

III. Entwicklung von Informationssystemen

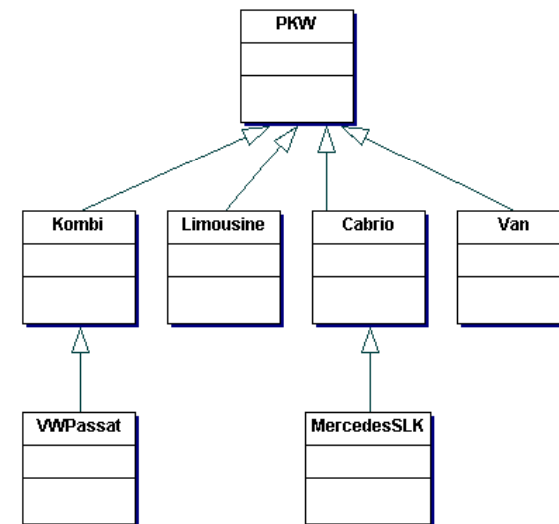
Kapitel 2 Modellierung von Informationssystemen

1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- Zerlegung eines Systems in einzelne Bestandteile (Objekte), die aus Daten und Operationen bestehen und miteinander interagieren können
- Zusammengenommen beschreiben die Daten und Operationen den Zustand und das Verhalten des Objekts.
- Objekte haben einen bestimmten Typ (Klasse).



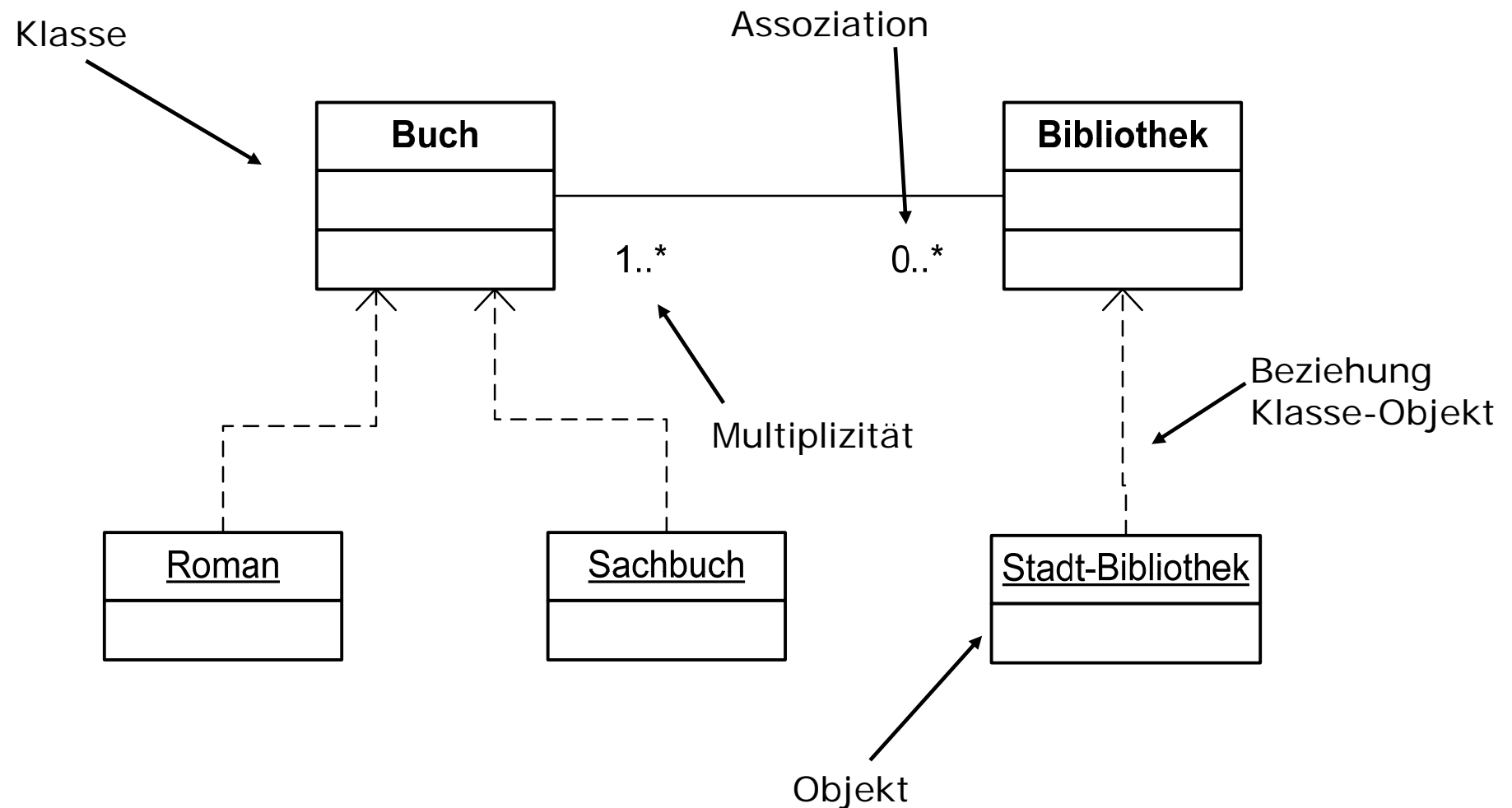
- Informationskapselung
 - Daten werden im Objekt gespeichert und sind von außen nur über die angebotenen Operationen zugreifbar, was ungewollte Manipulation verhindert.
- Austauschbarkeit
 - Durch Ersetzbarkeit einzelner Klassen
- Wiederverwendung
 - Durch Verwendung bereits existierender Klassen oder durch Vererbung
- Erweiterbarkeit
 - Durch Vererbung oder einfaches Hinzufügen zusätzlicher Funktionalität
- Verständlichkeit
 - Orientierung an der realen Umwelt (Objekte)
- Durchgängigkeit
 - Durchgängigkeit der Methoden von der Analyse bis zur Codierung

- Klassen
 - Klassen sind Konstruktionsvorschriften für Objekte. Sie bestehen aus Variablen, Konstanten und Methoden.

- Assoziationen
 - Beziehungen zwischen Klassen gleichen Ranges

- Objekte
 - Objekte sind Instanzen (Ausprägungen) von Klassen. Für eine Klasse können mehrere Objekte existieren.

- Instanziierung
 - Erzeugung von Objekten nach den Vorschriften einer Klasse



- Vererbung
 - Von Vererbung spricht man, wenn eine Klasse Eigenschaften und Verhaltensmerkmale ihrer Oberklasse erbt.

- Nachrichten
 - Nachrichten an Objekte weisen diese zum Aufruf ihrer Operationen an.

- Polymorphismus
 - Wenn eine Nachricht an Objekte von verschiedenen Klassen verschickt wird, dann liefern diese Objekte unterschiedliche Ergebnisse zurück.
 - Bsp.: Nachricht „Drucke“ an die Objekte „Adressliste“ und „Auftrag“

- Vielzahl verschiedener objektorientierter Ansätze unterschiedlicher Autoren
 - OOSE (Jacobsen)
 - OMT (Rumbaugh)
 - OOA (Coad/Yourdon)
 - ...

- Probleme durch unterschiedliche Ansätze
 - Schwierige Zusammenarbeit im Team, da unterschiedliche Analysten/Designer/Entwickler i.d.R mit nur einem bestimmten Ansatz vertraut sind.
 - Erschwerte Nutzung von Referenzmodellen
 - ...

