Projekt Erdbebenfrühwarnung
im WiSe 2010/11

Entwicklung verteilter eingebetteter Systeme

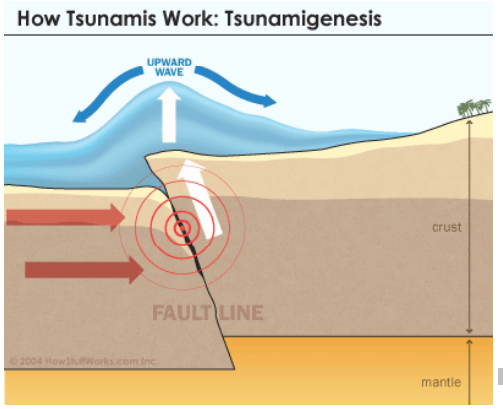
Prof. Dr. Joachim Fischer
Dipl.-Inf. Ingmar Eveslage
Dipl.-Inf. Frank Kühnlenz

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

Schlechte Presse: *Deutsches Tsunami-Warnsystem hat versagt*

28.10.2010
100 Millionen Euro hat das Frühwarnsystem gekostet. Vor dem neuen Tsunami hat es dennoch nicht gewarnt – es war nicht richtig gewartet worden.

300.000 €



GFZ verteidigt Frühwarnsystem

29.10.2010
ARD-Tagesschau

- Eine der beiden vor den indonesischen Mentawai-Inseln ausgebrachten Tsunami-Warnbojen funktionierte nicht.
- Es besteht derzeit kein Kontakt zur Boje

Dies hatte aber auf die Tsunami-Warnung keinen Einfluss: |P-Welle - S-Welle| =5 min
(vom Tsunami-Warnzentrum in Jakarta).

Lauterjung widerspricht der Darstellung indonesischer Behördenvertreter.

Die Inselkette vor Sumatra liegt so nah am Epizentrum des Bebens, so dass praktisch keine Vorwarnzeit bestand.
In weniger als zehn Minuten sei die Flutwelle auf die Inseln getroffen –
nicht ausreichend Zeit also, damit der Katastrophenschutz auf Sumatra die Bewohner zum Verlassen der Häuser und zur Flucht ins Inselinnere bewegen konnte.

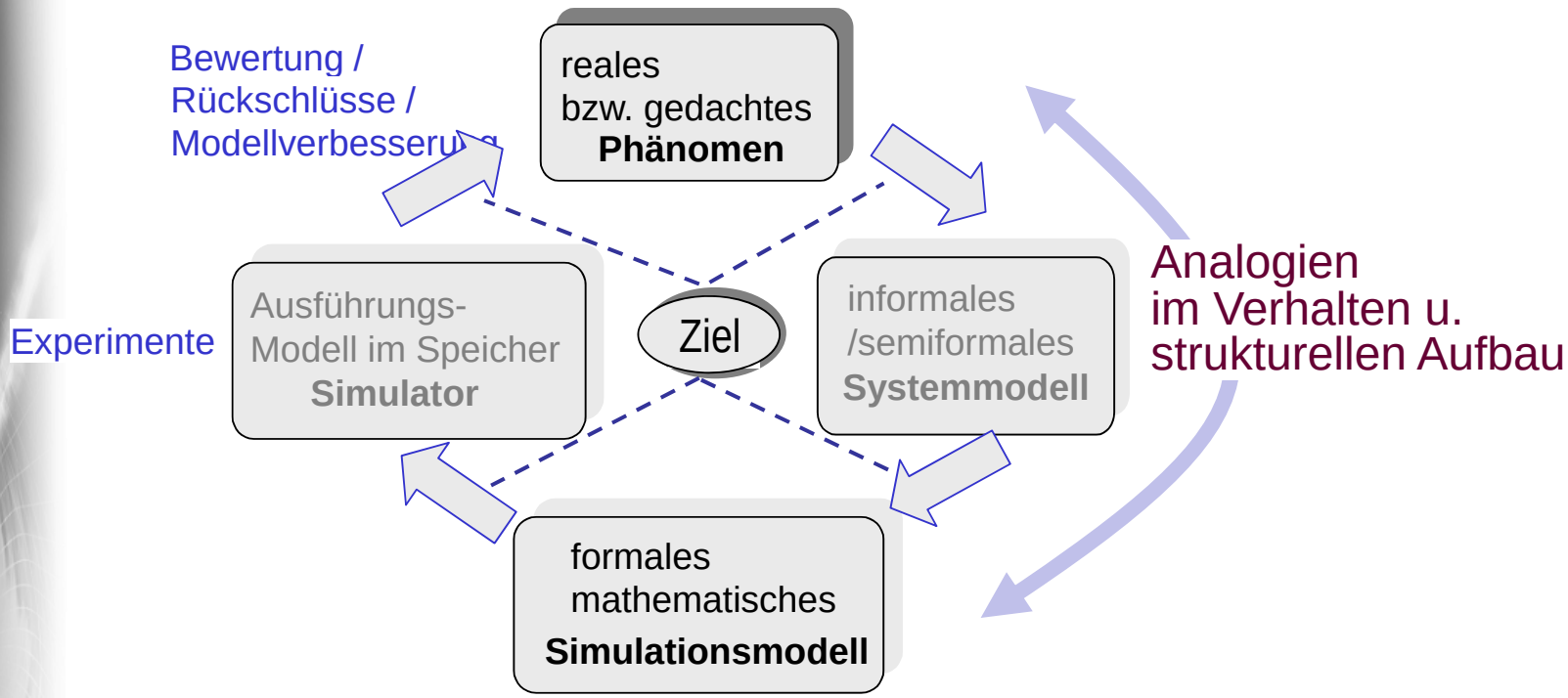
"Perfekt funktioniert"

"Das Tsunami-Frühwarnsystem hat perfekt funktioniert. Für die Bewohner der Mentawai-Inseln könne es keinen kompletten Schutz geben - die Inseln lägen nur 25 Kilometer von der tektonisch aktivsten Stelle des Pazifischen Feuerrings entfernt, dort wo häufig Erdbeben entstehen.
Wichtiger sei es da, dass die indonesischen Behörden mit den Inselbewohnern übten, die Warnsignale eines Erdbebens selbstständig richtig zu deuten.

