

# ***Software-Engineering***

Sebastian Iwanowski  
FH Wedel

## **Kapitel 2: Grundbegriffe und Prinzipien**

# Grundbegriffe der Software-Entwicklung: Systeme

## System

- Ausschnitt aus der realen oder gedanklichen Welt
- besteht aus konkreten oder abstrakten Komponenten und deren Beziehungen untereinander

## Systemgrenze

- Festlegung, was zum System gehört und was nicht
- Interessierender Ausschnitt der (realen) Welt

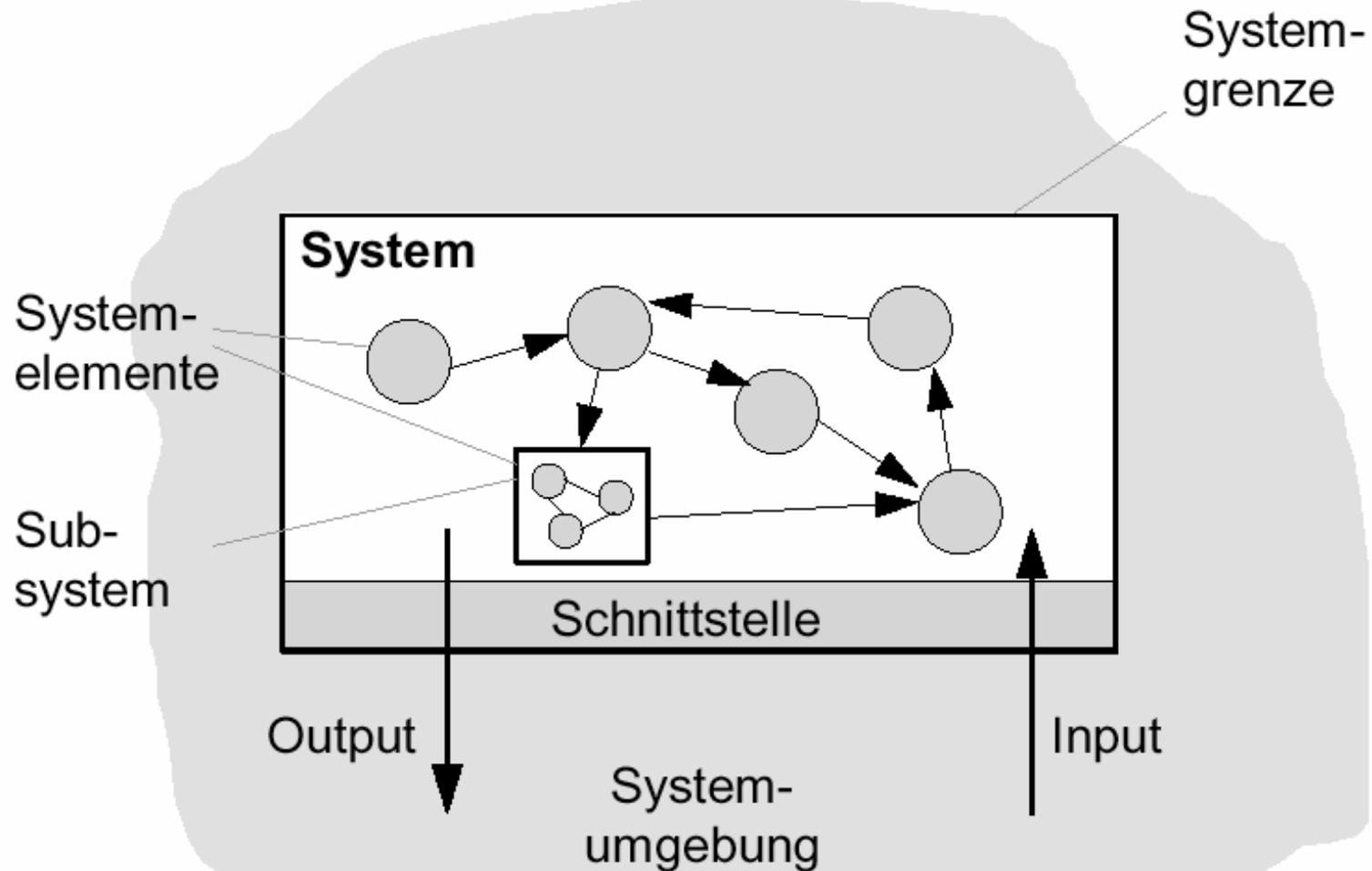
## Systemstruktur

- Stellung der Systemelemente zueinander, die sich durch die Beziehungen ergibt

## Systemhierarchie

- Elemente des Systems sind entweder atomar oder wiederum Systeme (Subsysteme)

