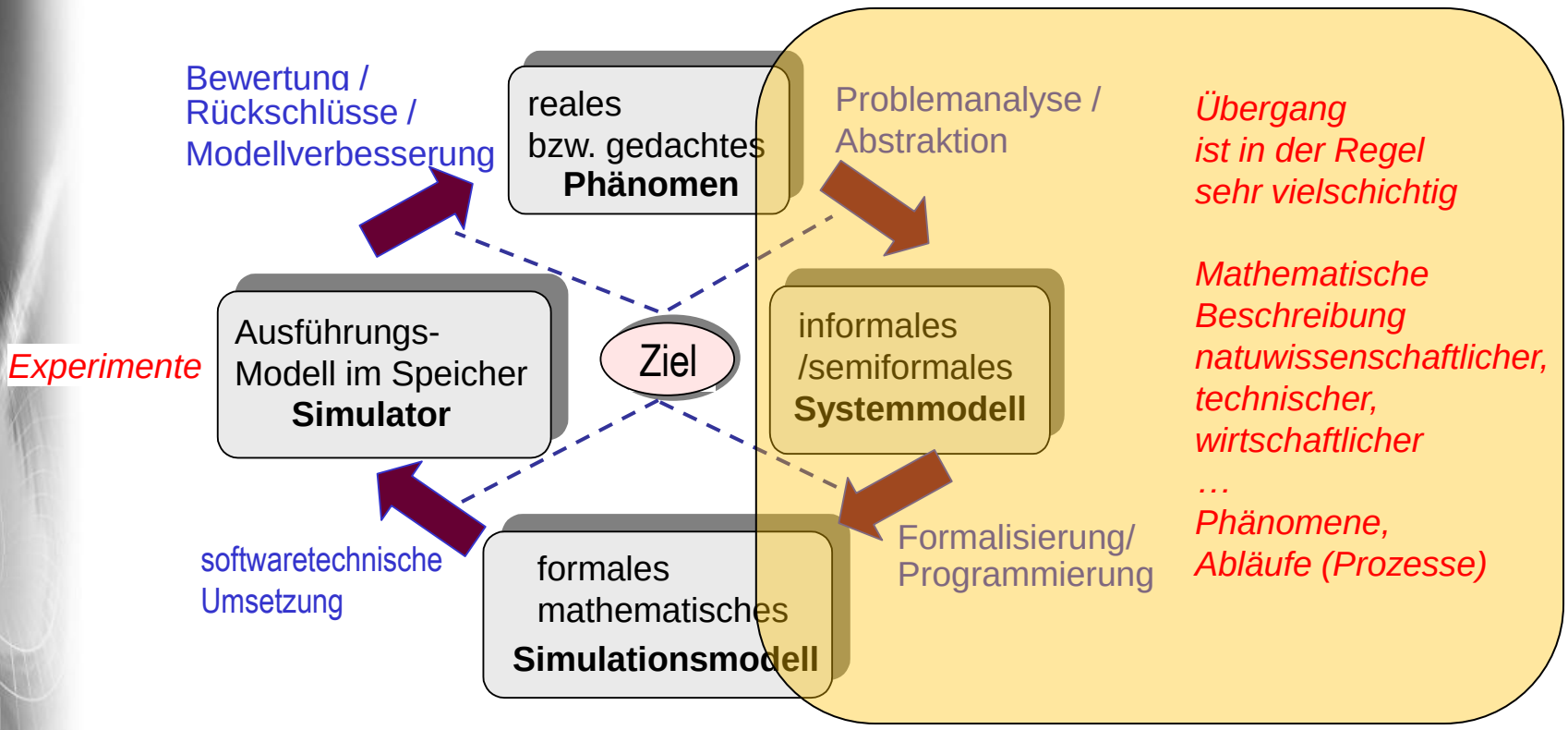
Vorgehensweise bei der Systemsimulation

Experimentieren mit (ausführbaren abstrakten) Modellen auf dem Computer - anstatt mit Originalen -



Vorgehensweise bei der Systemsimulation

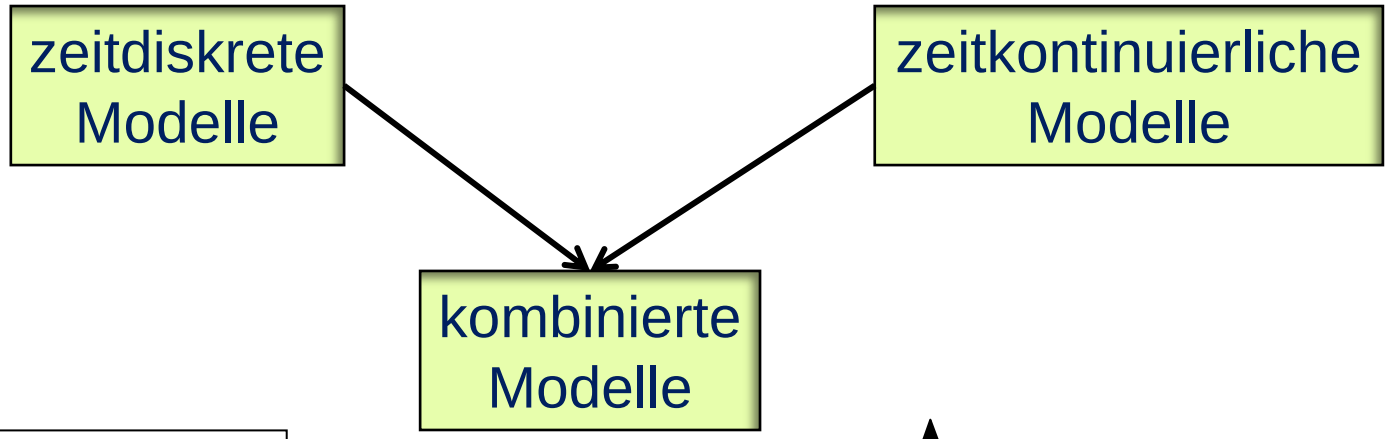
Experimentieren mit (ausführbaren abstrakten) Modellen auf dem Computer - anstatt mit Originalen -

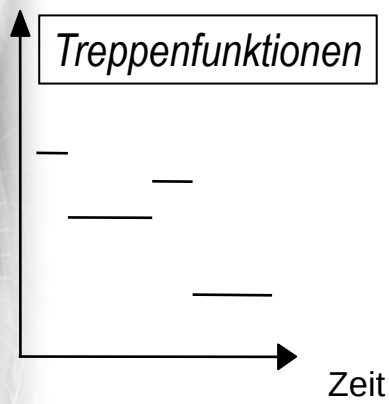


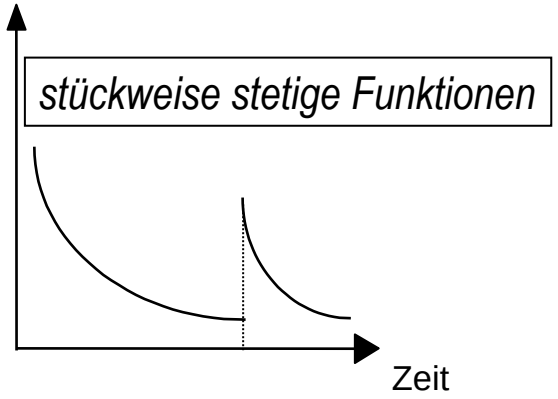
Besonderheit:
Zustandsgrößen ändern sich zeitabhängig (kontinuierlich, diskret / ereignishaft)

➡ dynamische Systeme

Modellierung von Struktur und Verhalten







unsere OO-Auffassung:
System als Konfiguration nebenläufiger Prozesse
mit zeit- und zustandsbedingten Abhängigkeiten

Modelle (in erste Näherung)

- … sind vereinfachte Abbilder der Realität
- ... helfen, die zu entwickelnden Systeme besser zu verstehen
- … ermöglichen (meist vereinfachte) Beschreibung/Darstellung von Struktur
  und Verhalten komplexer Systeme
- … dienen als Vorlagen zum Bau realer Systeme
- … dokumentieren getroffene Entwurfsentscheidungen

abstrakt
mathematische/formale
oder informale Modelle

gegenständliche
Modelle

Simulationsbegriff

Computersimulation ist

- eine **experimentelle** Untersuchungsmethode
- von realen oder gedachten **Systemen**,
- unter Verwendung von formalen **Modellen**,
- die als ausführbare Softwarekomponenten das Verhalten dieser Modelle **näherungsweise**
- im Hinblick auf ein bestimmtes **Untersuchungsziel**

nachbilden.