3. Grundlagen der Systemmodellierung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Klassifikation dynamischer Systeme

Bedeutung von Analogien



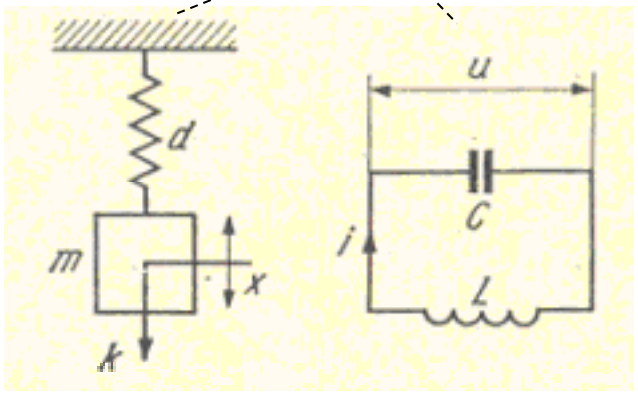
Besonderheit:
Zustandsgrößen ändern sich zeitabhängig
(kontinuierlich, diskret / ereignishaft)

➡ dynamische Systeme

Analogie im Systemverhalten

Basis für jede Verhaltensmodellierung

betrachten zwei Schwingungssysteme

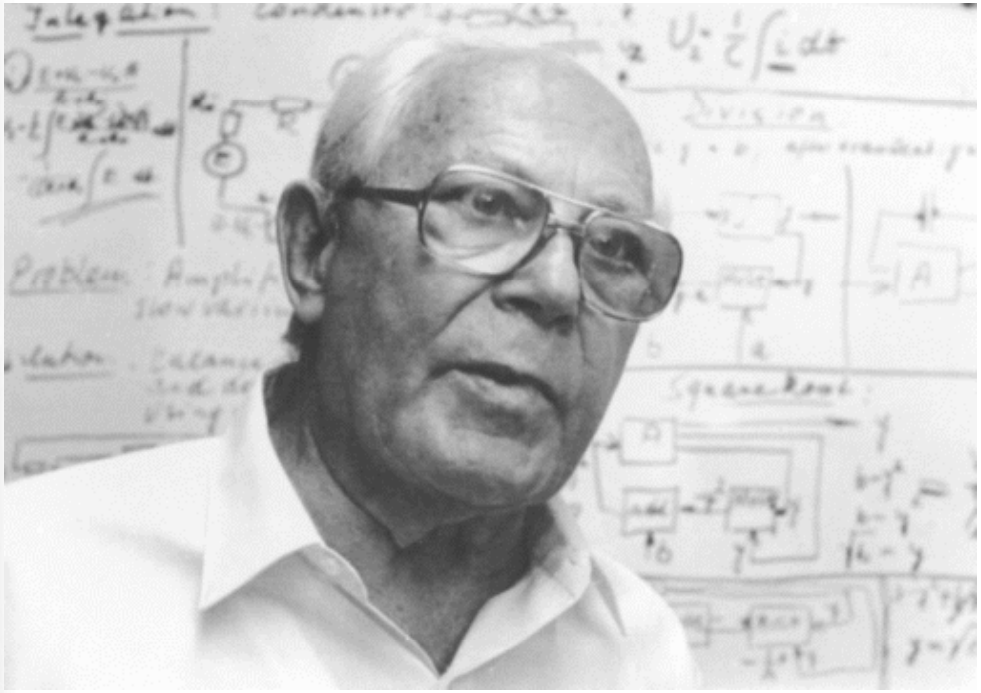


$$x \mathrel{\hat{=}} i$$
$$k \mathrel{\hat{=}} \frac{\mathrm{d}u}{\mathrm{d}t}$$
$$m \mathrel{\hat{=}} L$$
$$d \mathrel{\hat{=}} \frac{1}{C} \quad .$$

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)