# Grundbegriffe der Software-Entwicklung: Systeme



# Grundbegriffe der Software-Entwicklung: Systeme

## Systeminteraktion

- Austausch des Systems mit der Systemumgebung über die Schnittstelle durch Inputs und Outputs

## Komplexität von Systemen

- Strukturelles Merkmal, das von der Anzahl der Komponenten und der Verbindungen abhängt

## Systemzustand

- Belegung von Komponenten(teilen) mit Werten

## Dynamik von Systemen

- Veränderung des Systemzustands in Abhängigkeit von der Zeit (unabhängig von Eingriffen von außen)

## Statische Systeme

- Systemzustand ändert sich nur bei Eingriffen von außen, und auch das nur unmittelbar

# Grundbegriffe der Software-Entwicklung: Modelle

## Modelle

- sind Beschreibungen von Systemen
- dienen der Reduzierung der Systemkomplexität
- treten bei der Bearbeitung an die Stelle von Systemen
- dienen einem bestimmten Zweck
- können dasselbe System auf *unterschiedliche* Weise beschreiben

**Modelle sind die Voraussetzung für die Entwicklung von Software für  
die Systembearbeitung**

**Die Fähigkeit zur Bildung und kompetenten Handhabung von Modellen  
ist eine Schlüsselanforderung an Informatiker**

# Prinzipien des Software-Engineering

Die folgenden Prinzipien sind bei jeder Phase des Software-Engineering relevant

## Die drei Grundprinzipien:

- Abstraktion
- Zerlegung
- Perspektivenbildung

## Weitere wichtige Prinzipien für die Softwareentwicklung

- Integrierte Dokumentation
- Mehrfachverwendbarkeit
- Standardisierung

# Das Prinzip der Abstraktion

## Vorgehen der Abstraktion

- Abstraktion verallgemeinert durch Vernachlässigen von Eigenschaften
- Abstraktion trennt Wesentliches vom Unwesentlichen
- Was "unwichtig" ist, ist im Einzelfall schwer zu entscheiden und hängt von der Problemstellung ab
- Abstraktion ist insbesondere in den frühen Phasen der Software-Entwicklung von Bedeutung

# Das Prinzip der Abstraktion

## Vorteile der Abstraktion

- Durch Weglassen "unwichtiger" Eigenschaften kann die Systemkomplexität reduziert werden.
- Abstraktion erreicht in der Regel eine größere Allgemeingültigkeit.

## Nachteile der Abstraktion

- Manchmal werden leider doch wichtige Dinge weggelassen.
- Modellbildung und Modellverständnis setzt Übung voraus.

# Das Prinzip der Zerlegung

## Vorgehen der Zerlegung

- Zerlegung grenzt Bestandteile vom Ganzen ab.
- Zerlegung betrachtet und bearbeitet die Bestandteile getrennt.
- Zerlegung strukturiert das System.
- Verschiedene Zerlegungstypen: Hierarchisierung, Modularisierung.

## Vorteile

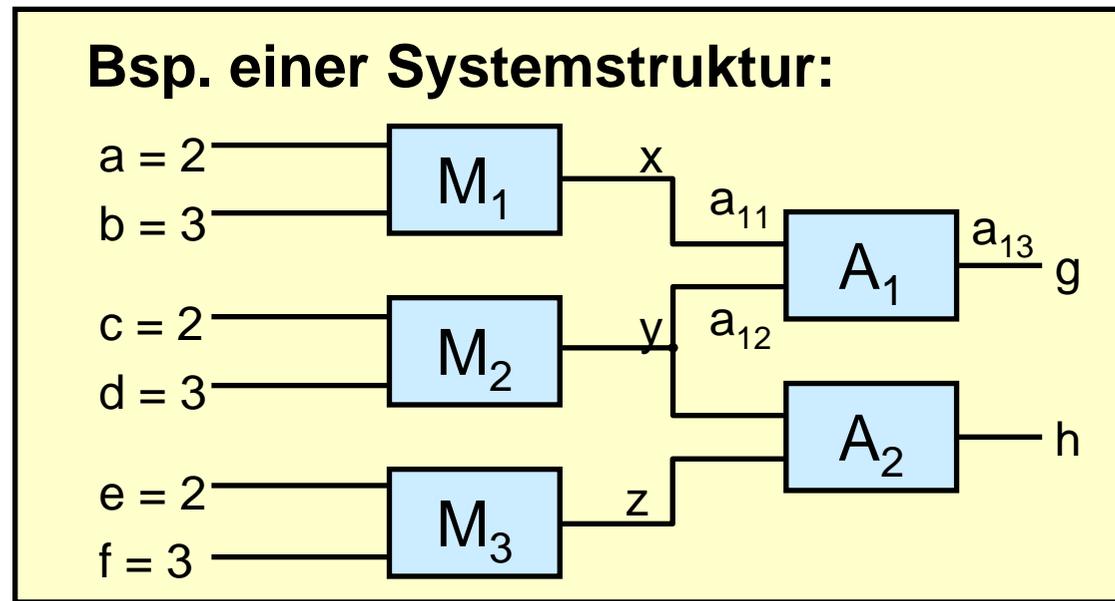
- Komplexität der Bestandteile geringer als die des Ganzen: besseres Verständnis, weniger Fehler
- Arbeitsteilige, parallele Bearbeitung der Bestandteile möglich
- Bessere Änderungsfähigkeit, Wiederverwendbarkeit