weitere Aspekte

➔ überwiegend: Untersuchung dynamischer Systeme

➔ Modellierung ist prinzipielle Voraussetzung einer Simulation

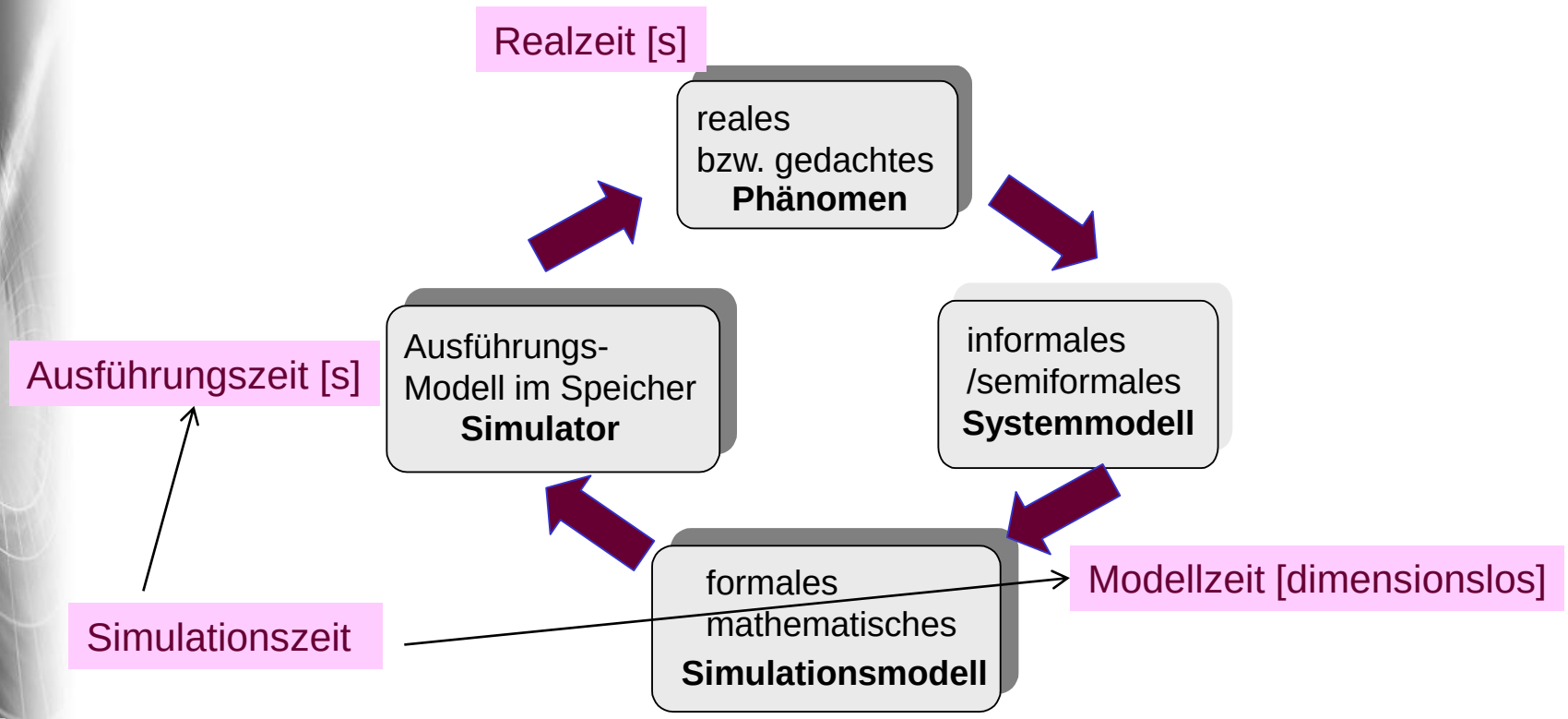
➔ Einsatz von Rechnern
(Simulator= programmierte Maschinenkonfiguration)

Zeitkonzepte

Unterscheidung unterschiedlicher Zeitkonzepte
für die Dauer einer Aktivität oder
die Distanz zweier Ereignisse

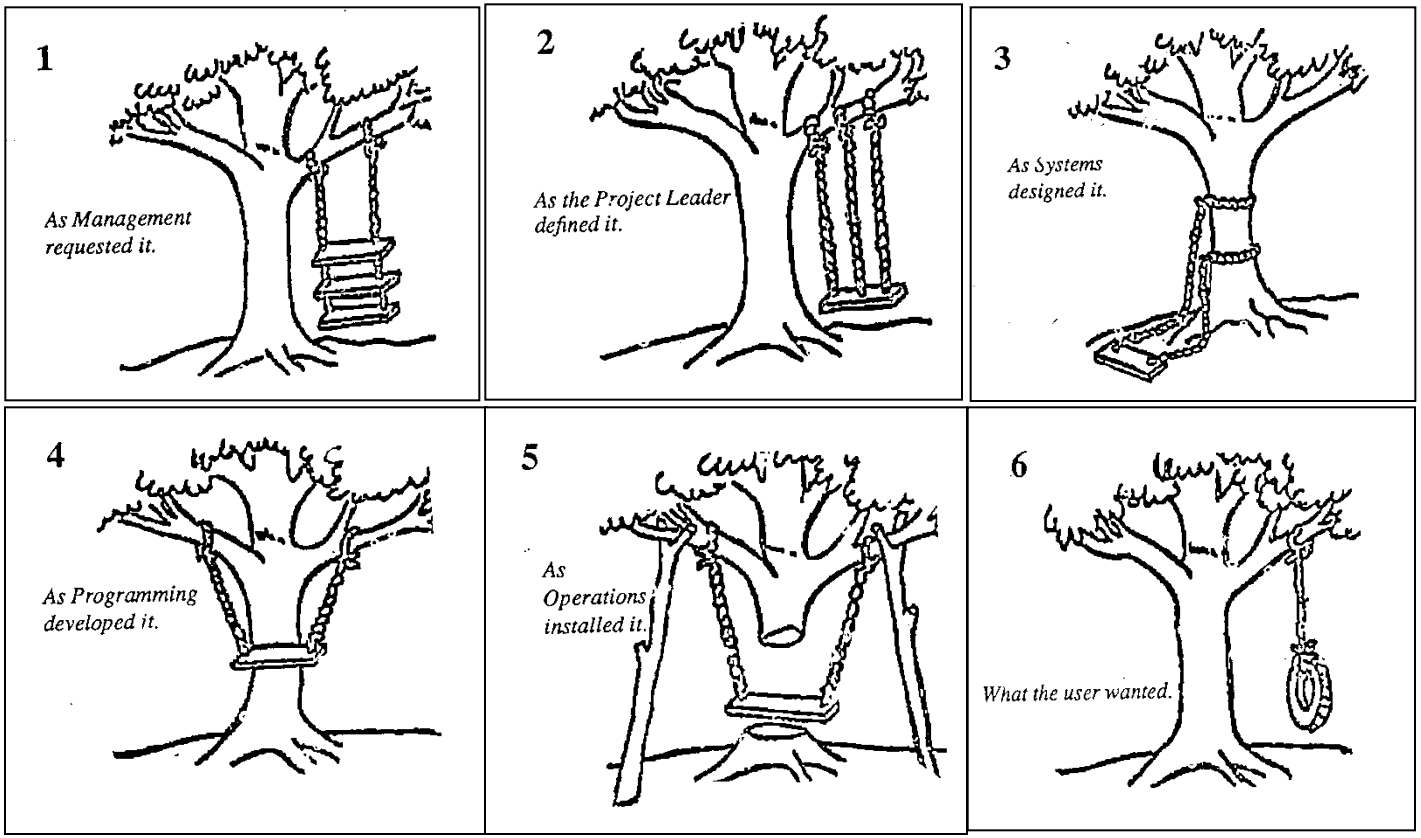
Spezialfall: Echtzeitsimulation
Ausführungszeit ≤ Realzeit