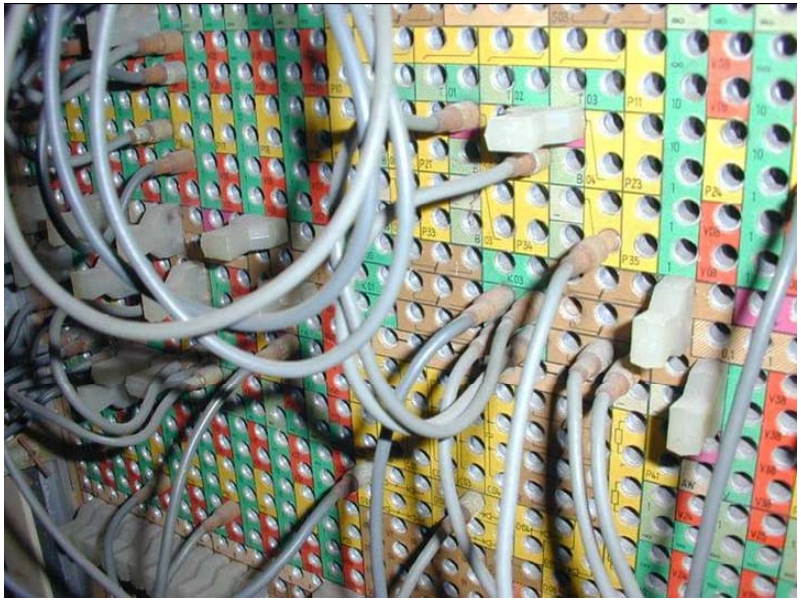


Erfinder des ersten frei programmierbaren Analogrechners (1941)

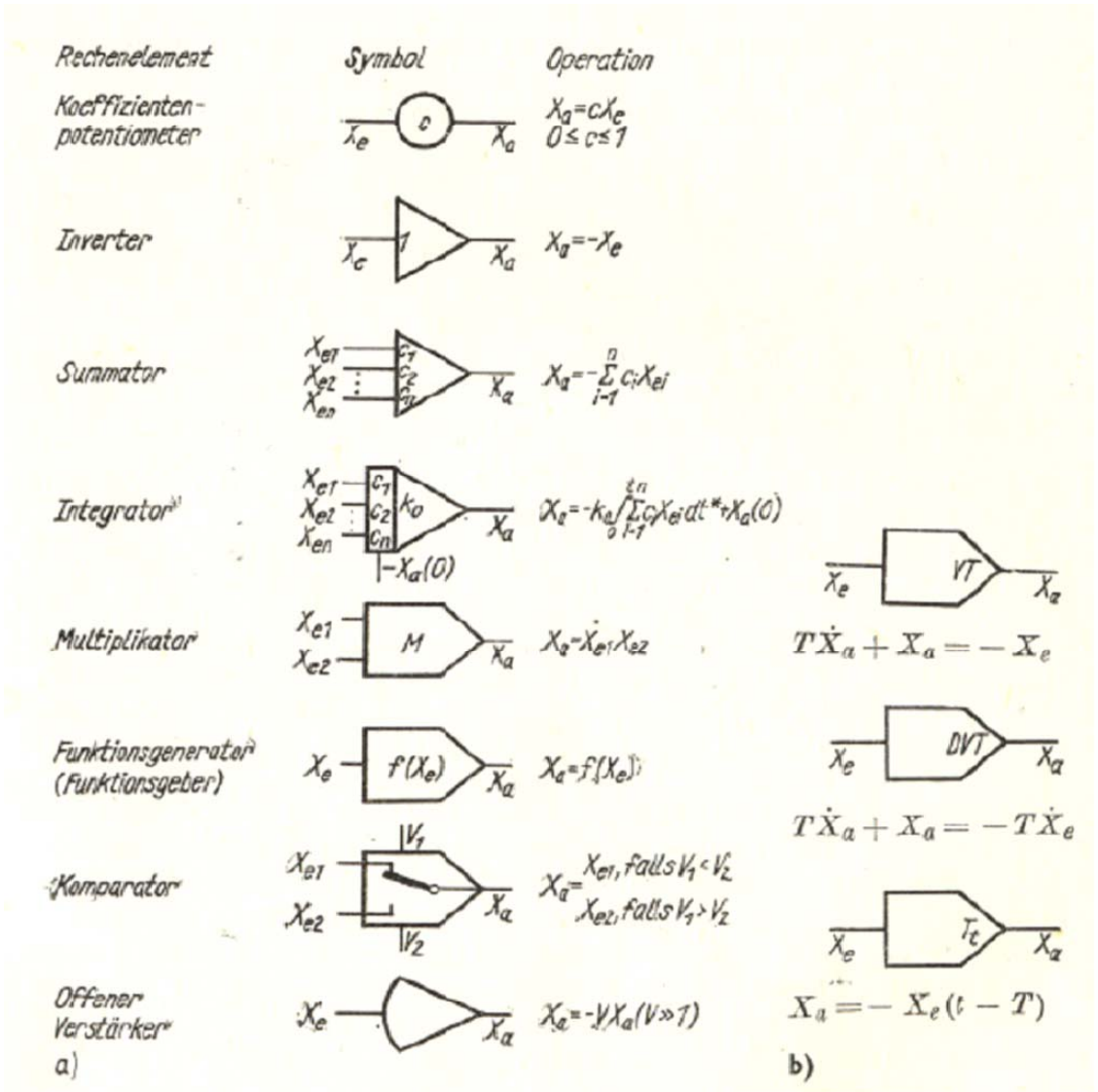
- TH Darmstadt (Diplom)
- Heeresversuchsanstalt Peenemünde (ab 1939)
- Marshal Space Flight Centre Huntsville (ab 1946)