# Das Prinzip der Zerlegung

## Die Zerlegungstypen: Hierarchisierung und Modularisierung

### Modularisierung

- Schaffung von funktional abgeschlossenen Bausteinen auf einer Hierarchiestufe
- Festlegung der Schnittstellen zwischen den Modulen

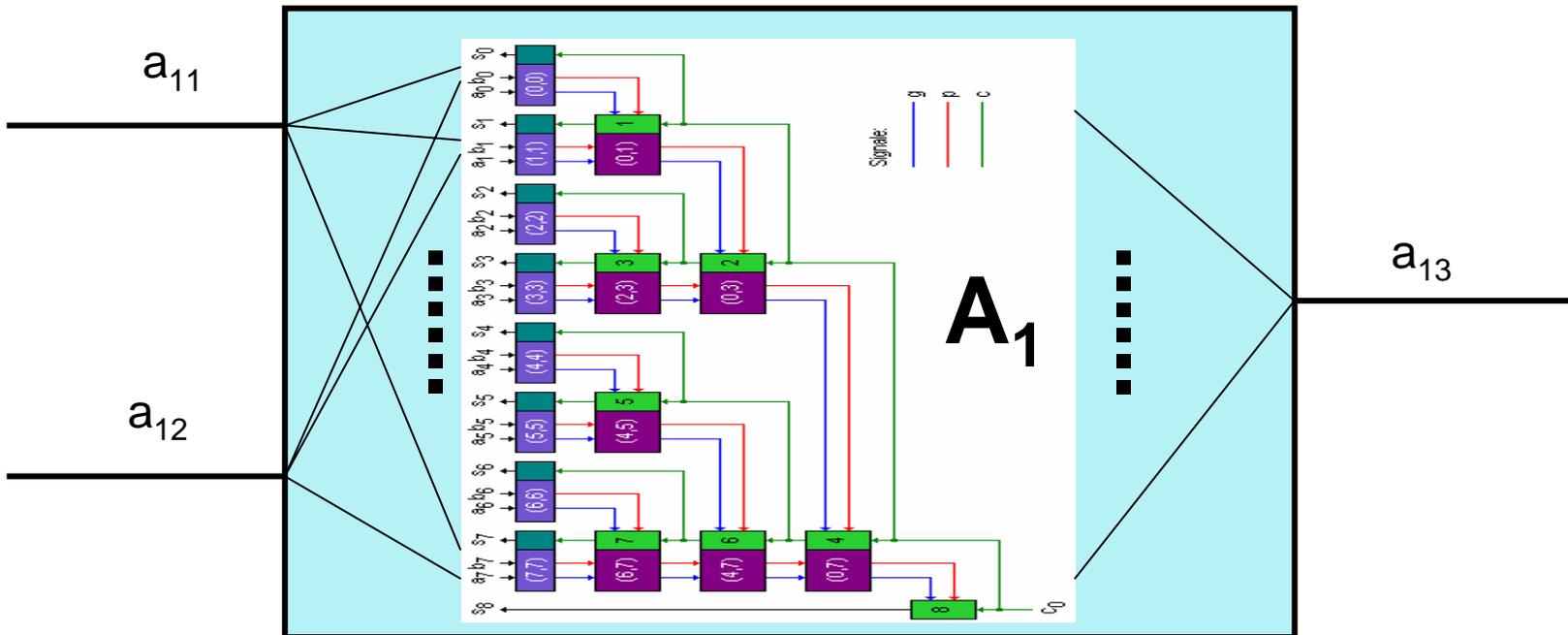


# Das Prinzip der Zerlegung

## Die Zerlegungstypen: Hierarchisierung und Modularisierung

### Hierarchisierung

- Anordnung von Elementen nach einer Rangordnung
- Elemente gleicher Rangordnung bilden eine Hierarchiestufe