i.allg aber: Zeitlupen oder Zeitraffer

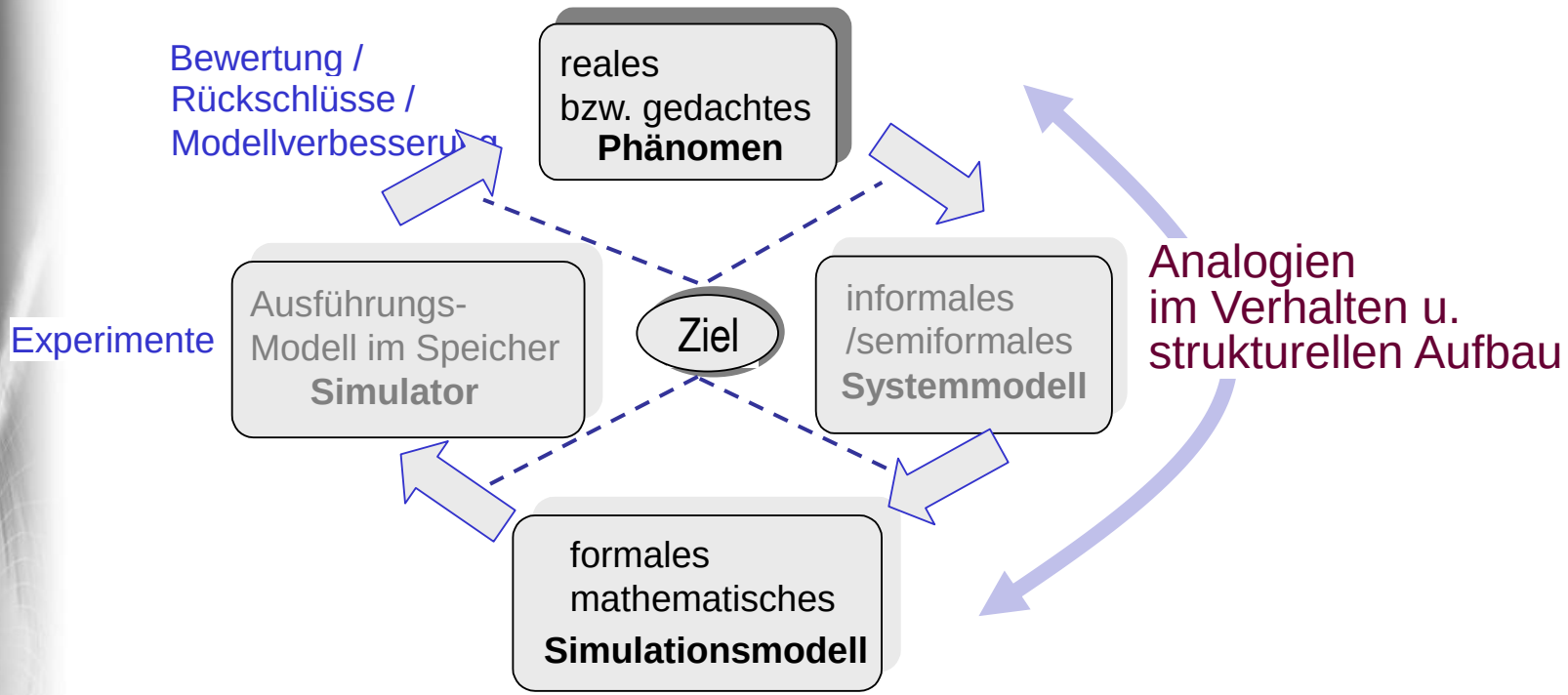


Entwicklung komplexer Systeme

Wozu Modellierung ?



Bedeutung von Analogien



Beachtung: Zustandsgrößen ändern sich zeitabhängig (kontinuierlich, diskret / ereignishaft)

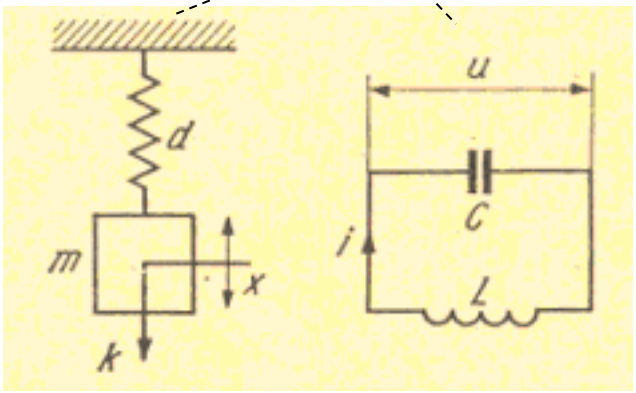
1. Einführung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Bespiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Klassifikation dynamischer Systeme
8. M&S eines Niedertemperaturofens

Analogie im Systemverhalten

Basis für jede Verhaltensmodellierung

betrachten zwei Schwingungssysteme

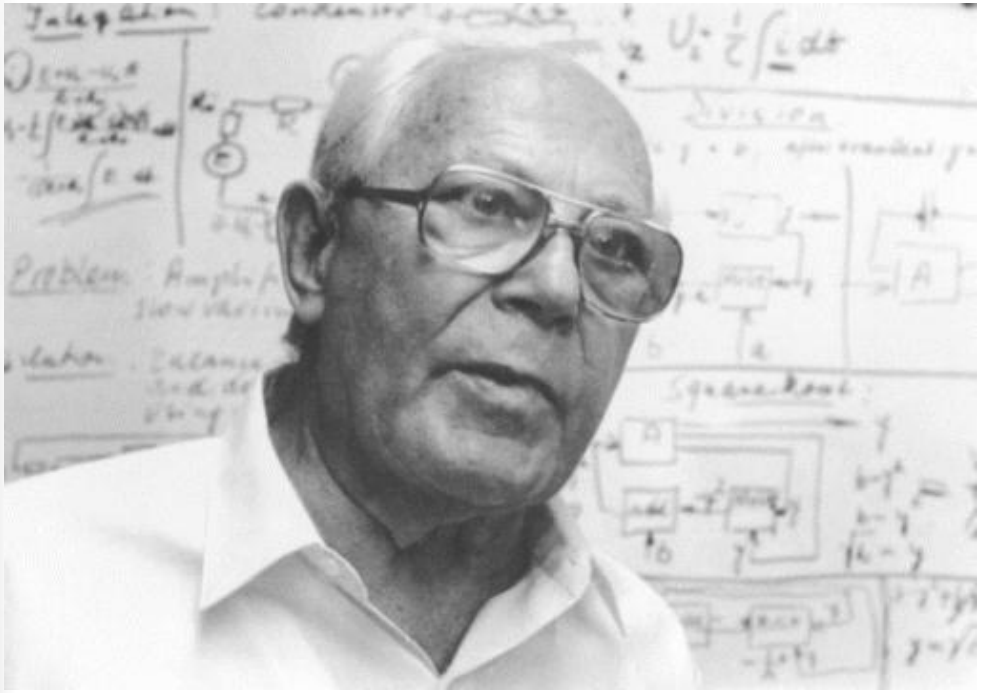


$$x \mathrel{\hat{=}} i$$
$$k \mathrel{\hat{=}} \frac{\mathrm{d}u}{\mathrm{d}t}$$
$$m \mathrel{\hat{=}} L$$
$$d \mathrel{\hat{=}} \frac{1}{C} \quad .$$

Phänomen: strukturell ähnlich Verhaltensbeschreibungen

$$m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + dx = k \qquad L\frac{\mathrm{d}^2i}{\mathrm{d}t^2} + \frac{1}{C}i = \frac{\mathrm{d}u}{\mathrm{d}t}$$

Helmut Hoelzer (1912 – 1996)

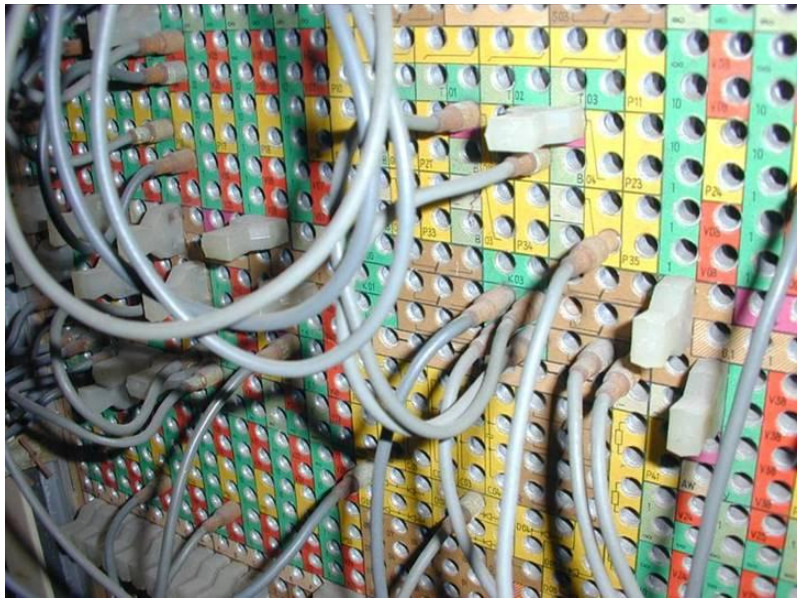


Erfinder des ersten frei programmierbaren Analogrechners (1941)