- Bedürfnis nach einem gemeinsamen Standard

1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

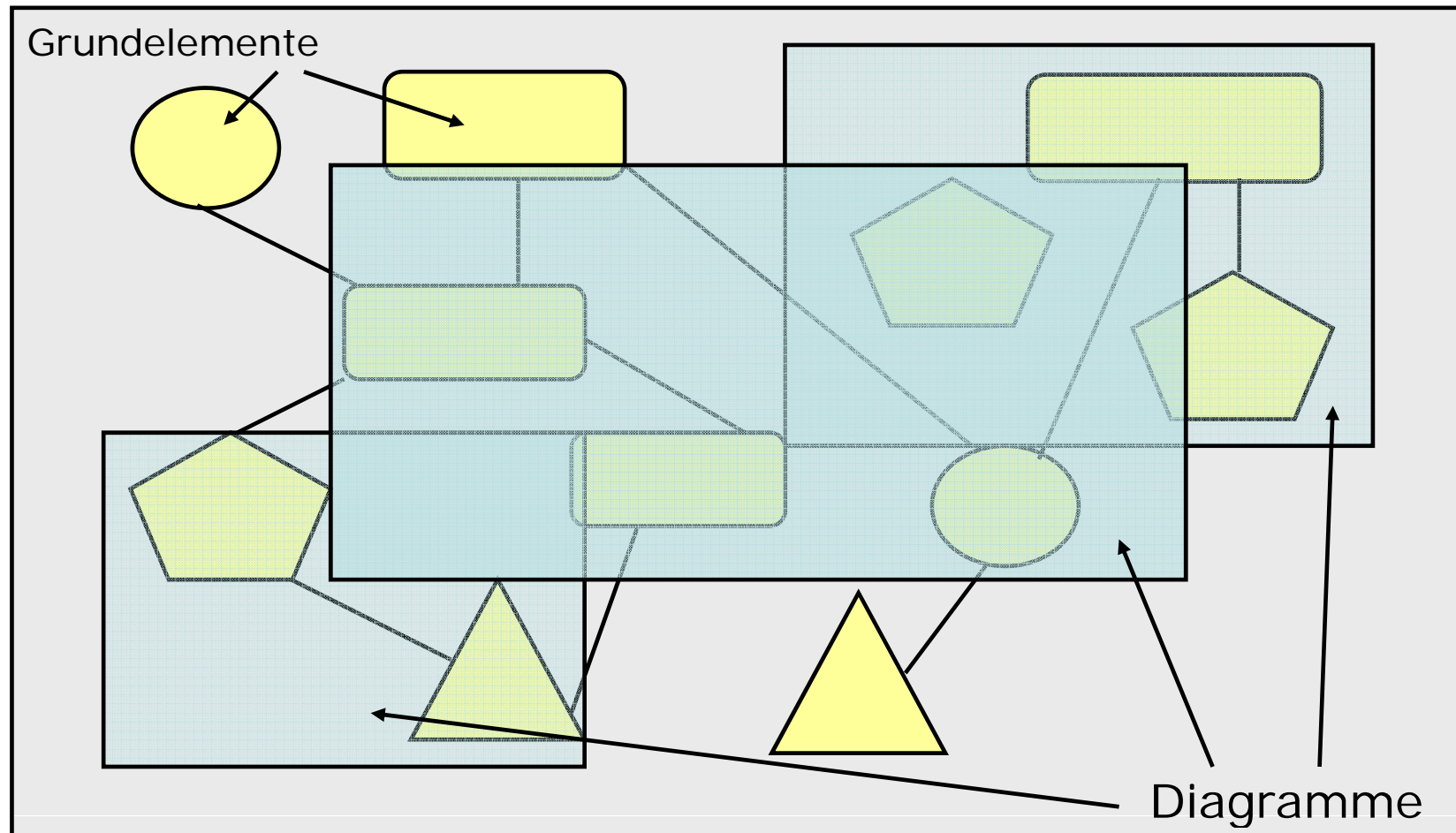
3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- 1996 von Booch, Jacobson und Rumbaugh entwickelte Modellierungssprache.
- Standard der OMG (Object Management Group), aktuell: UML Version 2.0
- Erreichte Vereinheitlichung
 - von verschiedenen objektorientierten Notationen
 - der Methodik durch alle Phasen der Softwareentwicklung
 - der verschiedenen Modellierungssichten (datenorientiert, funktionsorientiert, etc.)

- UML definiert
 - Diagrammarten für alle Phasen der Entwicklung von der Analyse bis zur Implementierung.
 - Notationselemente für die jeweiligen Diagrammarten.

- UML ist kein Vorgehensmodell
 - D.h. UML definiert keine Reihenfolge für die Erstellung von Diagrammen.
 - Entwickler sind angehalten, eigene, problembezogene Vorgehensmodelle zu entwickeln und zu nutzen.

- Grundelemente
 - Notationselemente aus der Objektorientierung
 - Weitere Elemente für die Darstellung eines zu modellierenden Systems (z.B. Aktivitäten, Akteure, etc.)
- Diagramme
 - Zusammensetzung aus bestimmten Grundelementen
 - Stellen bestimmte Eigenschaften eines Modells dar
- Gesamtmodell
 - Das Gesamtmodell setzt sich aus Grundelementen zusammen.
 - Unterschiedliche Sichten auf das Gesamtmodell durch verschiedene Diagrammtypen

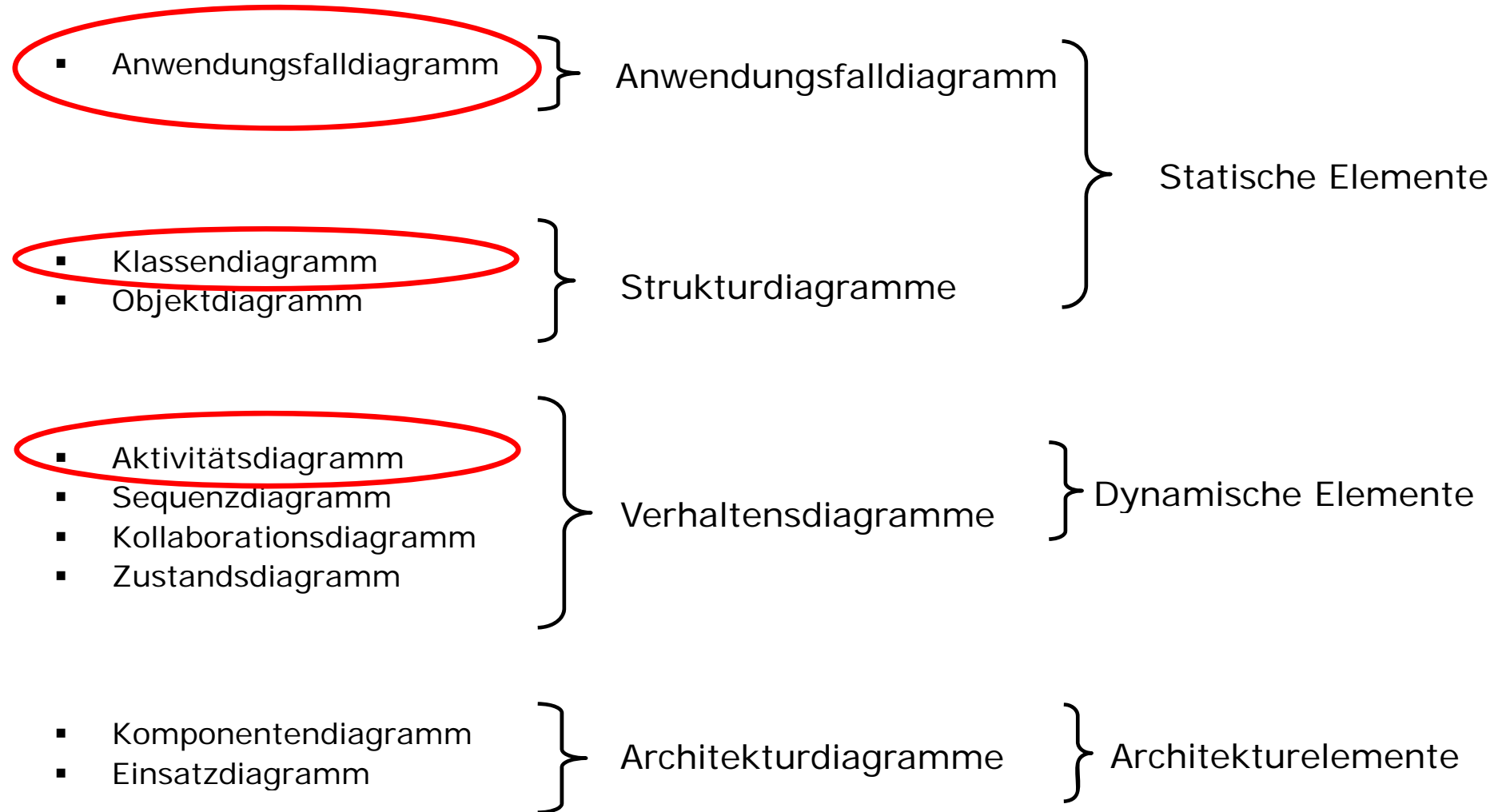


Gesamtmodell

1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

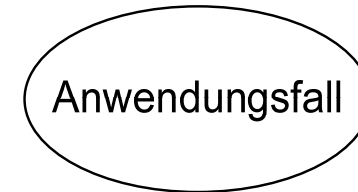
2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