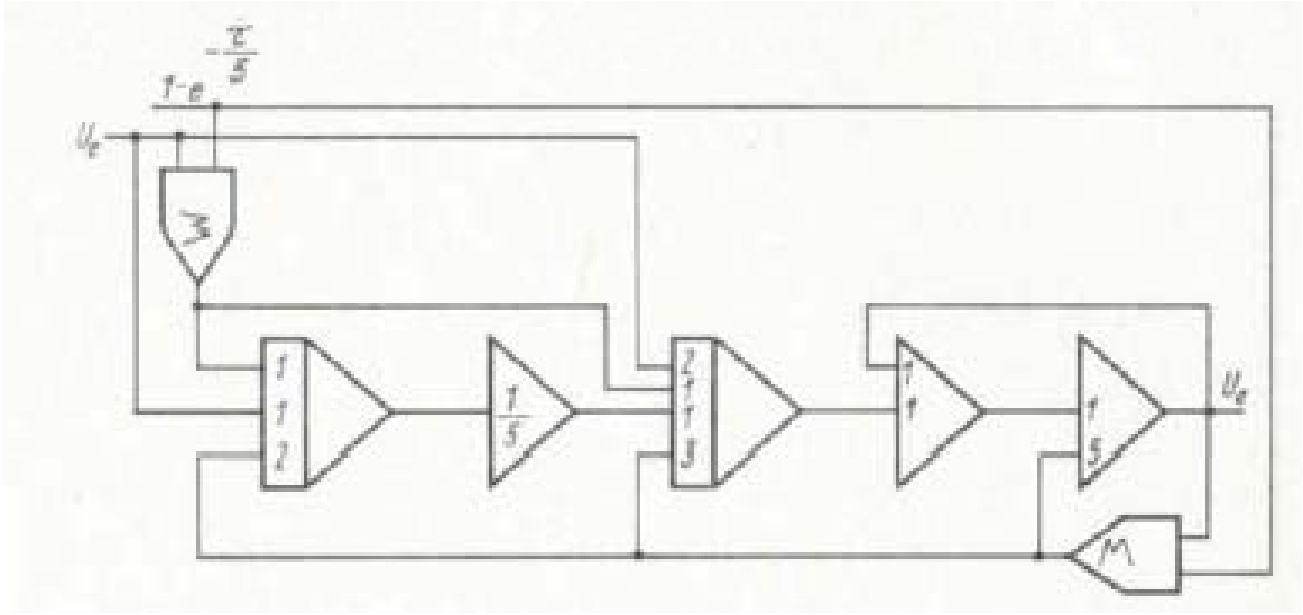
  … Appollo-Programm der Nasa

Anlogrechner MEDA-4



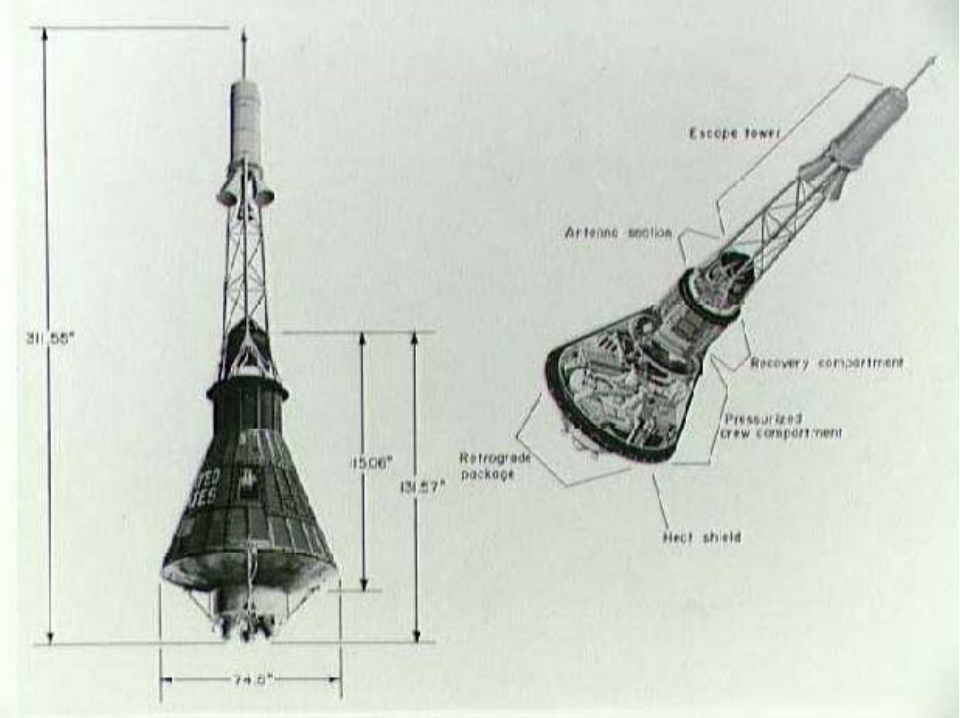
Rechenelemente eines Analogrechners





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

Mercuri-Kapsel



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

3. Grundlagen der Systemmodellierung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Klassifikation dynamischer Systeme

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 Vereinfachungen aus einer bestimmten Sicht mit einer bestimmten Zielstellung