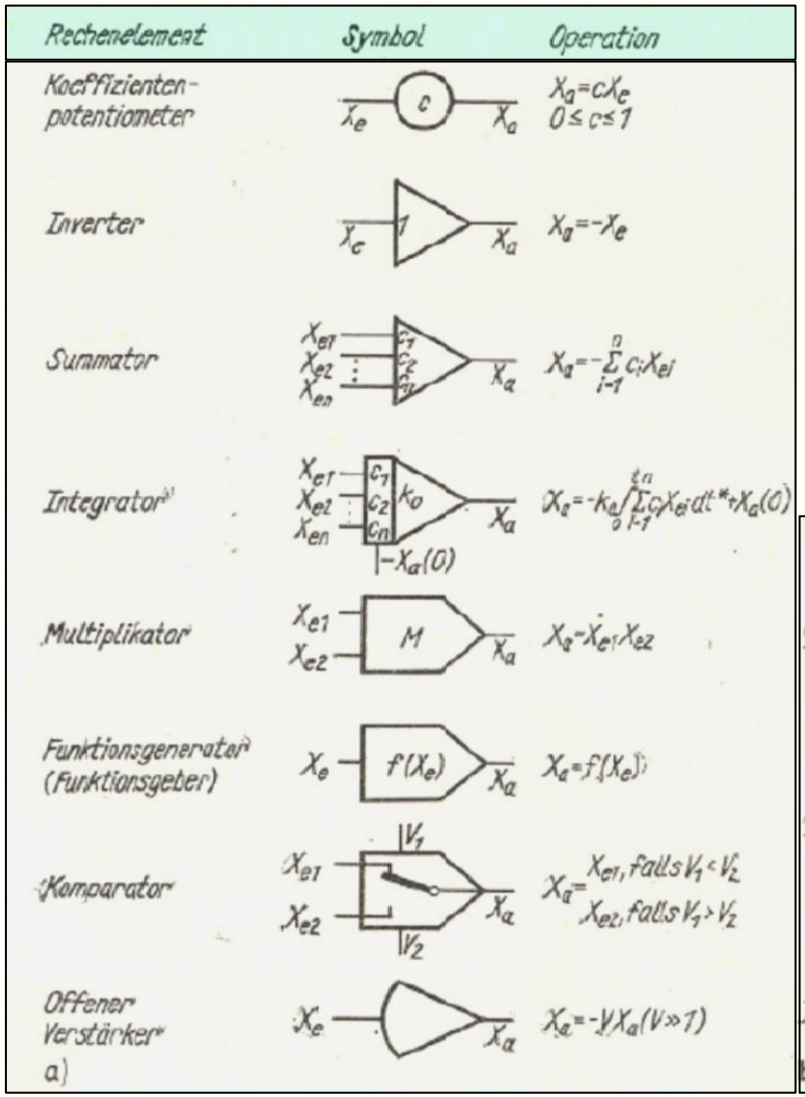
- TH Darmstadt (Diplom)
- Heeresversuchsanstalt Peenemünde (ab 1939)
- Marshal Space Flight Centre Huntsville (ab 1946)