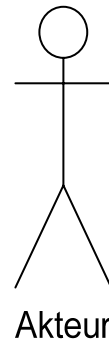


- Anwendungsfälle beschreiben die Aufgaben, die ein System leisten soll.
- Die Summe aller Anwendungsfälle ergibt die fachlichen Anforderungen an ein System.
- Definition der Schnittstelle zwischen Anwender und System
- Spezifikation mit Fachabteilung/Auftraggeber

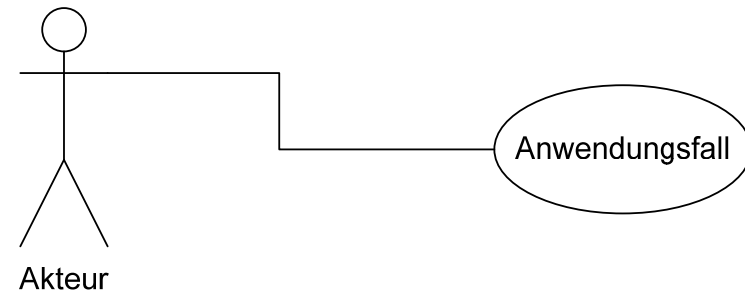
- Anwendungsfall
 - Darstellung eines Arbeitsschritts im System



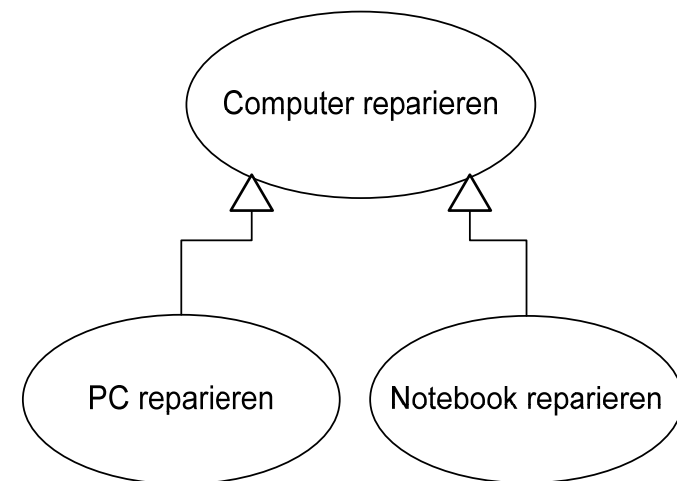
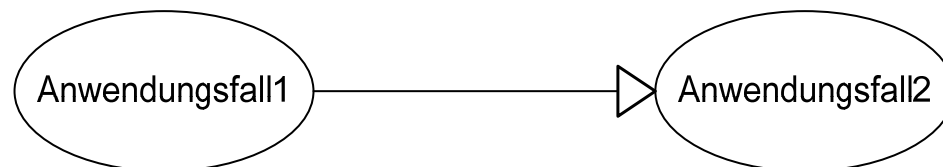
- Beteiligter am System



- Assoziation
 - Beteiligung eines Akteurs an einem Anwendungsfall

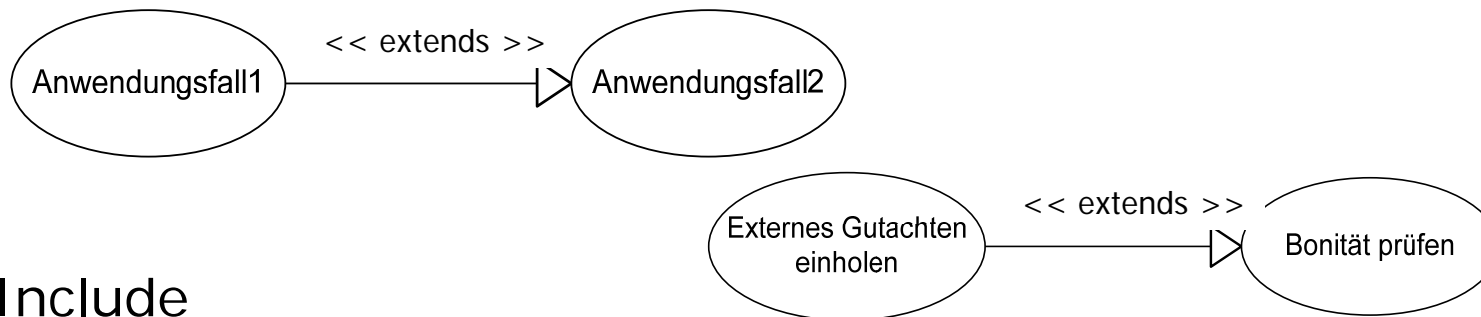


- Generalisierung
 - Generalisierung von Anwendungsfällen
 - Anwendungsfall2 generalisiert das Verhalten von Anwendungsfall1.



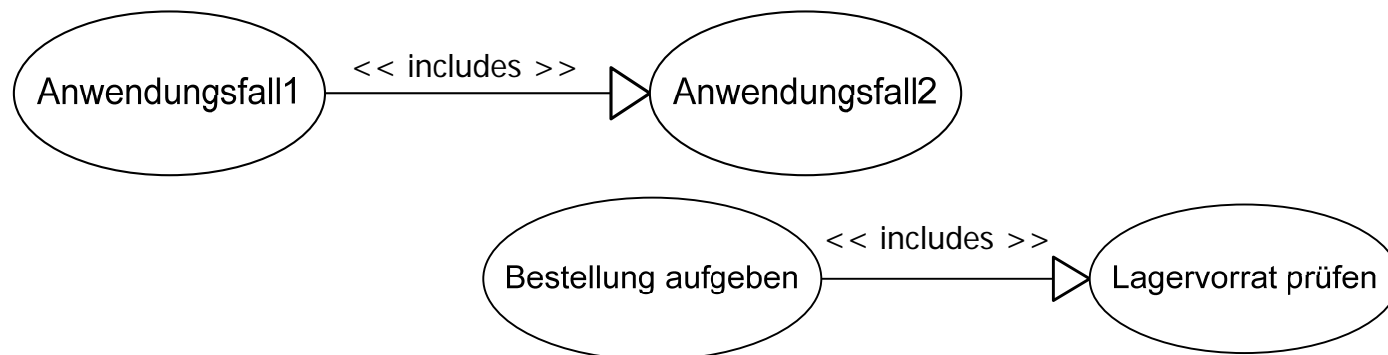
- Extend

- Erweiterung des Anwendungsfalls
- Anwendungsfall2 wird durch Anwendungsfall1 erweitert