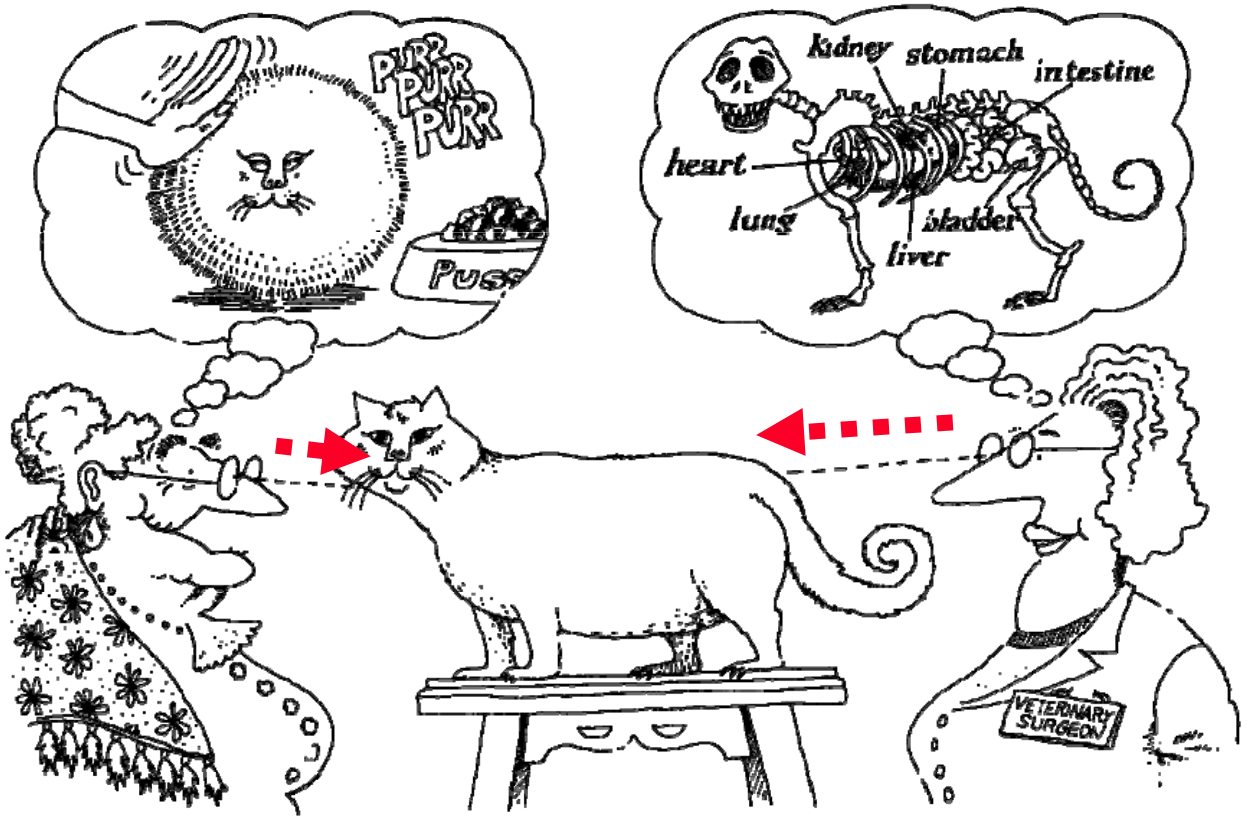
- Modelle sind Abstraktionen kompletter Systeme (Modellsystem) oder einzelner Systemelemente

- Struktur- und Verhaltensmodelle

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

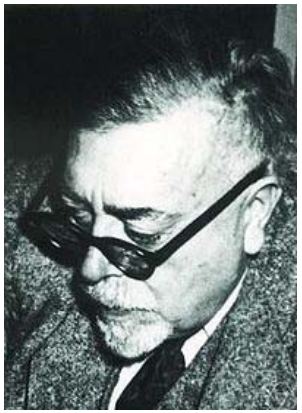
Modelle in unterschiedlichen Sichten auf einen Realitätsausschnitt



© 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

Was ist das beste Modell einer Katze ?

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

eine Katze.

Am besten dieselbe Katze.“

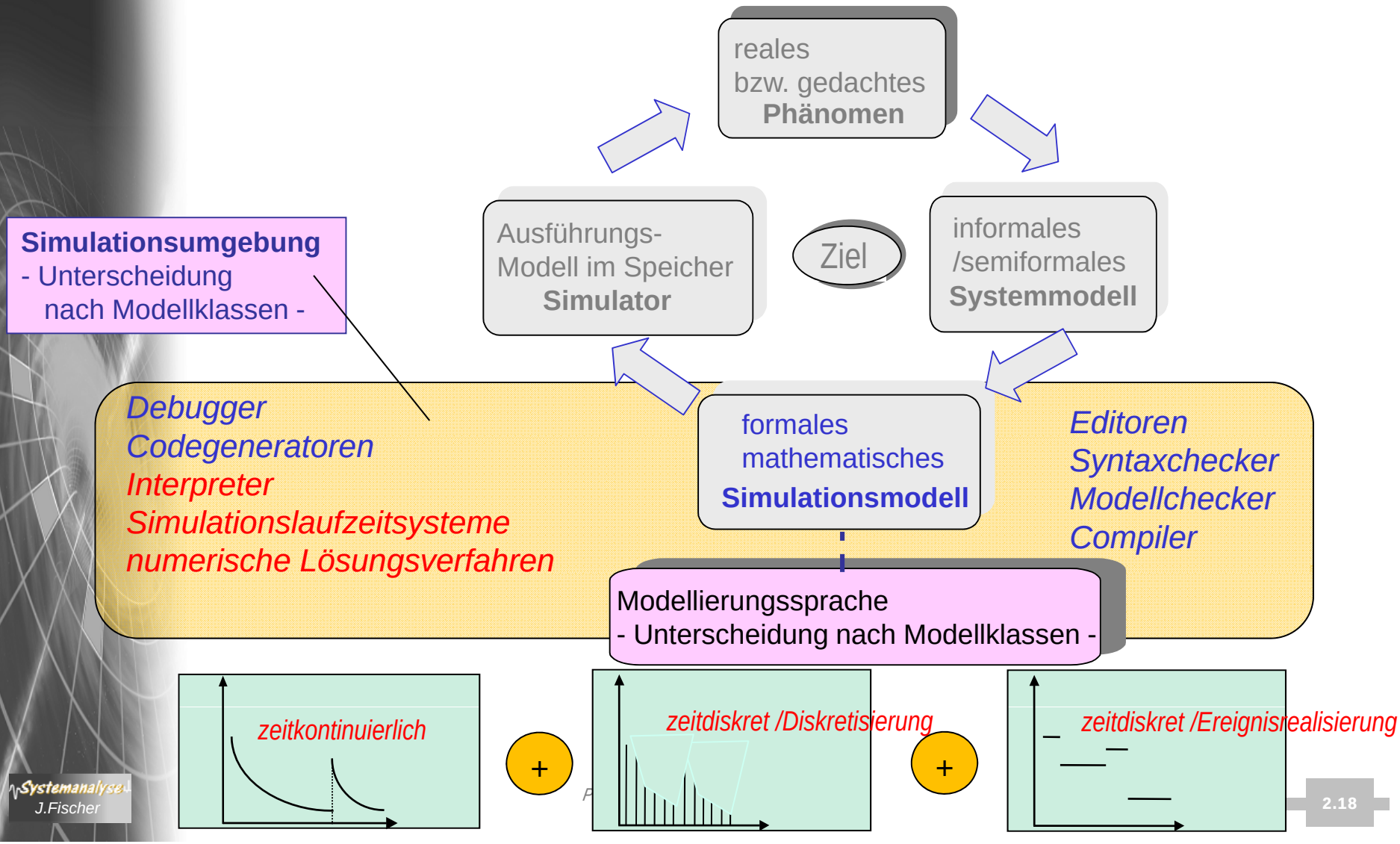
Modelle ersetzen ihre Originale nur bedingt