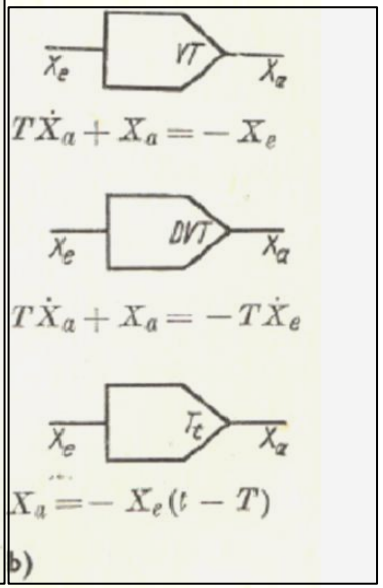
  … Appollo-Programm der Nasa

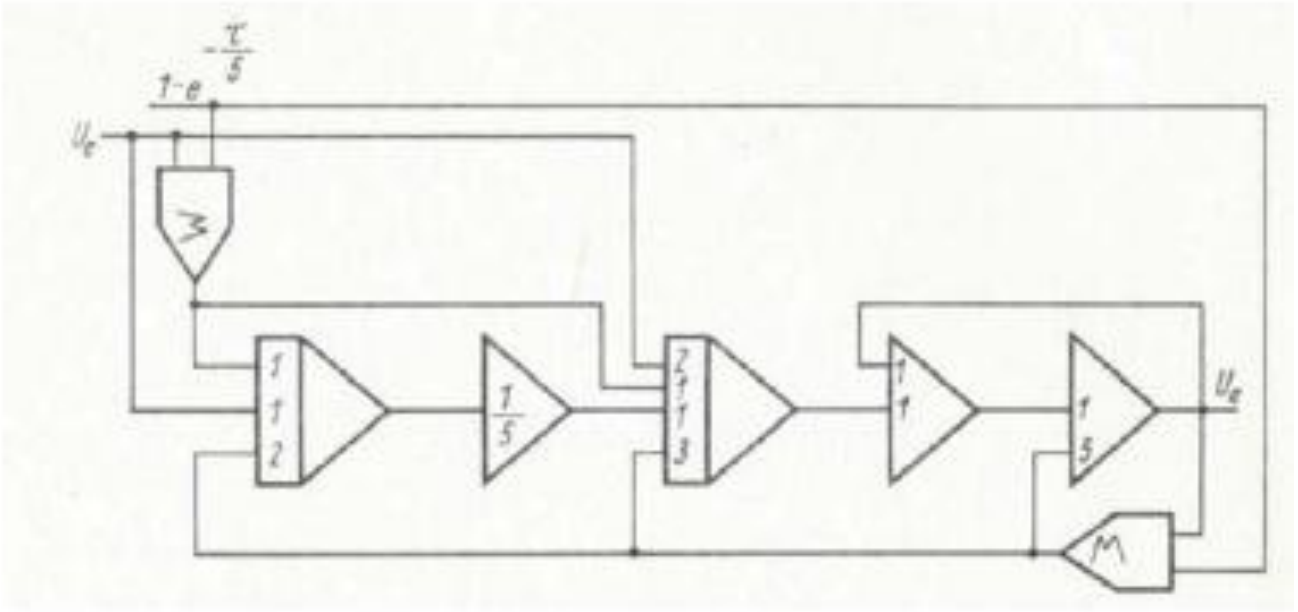
Anlogrechner MEDA-4



Rechenelemente eines Analogrechners

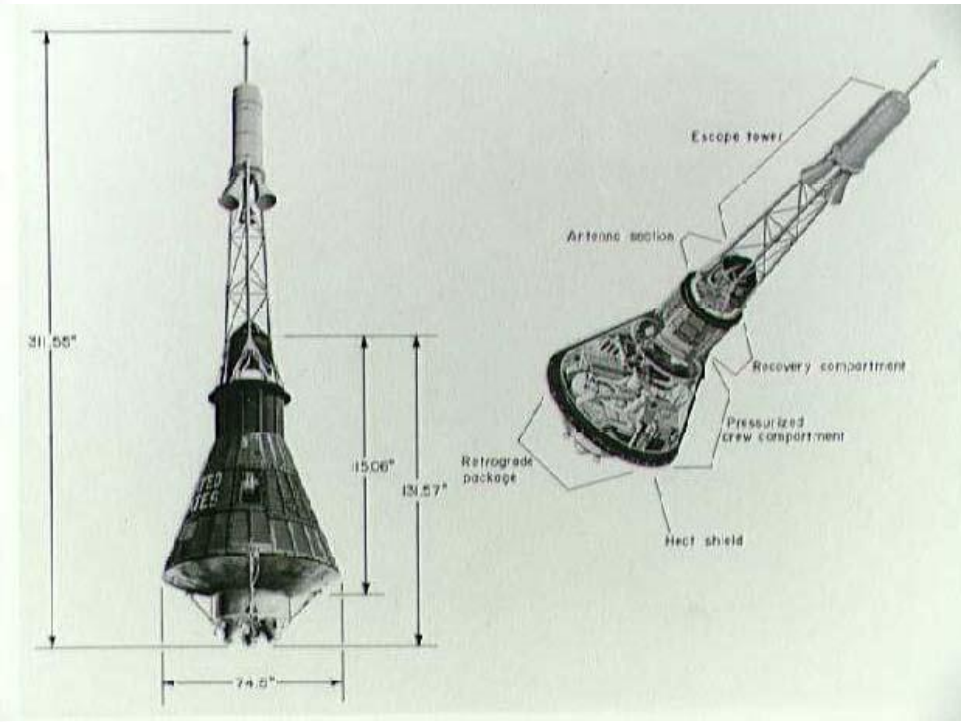






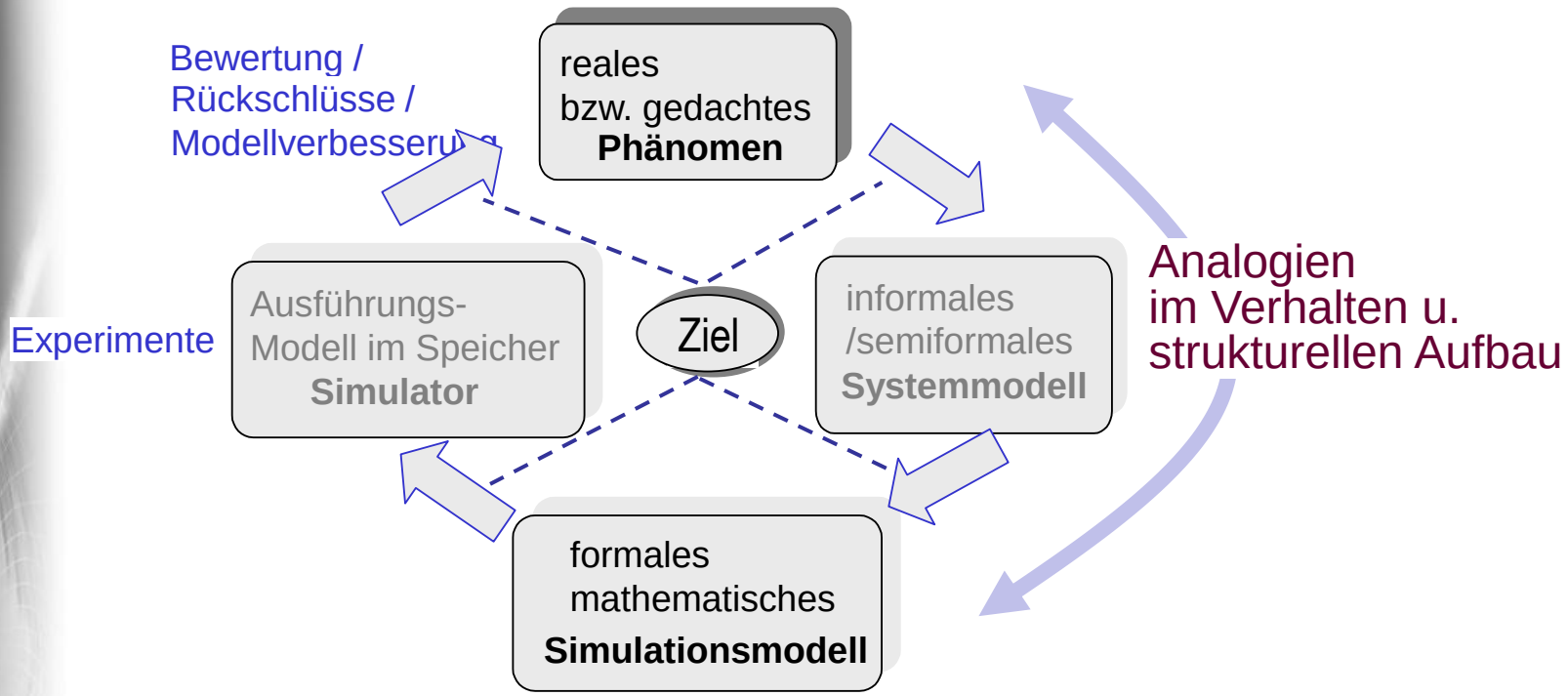
$$5\left(1-e^{-\frac{\tau}{5}}\right)\frac{\mathrm{d}^2x_a}{\mathrm{d}\tau^2}+\left(3-e^{-\frac{\tau}{5}}\right)\frac{\mathrm{d}x_a}{\mathrm{d}\tau}+\frac{2}{5}x_a=\left(3-e^{-\frac{\tau}{5}}\right)\frac{\mathrm{d}x_e}{\mathrm{d}\tau}+\frac{2}{5}x_e$$

Mercuri-Kapsel



Granino Arthur Korn
(Prof. für Elektrotechnik
University of Arizona)

Bedeutung von Analogien



1. Einführung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Bespiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Klassifikation dynamischer Systeme
8. M&S eines Niedertemperaturofens

Präzisere Begriffsbestimmung

Original

- Ausschnitt einer gedachten oder real existierenden Welt
  als System
  (Systemzweck, Abgrenzung zur Systemumgebung,
  Systemstruktur, Systemverhalten)

- Originale als statische oder dynamische **Systeme**

Modelle

- sind als Abstraktionen von Originalen
  Abstraktionen= Vereinfachungen aus einer bestimmten Sicht
  mit einer bestimmten Zielstellung

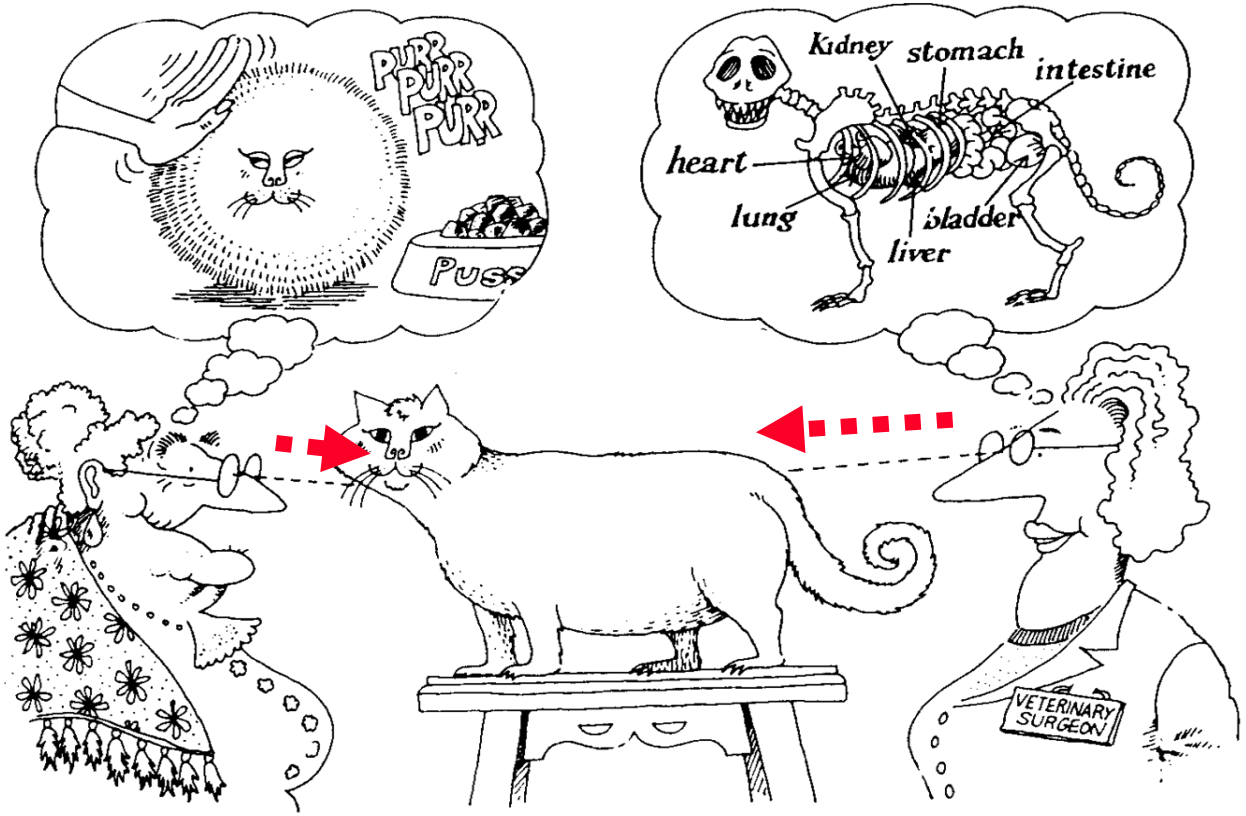
- Modelle sind Abstraktionen kompletter Systeme
  oder einzelner Systemelemente

- Systemmodelle= Struktur- und Verhaltensmodelle des Systems

Gibt es perfekte Modelle?