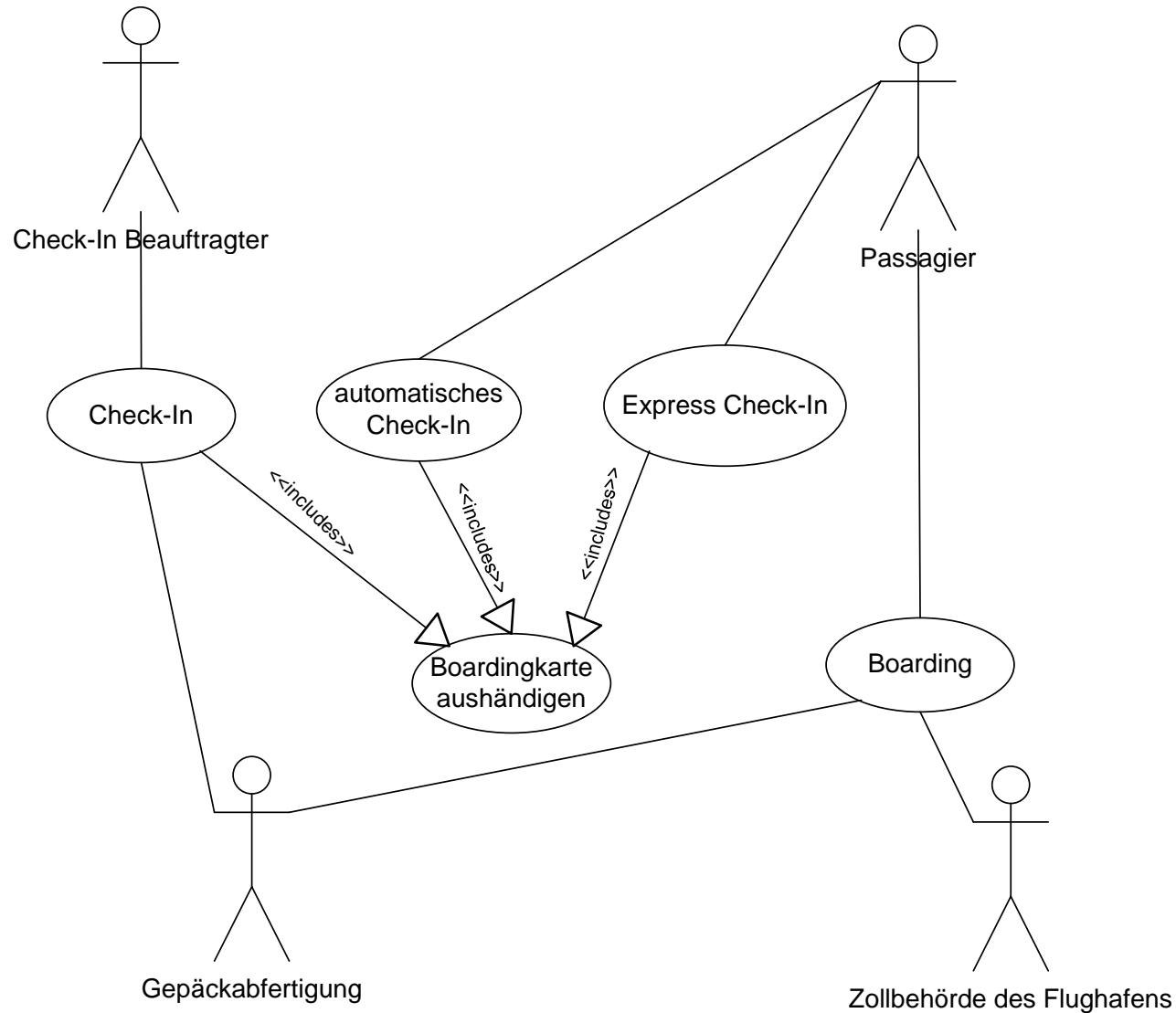


- Include

- Einbeziehung eines Anwendungsfalls
- Anwendungsfall1 erhält das Verhalten von Anwendungsfall2



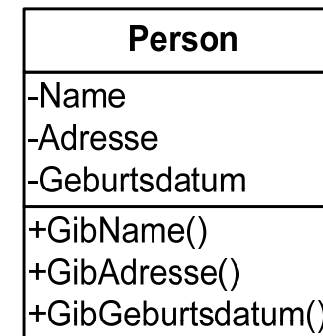
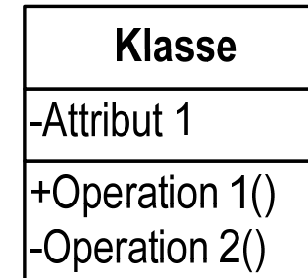
Anwendungsfalldiagramm (Beispiel)



- Klassendiagramm
 - Darstellung der statischen internen Struktur eines Systems
 - Beschreibung der logischen Beziehungen von Strukturelementen
 - Keine Ablauf- bzw. Steuerungslogik

- Objektdiagramm
 - Instanzen eines Klassendiagramms
 - „Snapshot“ eines Systemzustands

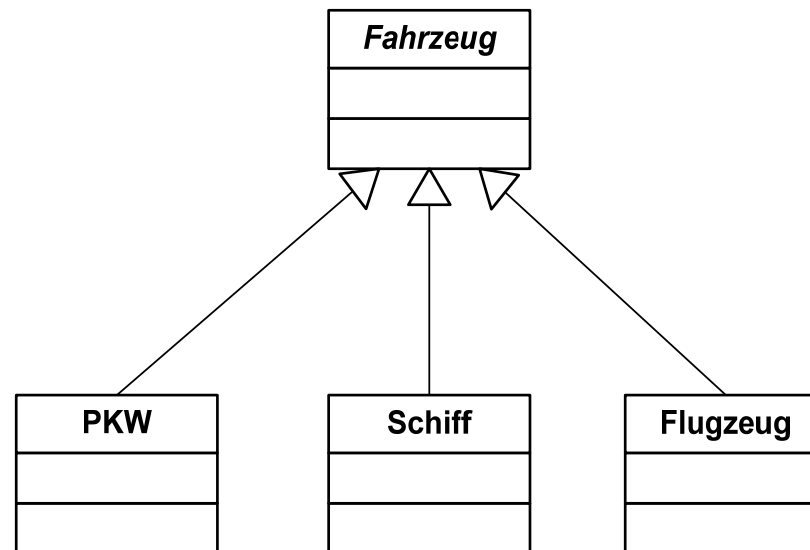
- Klassen werden durch Rechtecke dargestellt, die den Namen der Klasse, die Attribute und die Operationen beinhalten.
- Klassenname ist Singular und beginnt mit Großbuchstaben.
- Attribute und Operationen werden durch horizontale Linie getrennt.
- „+/-“: Attribut/Operation ist public/private

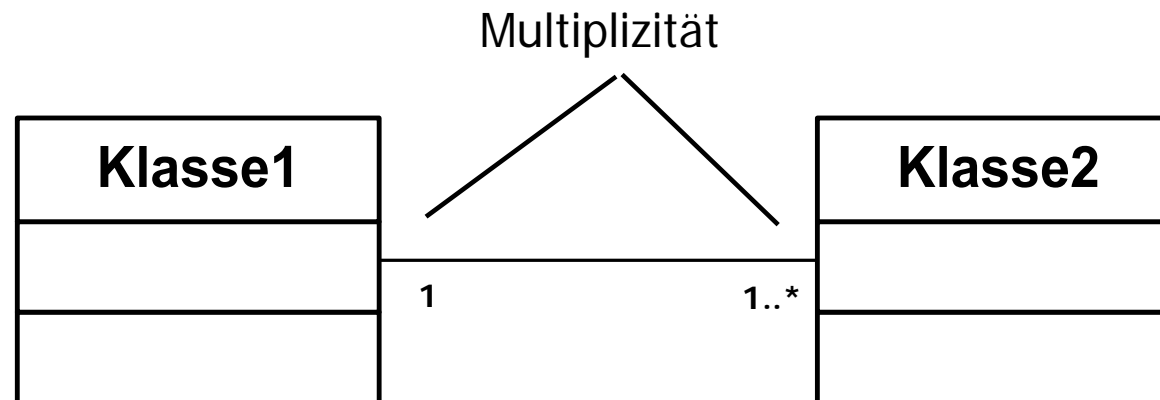


- **Klassenattribut**
 - Attribut gehört zur Klasse, nicht zum Objekt.
 - Gleicher Wert für alle Instanzen.
 - Z.B. Attribut „Anzahl“ für die Anzahl der erzeugten Objekte einer Klasse.
 - Klassenattribut wird im Klassendiagramm unterstrichen.

- **Klassenoperation**
 - Operation wird auf der Klasse ausgeführt, nicht auf dem Objekt.
 - Z.B. „Zähle erzeugte Objekte der Klasse“
 - Klassenoperation wird im Klassendiagramm unterstrichen.

- Definition/Zusammenfassung gemeinsamer Eigenschaften
- Von abstrakten Klassen sind keine Objekte instanzierbar.
- Basis für die Ableitung von Unterklassen.
- Abstrakte Methoden werden überschrieben.
- Der Name der abstrakten Klassen wird *kursiv* dargestellt.