# Das Prinzip der Perspektivenbildung

## Vorgehen

- Separate Analyse desselben Sachverhaltes unter verschiedenen *Sichten*
- Einnehmen unterschiedlicher Perspektiven stellt große Anforderung an Systementwickler

## Vorteile

- Berücksichtigung aller relevanten Aspekte einer Problemstellung
- Weniger Fehler in frühen Phasen der Software-Entwicklung (Analyse)

# Das Prinzip der Perspektivenbildung

## Die verschiedenen Sichten der Perspektivenbildung:

### Funktionsorientierte Sicht (datenflußorientiert)

- z.B. HIPO (Hierarchy of Input-Process-Output) der IBM (1965-)
- z.B. SA (Structured Analysis) von DeMarco (1979-)

### Datenorientierte Sicht

- z.B. ERM (Entity-Relationship-Model) von Chen (1976-)

### Objektorientierte Sicht

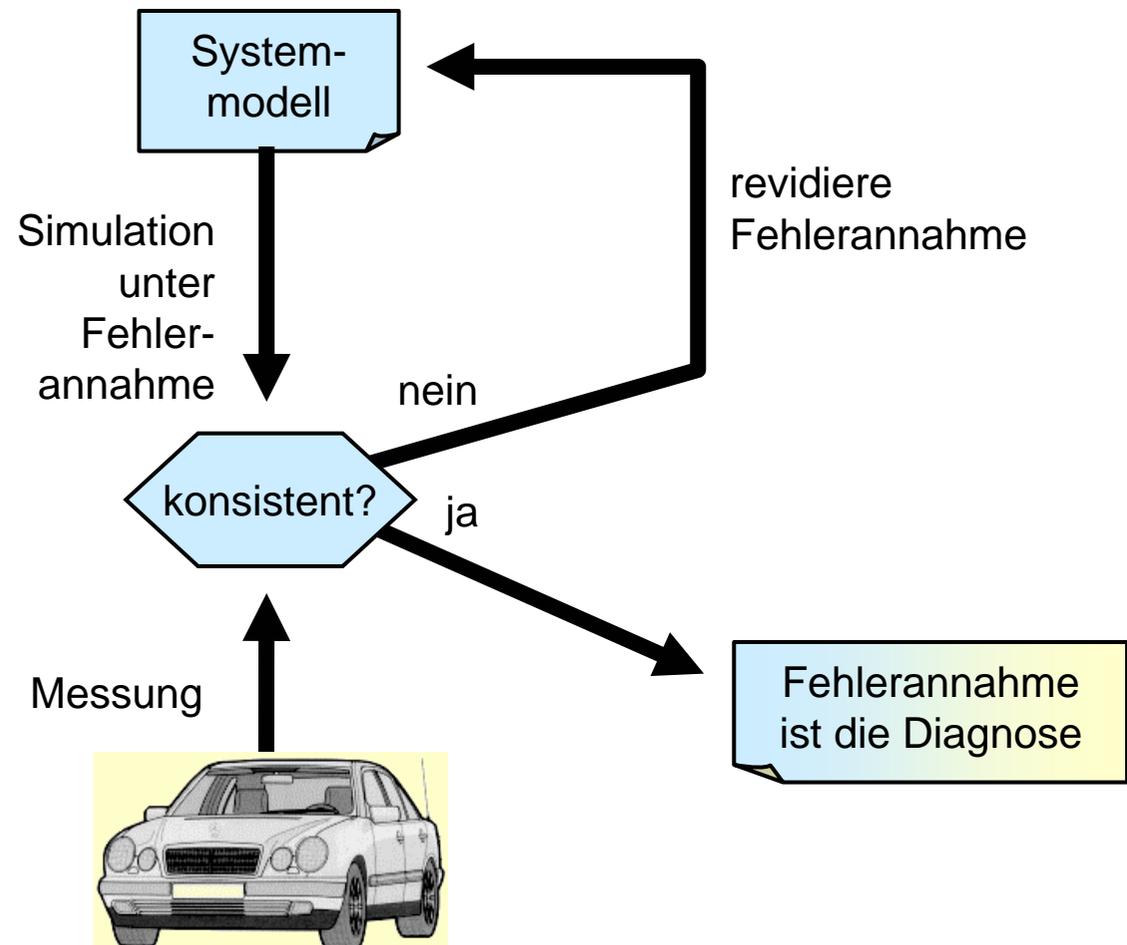
- z.B. UML (Unified Modeling Language) von Booch, Rumbaugh, Jacobson (91-)