3. Grundlagen der Systemmodellierung

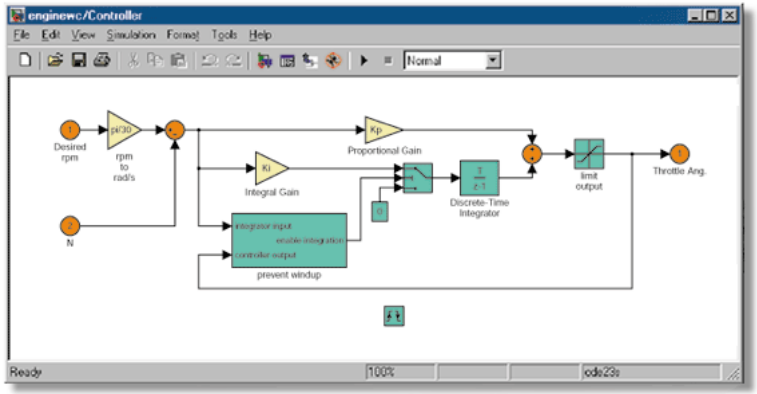
1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Klassifikation dynamischer Systeme

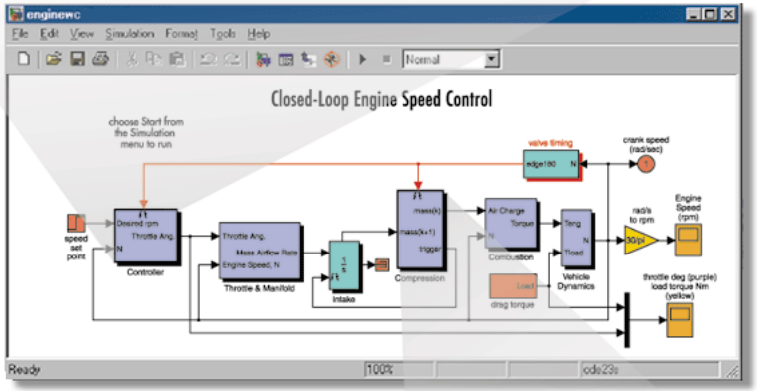
Modellierungssprachen und Simulationsumgebung

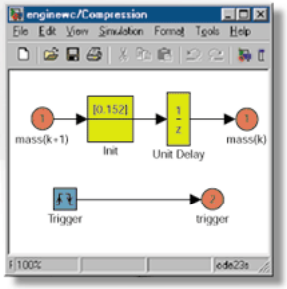
Zustandsänderungen kontinuierlich oder/und diskret in Raum und Zeit