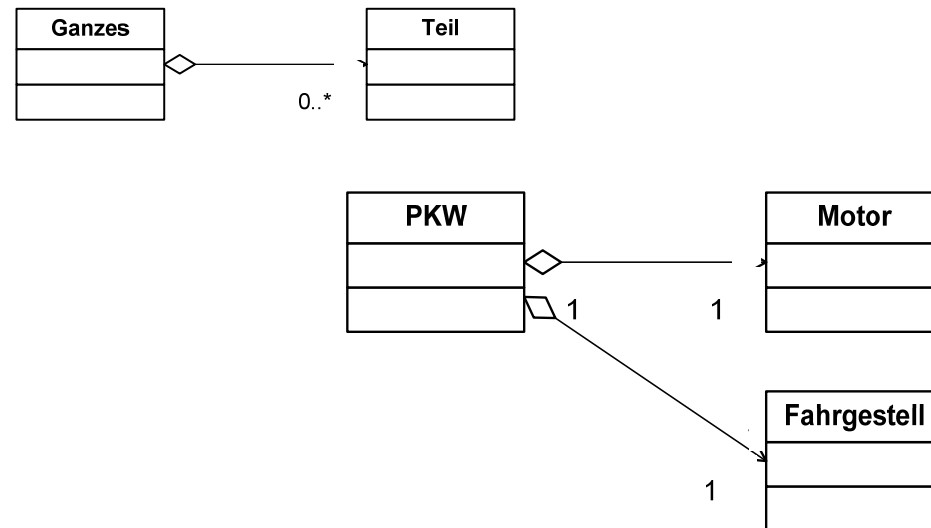




- Beschreibt als Relation zwischen Klassen die gemeinsame Semantik und Struktur einer Menge von Objektverbindungen.
- Wird durch eine Linie zwischen den Klassen dargestellt.
- Die Multiplizität min..max an der Assoziation beschreibt die minimale bzw. maximale Anzahl von Objekten, mit der ein Objekt der gegenüberliegenden Klasse verbunden ist.
Der Stern (*) steht dabei für eine beliebige Anzahl von Objekten.