### Geschäftsprozessorientierte Sicht

- z.B. EPK (Ereignisorientierte Prozessketten)

# Bsp.: Diagnose aus datenflussorientierter Sicht

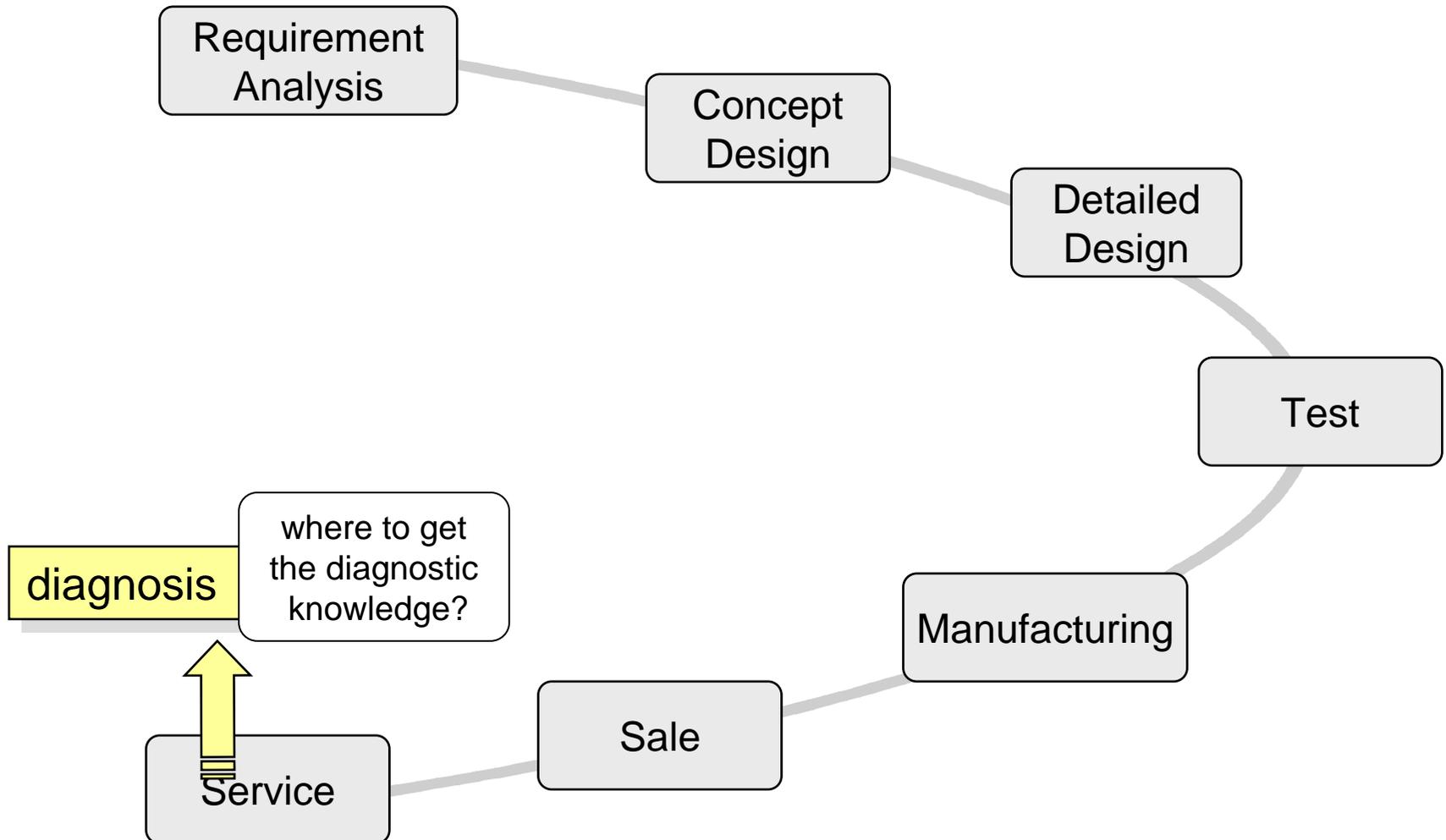
## Betrachtung der Steuerung für die Diagnoseberechnung



**Bsp.:**

# Diagnose aus geschäftsprozessorientierter Sicht

**Betrachtung der Prozesskette für die Diagnoseanforderungen**



# **Zusammenfassung Grundbegriffe und Prinzipien**

**System**

**Modell**

**Abstraktion**

**Zerlegung**

**Perspektivenbildung (Einnehmen verschiedener Sichten)**