Domänenspezifische Modellierungssprachen





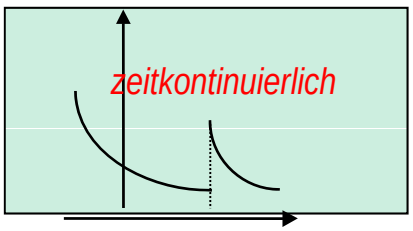


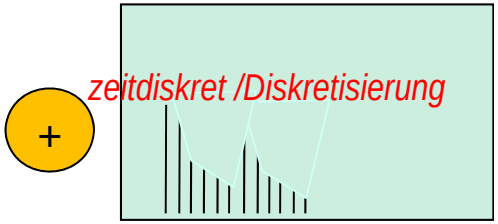
Simulink

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

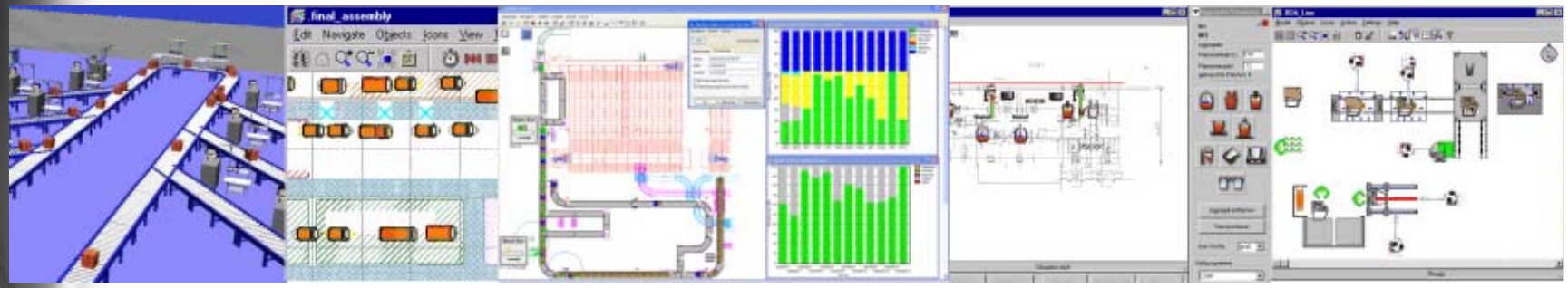
- für einzelne Domänen (wie mechanische, elektrische oder hydraulische Systeme) stehen spezielle Zusätze zur Verfügung, welche die Modellierung von physikalischen Systemen zusätzlich vereinfachen

➔dafür wurde das Konzept der unidirektionalen Signalverbindungen um bidirektionale logische Verbindungen - den sog. *physical networks* - erweitert.



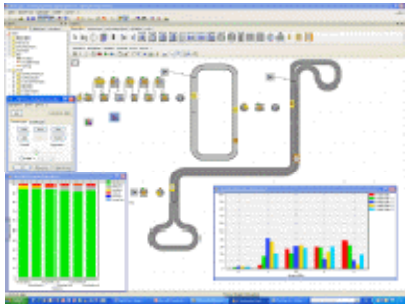


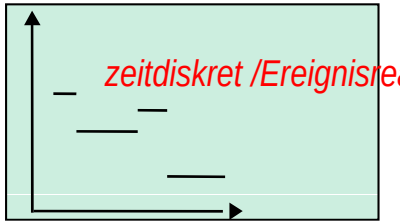
Domänenspezifische Modellierungssprachen