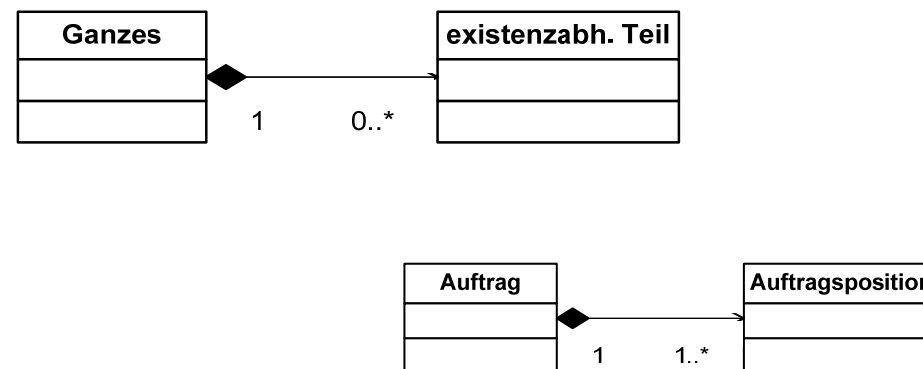
- Aggregation

- Beschreibung der Zusammengehörigkeit

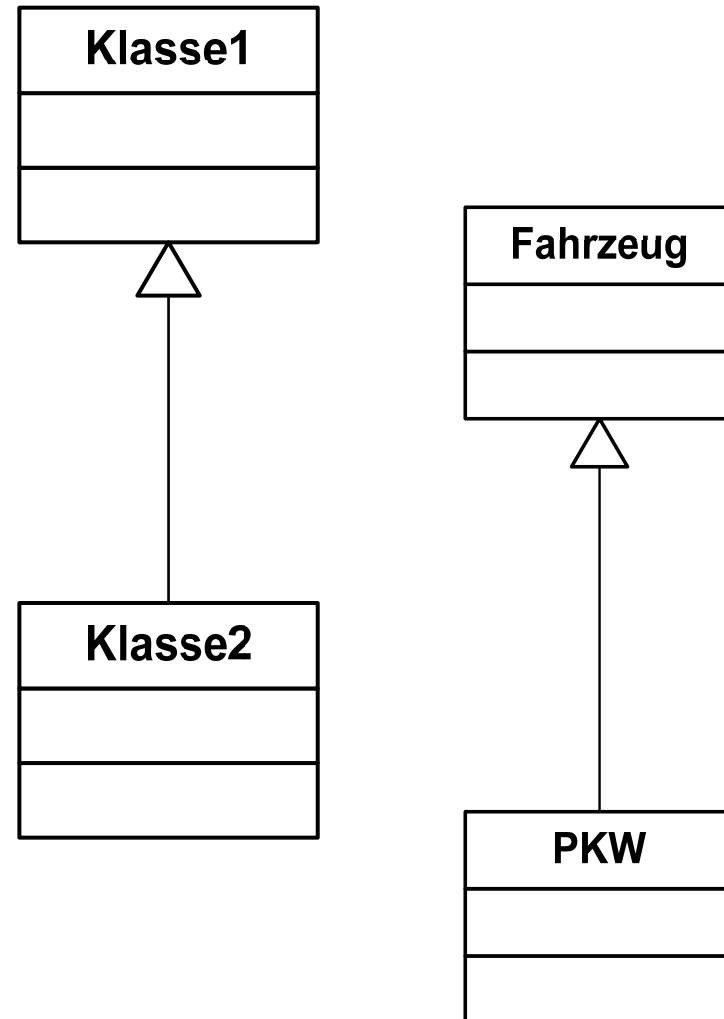


- Komposition

- Strenge Form der Aggregation
- Existenzabhängigkeit

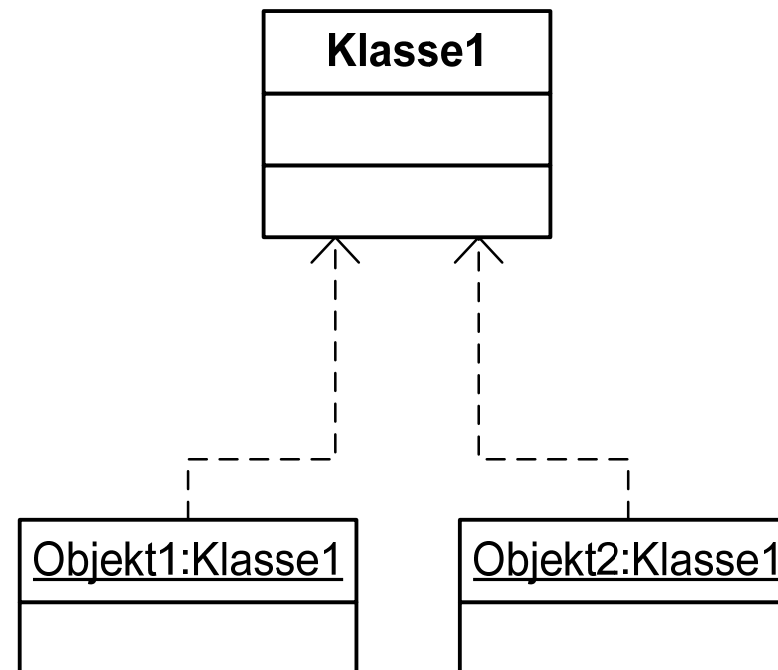


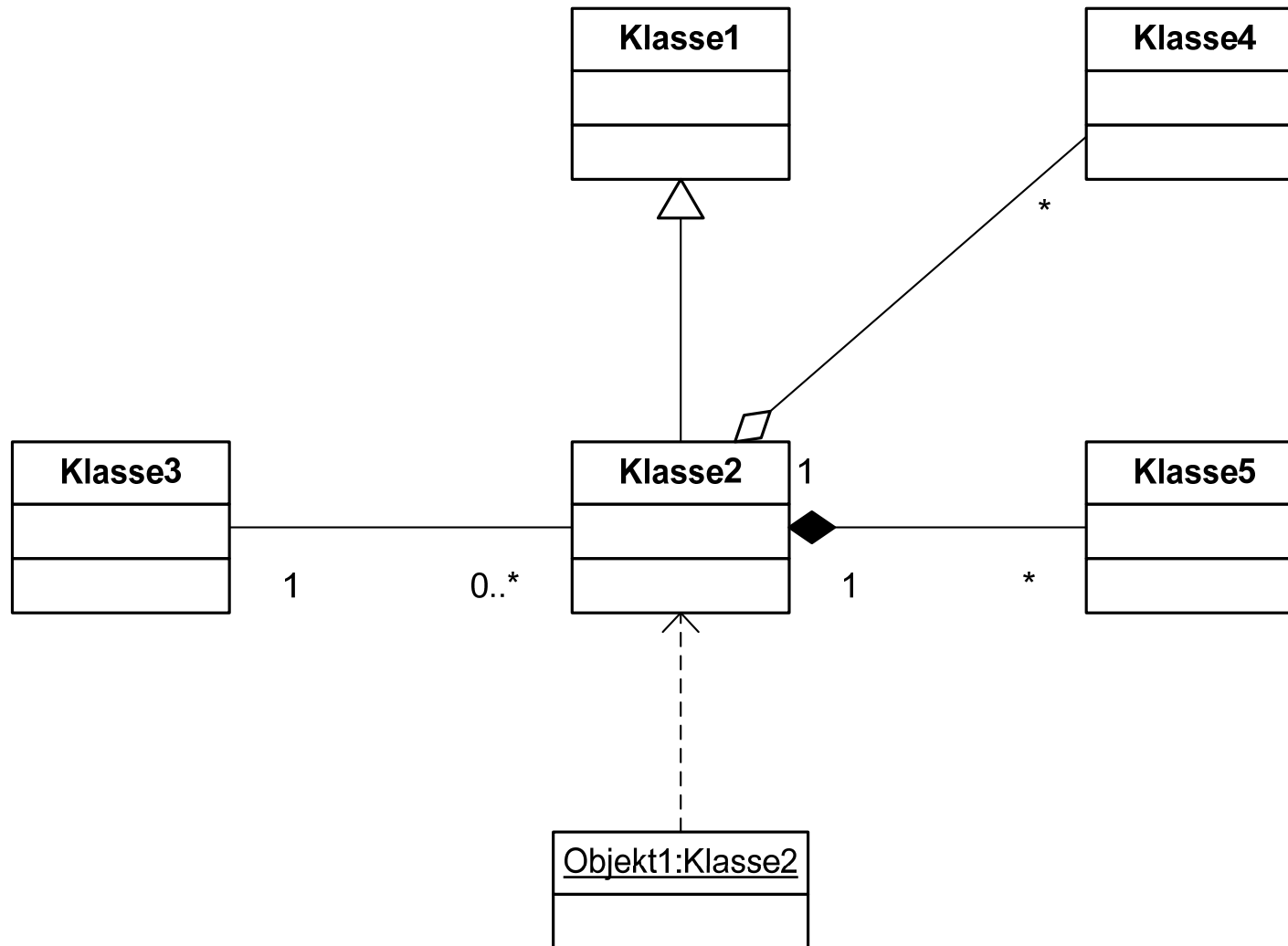
- Gibt die Relation zwischen Ober- und Unterklasse an.
- Wird als Pfeil von der Unterklasse auf die Oberklasse dargestellt.
- Klasse2 erbt die Eigenschaften von Klasse1.



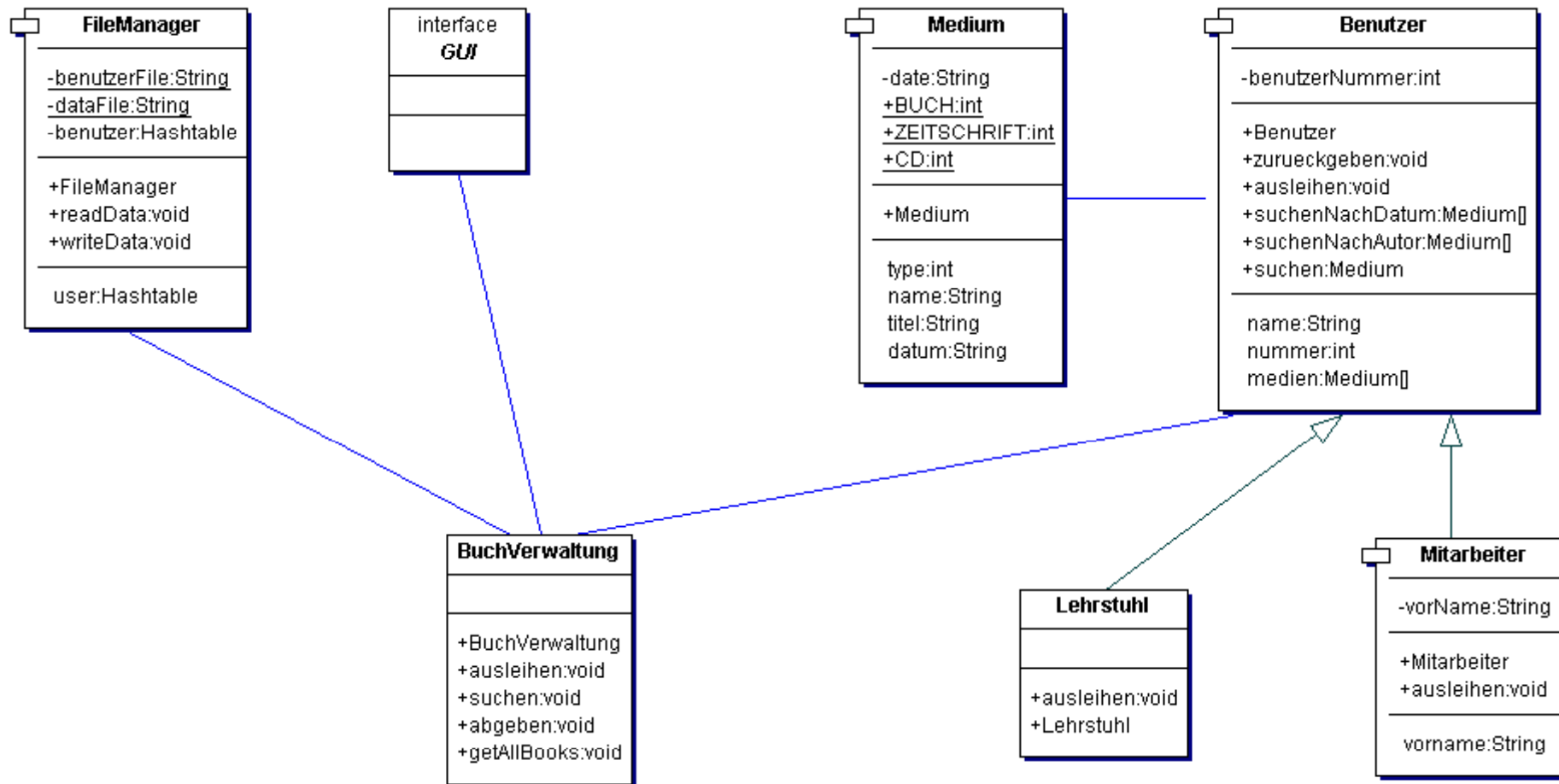
- Darstellung der Beziehung „Klasse - Objekt“
- Ein Objekt ist eine Instanz einer Klasse.

- Klasse
 - Attribute
 - Operationen
- Objekt
 - Attributwerte
 - Nachrichten





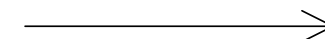
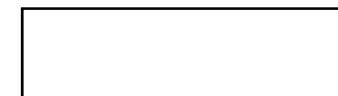
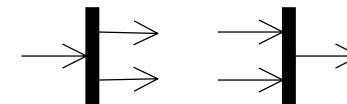
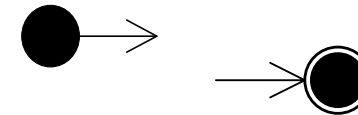
Klassendiagramm (Beispiel)