- Modelle werden aus einer bestimmten Sicht bei Verfolgung eines bestimmten Untersuchungsziels abgeleitet
- kein einziges Modell, keine einzige Sicht ist ausreichend um ein komplexes System zu erfassen
  → es gibt kein Modell an sich
- Entscheidung, welche Modelle erzeugt werden,
  hat großen Einfluss auf die Modelluntersuchung
- jedes Modell kann in unterschiedlichen Abstraktionsniveaus und aus unterschiedlichen Blickwinkeln dargestellt werden
- die besten Modelle sind realitätsnah
- **Gefahr**: bereits bewährte Modelle werden für Untersuchungen mit anderem Untersuchungsziel eingesetzt

Modelle in unterschiedlichen Sichten auf einen Realitätsausschnitt

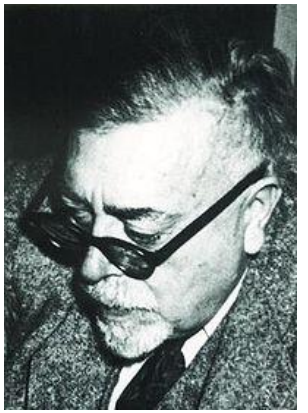


© Booch: Object-Oriented Design with applications

Benjamin/Cummings Publishing Company Inc.

Das jeweilige Untersuchungsziel bestimmt Sicht und Abstraktionsgrad

Begrenztheit von Modellen



Norbert Wiener (1894-1964)
Begründer der Kybernetik, Kommunikation, Steuerung und Regelung

Was ist das beste Modell einer Katze ?

scherzhaft: „Das beste Modell einer Katze ist ...

eine Katze.

Am besten dieselbe Katze.“

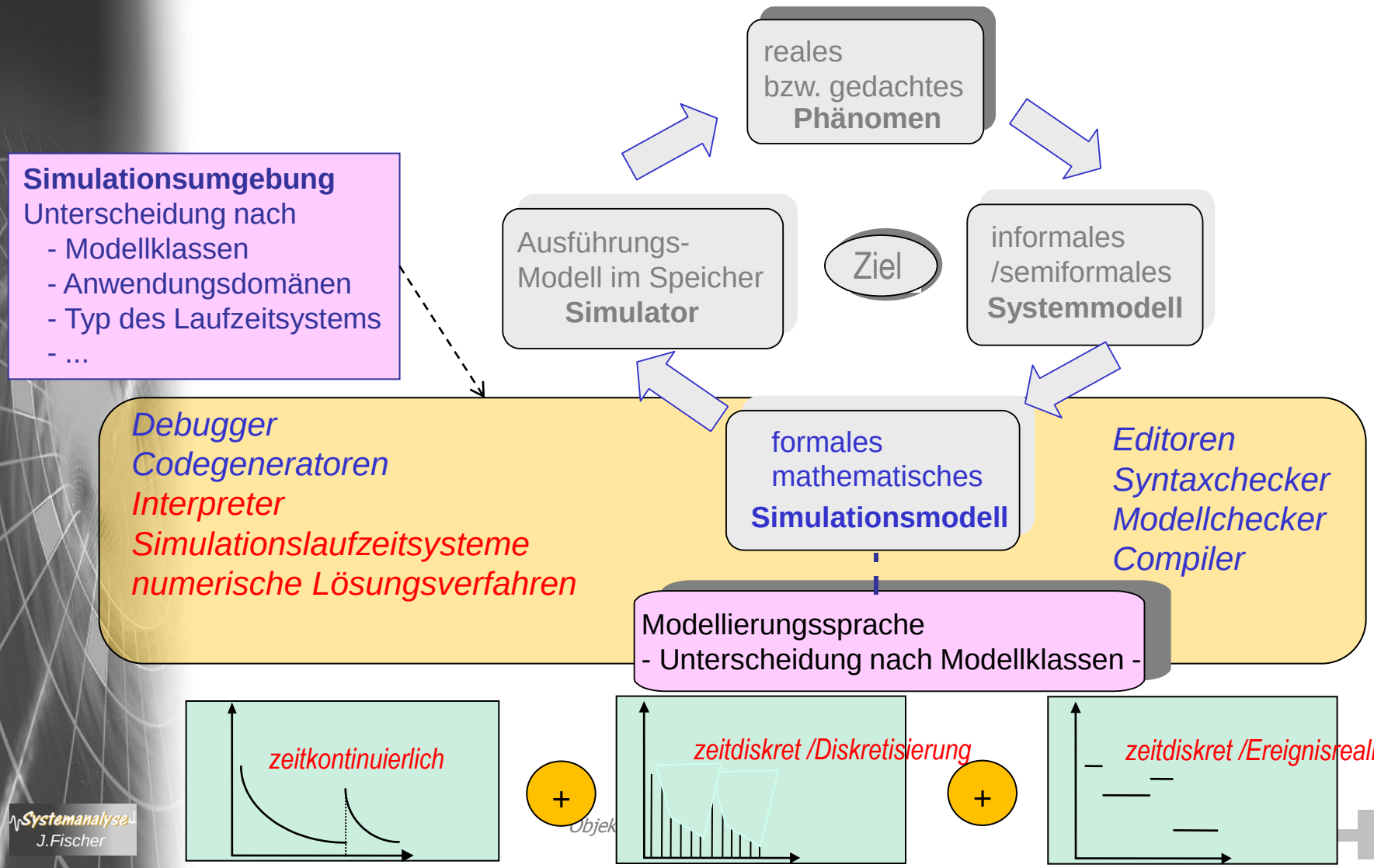
Modelle ersetzen ihre Originale nur bedingt

1. Einführung

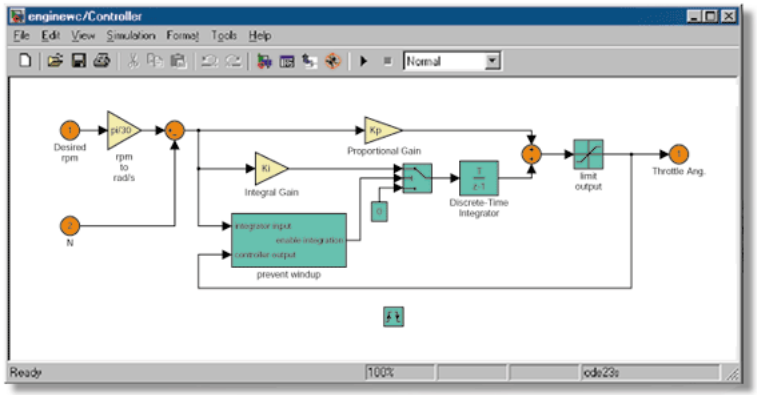
1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Bespiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Klassifikation dynamischer Systeme
8. M&S eines Niedertemperaturofens

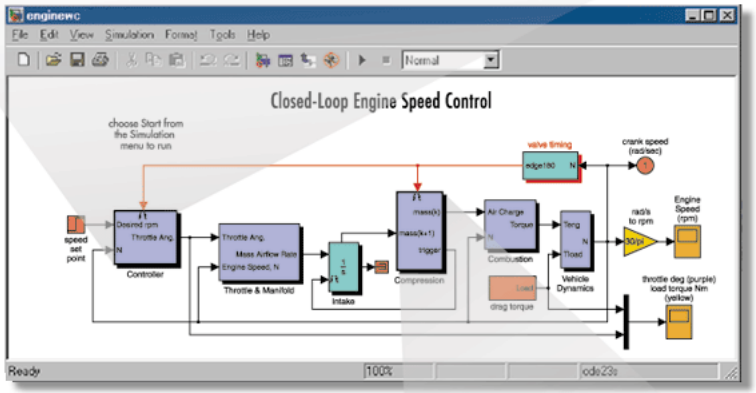
Modellierungssprachen und Simulationsumgebung