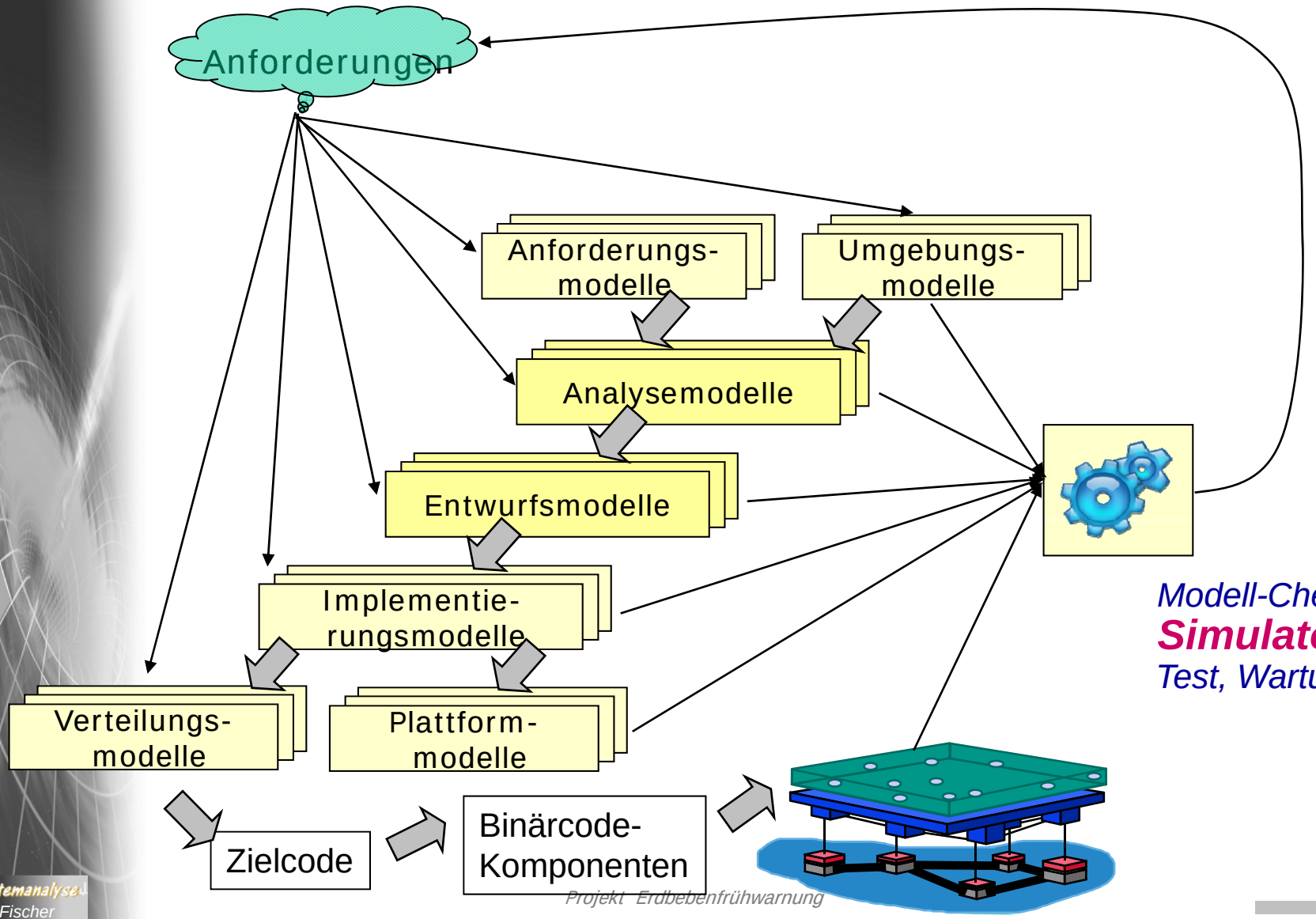
Plant Simulation

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





Modellbasierte Entwicklung (vereinfacht)



Modell-Checker
Simulator
Test, Wartung

Fazit

- Modellierung in allen Wissenschaftsdisziplinen das zentrale Paradigma zum Verständnis komplexer realer oder hypothetischer Systeme

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

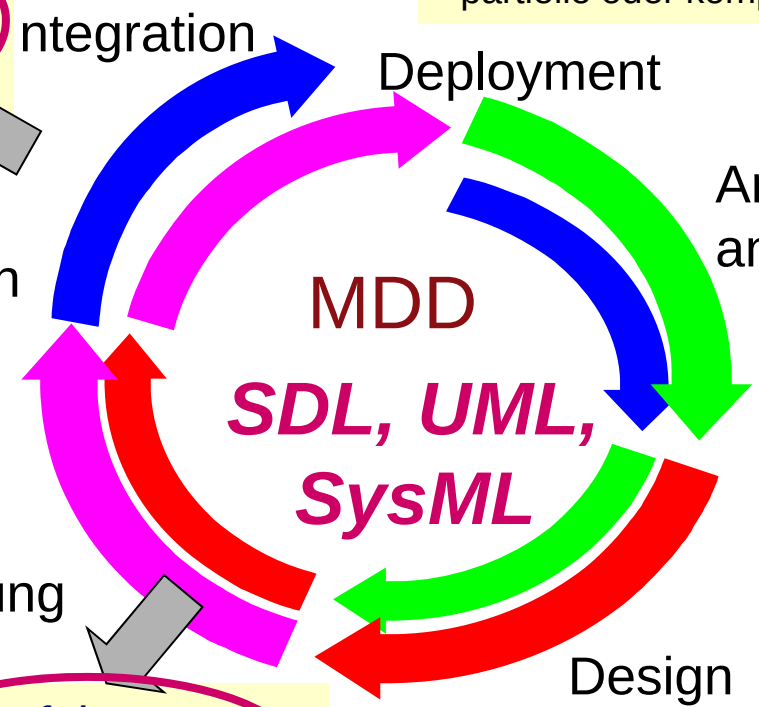
- **Achtung**: MDD→MDA verlangt nicht nur Konzepte, sondern integrierte Werkzeugunterstützung

SW-Entwicklungsprozess: *spiralförmig, inkrementell & iterativ*

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

Test funktionaler und nicht-funktionaler Rückkkopplungen

Wechselwirkung mit der Umgebung mit Simulation



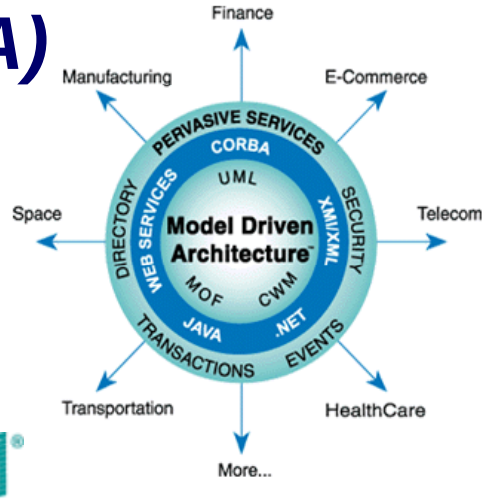
Echtzeit, Leistungsprognose Ausführung durch Simulation

Test nicht-funktionaler Eigenschaften

Ausführung durch Simulation

Test funktionaler Eigenschaften

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.

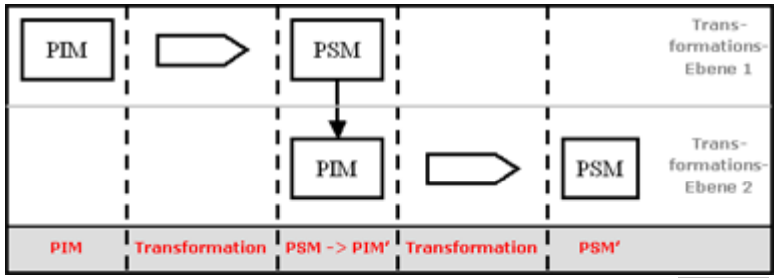
Spezielle Modelle der MDA

- 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 Metamodelle,
  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

So kann ein PSM auf einer höheren Ebene beispielsweise die Plattform einer objektorientierten Webanwendung auf J2EE-Basis beschreiben, eine konkretere Ebene würde hingegen möglicherweise den Einsatz einer bestimmten Middleware auf einen genau festgelegten Betriebssystem darstellen.



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.