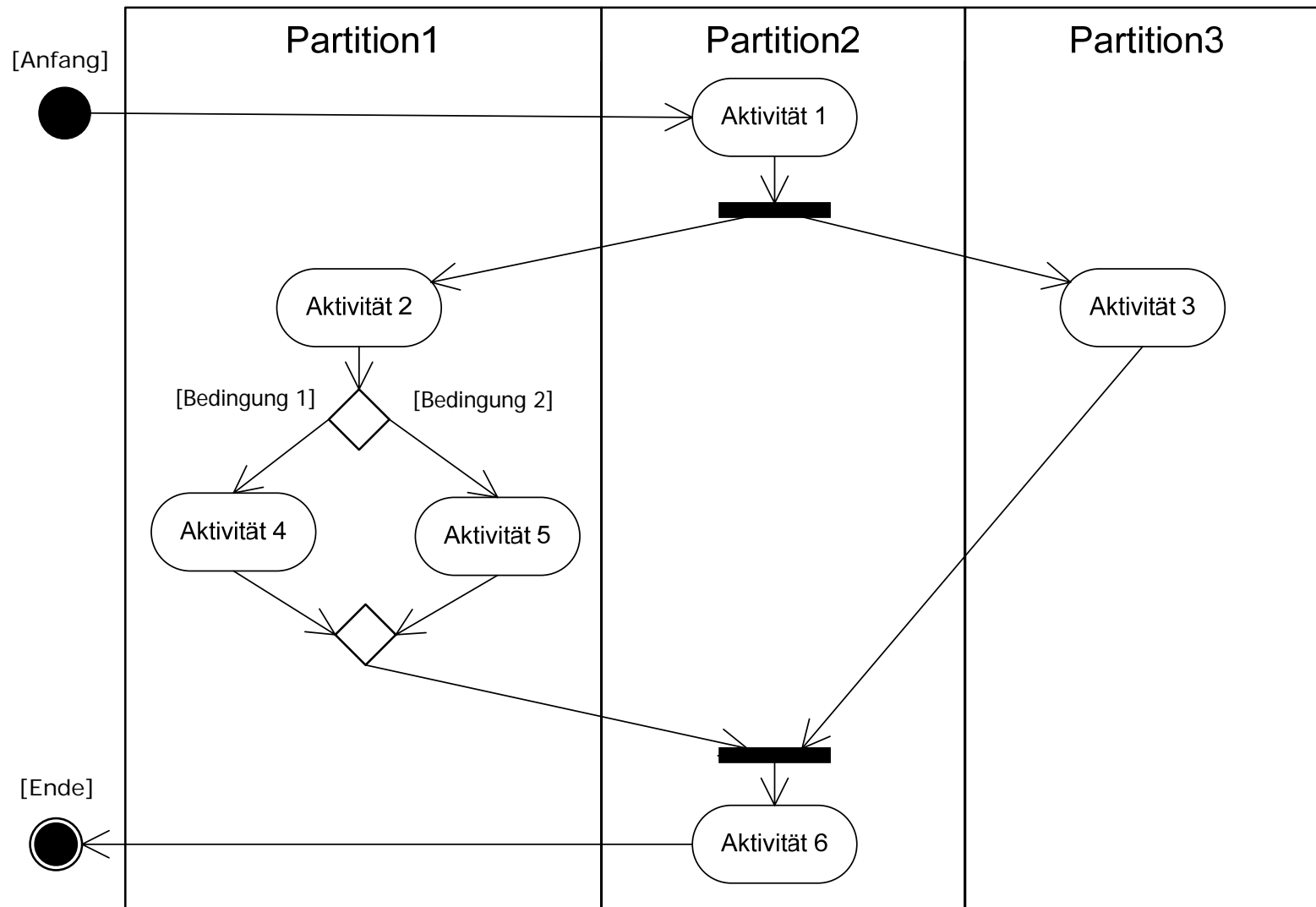


- Aktivitätsdiagramme beschreiben Verarbeitungsabläufe im System.
- Zentrales Element: Aktivität
Eine Aktivität beschreibt einen Arbeitsschritt innerhalb eines Verarbeitungsablaufes.
- Aktivitäten werden nach Verantwortlichkeiten strukturiert.
- Verschiedene Sichten
 - Aus konzeptioneller Sicht
 - Z.B. Geschäftsprozesse
 - Aus Implementierungssicht
 - Z.B. Methoden von Objekten

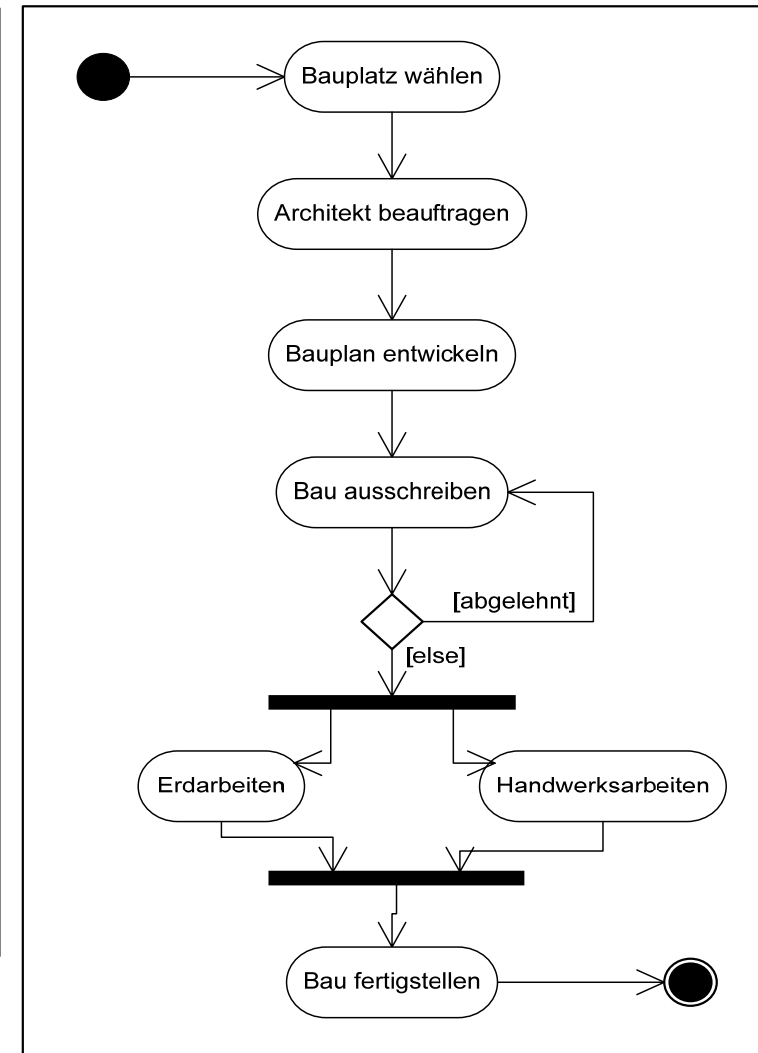
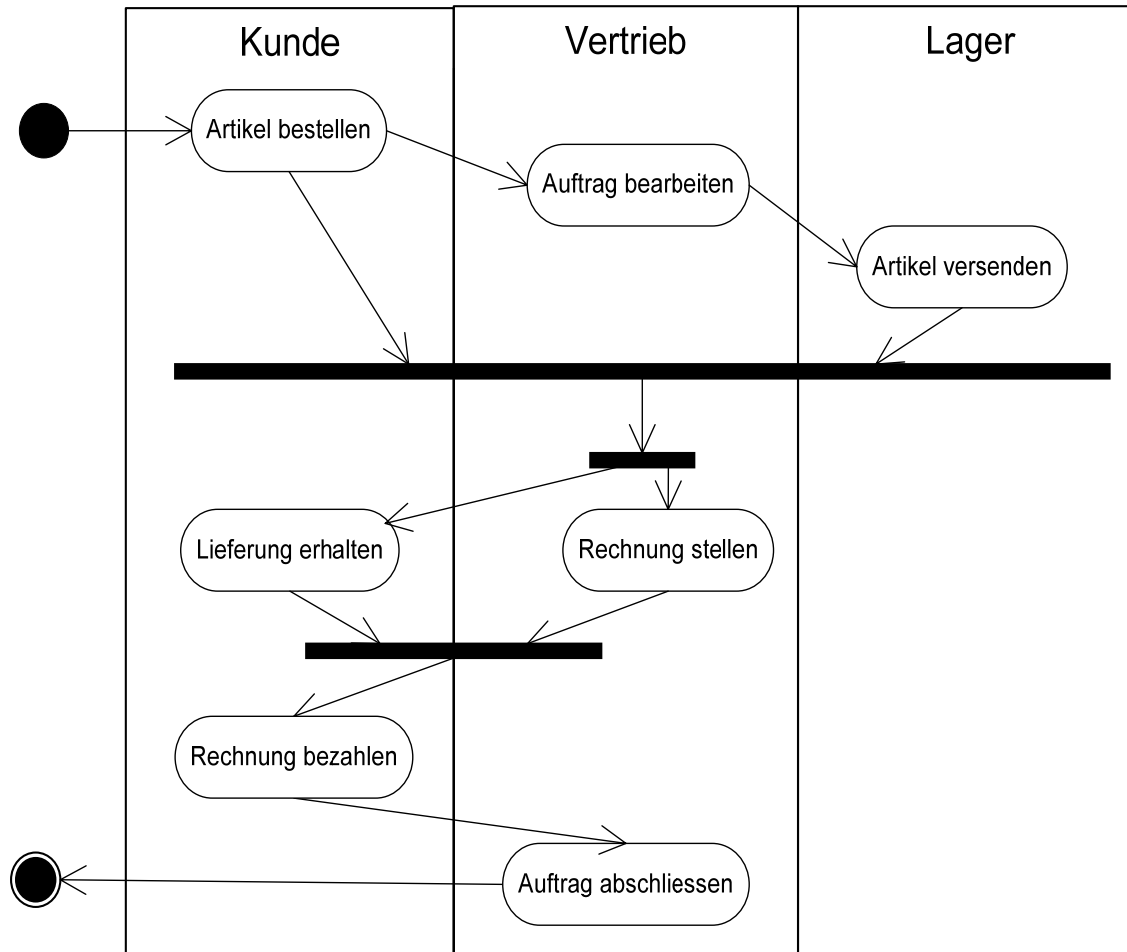
Notationselemente