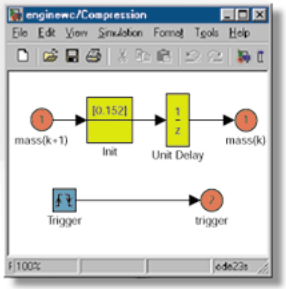
Zustandsänderungen kontinuierlich oder/und diskret in Raum und Zeit



Beispiele domänenspezifische Modellierungssprachen



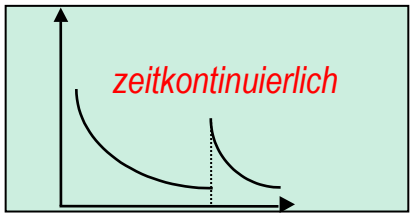




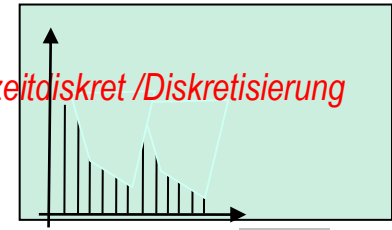
Simulink

- hierarchische graphische Modellierung
- kontinuierliche u. diskreter Schaltblöcke
- S-Functions: eigener Code u. MATLAB

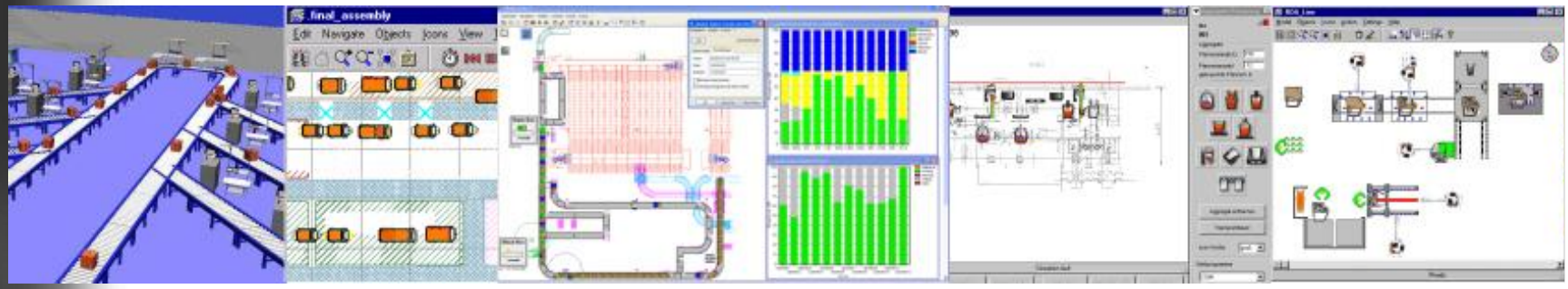
- für einzelne Domänen (wie mechanische, elektrische oder hydraulische Systeme) stehen spezielle Zusätze zur Verfügung, welche die Modellierung von physikalischen Systemen zusätzlich vereinfachen



+

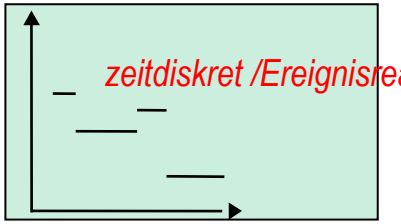


Beispiele domänenspezifische Modellierungssprachen



Plant Simulation

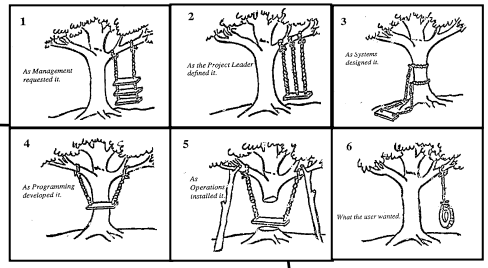
- graphische Modellierung, Simulation, Visualisierung
- Optimierung von Logistik- und Geschäftsprozessen



Solution Partner PLM SIEMENS

Frage:
Wie ist der Stand des Modelleinsatzes bei der Software-Produktion?

Modellbasierte Software-Entwicklung verteilter Systeme (vereinfacht)



Anforderungen

Anforderungs-modelle

Umgebungs-modelle

Analysemodelle

Entwurfsmodelle

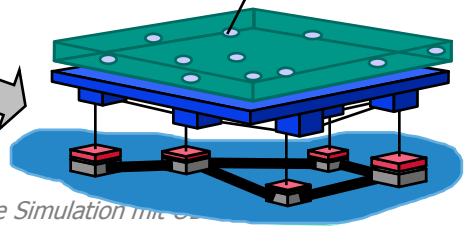
Implementie-rungsmodelle

Verteilungs-modelle

Plattform-modelle

Zielcode

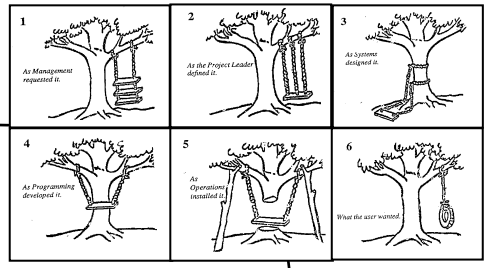
Binärcode-Komponenten



Modell-Checker
Simulator
Test, Wartung

Wasserfallmethode
später verbessert:
iterativ, inkrementell

Modellgetriebene Software-Entwicklung verteilter Systeme (vereinfacht)



Anforderungen

Anforderungs-modelle

Umgebungs-modelle

Analysemodelle

Entwurfsmodelle

Implementie-rungsmodelle

Verteilungs-modelle

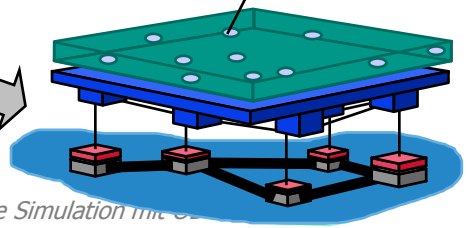
Plattform-modelle

Zielcode

Binärcode-Komponenten

Modell-Checker
Simulator
Test, Wartung

automatisierte Modelltransformation



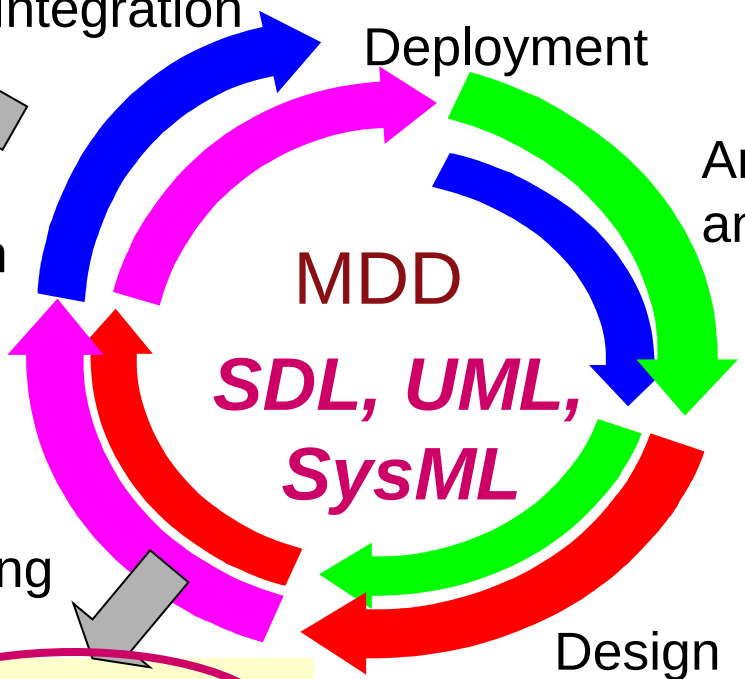
Modellgetriebene Software-Entwicklung

spiralförmig, inkrementell & iterativ

Test funktionaler und nicht-funktionaler Rückkkopplungen

Wechselwirkung mit der Umgebung mit Simulation

MDD:= Model Driven Development
- SW-Entwicklung ist modellzentriert (Modelle begleiten ges. SW-Lebenszyklus)
- automatische Transformationen für Modellübergänge
- spezifische Analysen (Checker, Simulatoren, …)
- partielle oder komplette Codegenerierung



Echtzeit, Leistungsprognose Ausführung durch Simulation

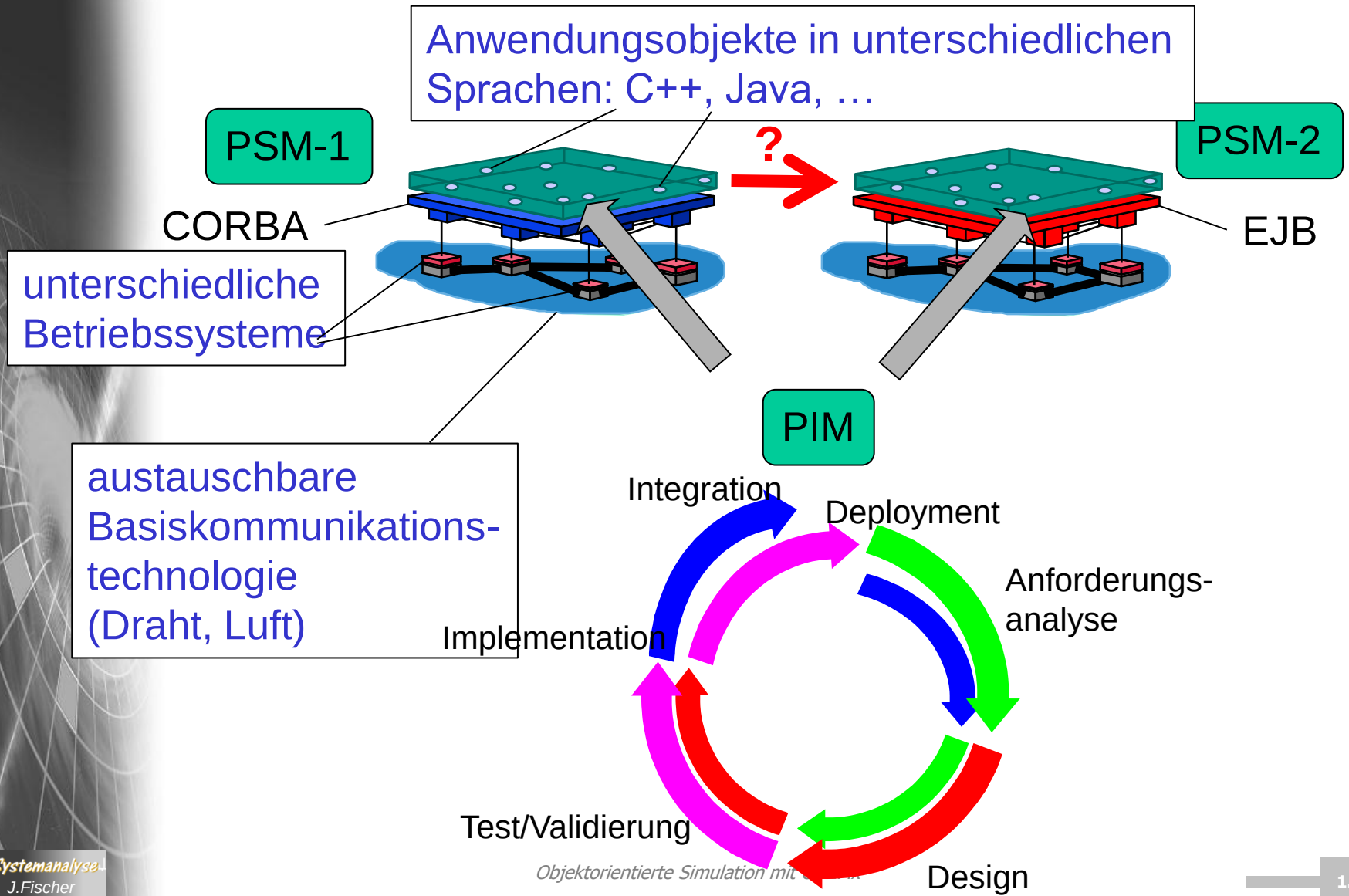
Test nicht-funktionaler Eigenschaften

Ausführung durch Simulation

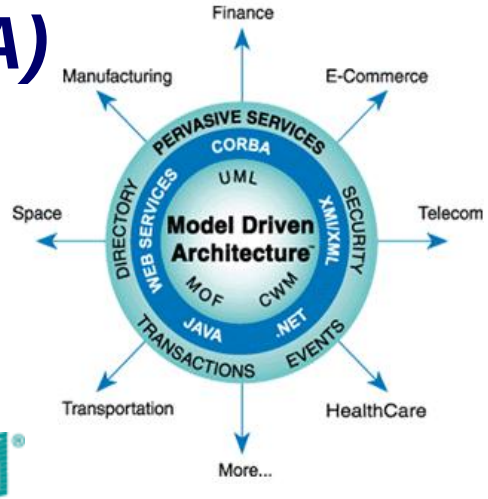
Test funktionaler Eigenschaften

Modellgetriebene Software-Entwicklung

spiralförmig, inkrementell & iterativ



Model-Driven Architecture (MDA)



- … fasst die gesammelten Erkenntnisse über
  - SW-Modelle, Modellierung und Transformation,
  - angereichert mit einer Reihe weiterer Standards

  zu einer offiziell anerkannten **Spezifikation** zur modellgetriebenen Softwareentwicklung zusammen

OMG
OBJECT MANAGEMENT GROUP

- **Ziel:**
  Abbildung des gesamter Softwareentwicklungsprozesses
  - von der Fachdomäne des späteren Anwenders,
  - über die Anforderungsanalyse
  - bis hin zur Implementierung des Zielsystems mit allen seinen Schichten)

  in Modellen,
  so dass das System selbst über Modelltransformation, erzeugt werden kann

- Sind alle Transformatoren geschrieben, so erreicht man auf diesem Weg eine hohe Wiederverwendbarkeit und Wartbarkeit

- Darüber hinaus gilt die MDA als ein möglicher Schlüssel zur anforderungsgetriebenen Softwareentwicklung, da die technischen Aspekte weitestgehend vollständig von den inhaltlichen (semantischen) Aspekten getrennt werden.

Spezielle Modelle der MDA

- Plattform Independent Models (PIM):
  die Modellierung der Fachdomäne (also der Zielwelt) ist **vollständig plattformunabhängig** zu gestalten,
  es sind also ausschließlich rein fachliche Aspekte zu betrachten und zu modellieren.

- **Plattform Description Models (PDMs)** sind (Meta-)Modelle,
  die die Zielplattform des Systems beschreiben.
  Über die Kombination von einem PIM, also einer formalen semantischen Beschreibung der Zusammenhänge und Abläufe mit einem PDM kann letztendlich über Modelltransformation das Zielsystem
  (welches im Sinne der MDA auch wieder nur ein Modell ist) generiert werden

- **Platform Specific Model (PSM)**
  ist das Ergebnis der Modelltransformation

Model-Driven Architecture (Leitsätze)

- **Formalisierung** ist ein wichtiger Baustein für ein erfolgreiches Qualitätsmanagement in Softwareprojekten.
  Speziell in den Bereichen der Anforderungs- und Systemanalyse besteht häufig noch ein hohes Optimierungspotential.

- Ein möglicher Weg, um den Formalisierungsgrad von Projektinformationen zu erhöhen, ist die Verwendung von **formal eindeutigen Modellen**.

  Für den erfolgreichen Einsatz von Modellen ist es jedoch unabdingbar, die **Syntax und die Semantik der Modelle über Metamodelle** exakt festzulegen.
  Ist dies einmal geschehen, ergibt sich meist eine deutliche Steigerung der Qualität wie auch der Effizienz in der Projektarbeit.

- Über den gezielten **Einsatz von Metamodellen** in der Softwareentwicklung können große Teile der Prozessaktivitäten automatisiert werden.

  Dennoch muss berücksichtigt werden, dass die Formalisierung eines Softwareentwicklungsprozesses **nicht in einem Schritt** erfolgen kann.
  Sie sollte vielmehr als ein **iterativer Prozess** verstanden werden, in dem die entstehenden Metamodelle von Projekt zu Projekt immer weiter verfeinert werden müssen.

Fazit

- Modellierung in allen Wissenschaftsdisziplinen das zentrale Paradigma zum Verständnis komplexer realer oder hypothetischer Systeme
  (auch in bestimmten Bereichen der Informatik)

- In der SW-Entwicklung lange Zeit nicht hoffähig :
  Alternative: von der Idee direkt zum gut dokumentierten Quellcode
  aber: Komplexität der Systeme bereiten praktische Probleme

- **Achtung**: MDD verlangt nicht nur Konzepte, sondern integrierte Werkzeugunterstützung

➔OMSI: Technologien zum Bau effizienter Simulatoren dynamischer Systeme