- Anfangs-/Endzustand
- Aktivität
- Entscheidung
- Aufspaltung/Zusammenführung
- Verantwortlichkeit
- Kontrollfluss





Aktivitätsdiagramm (Beispiele)



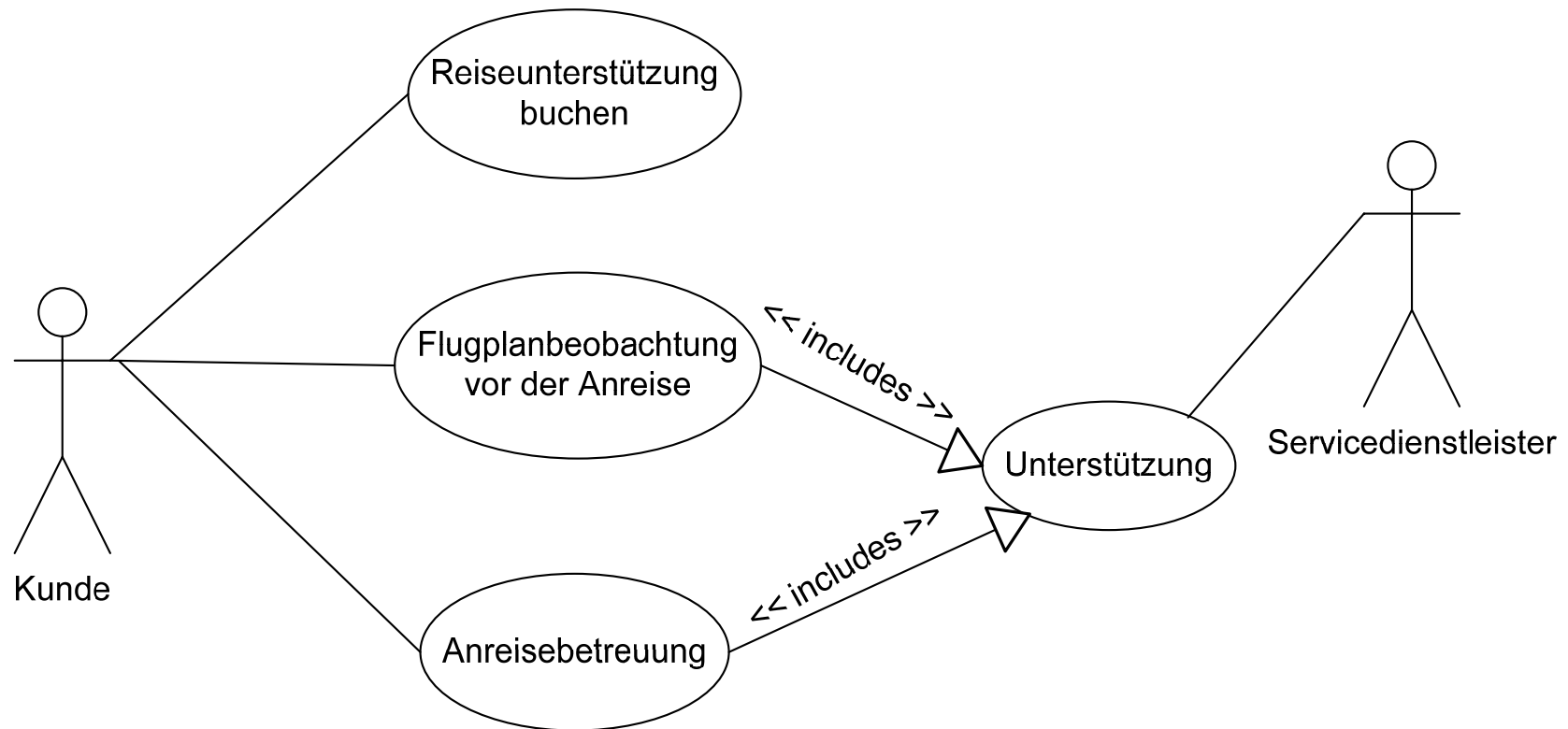
1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

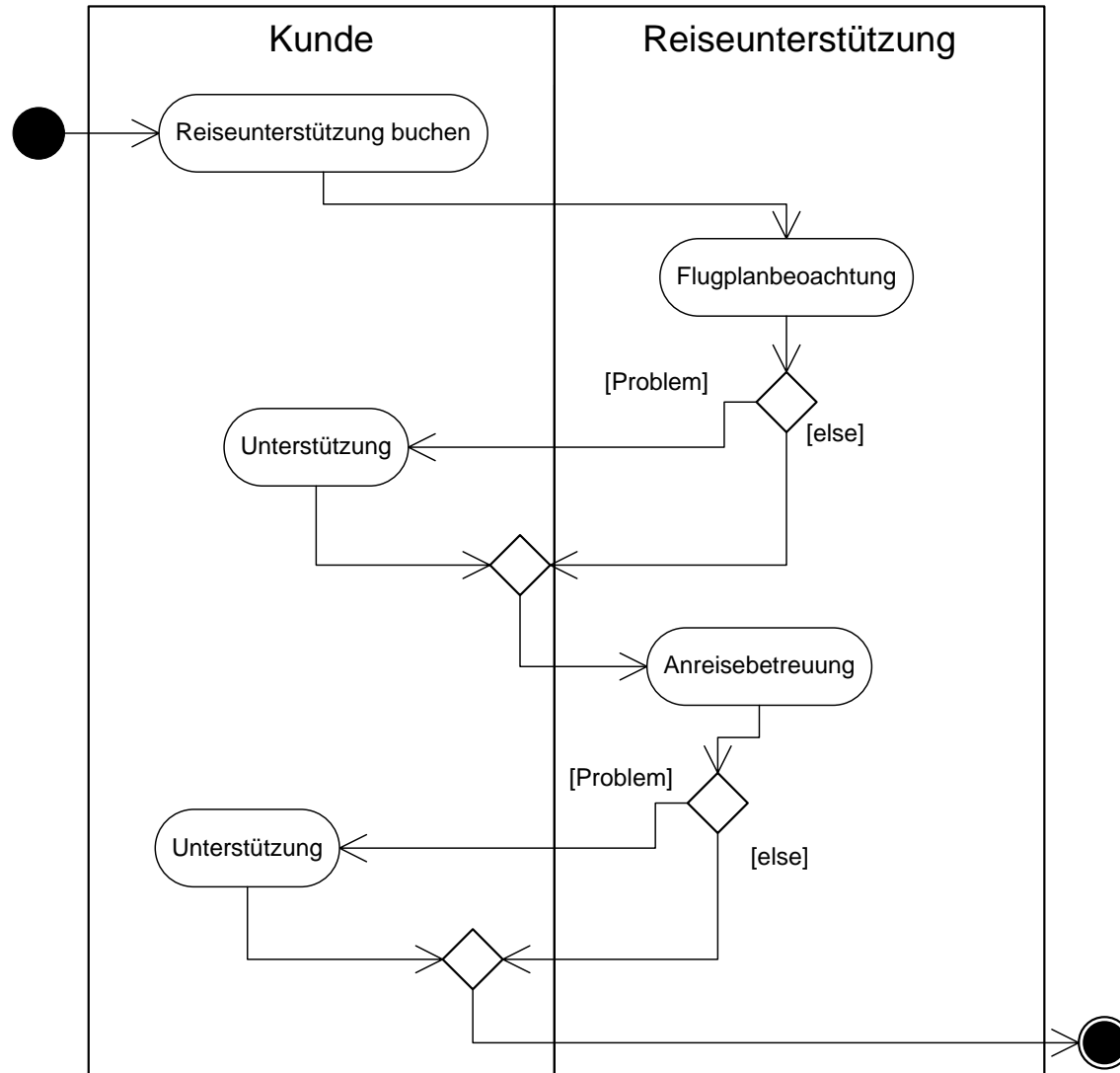
3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- Ziel
 - Modellierung eines Prototypen zur Reiseunterstützung von Frequent Flyers mittels UML
- Modellierung
 - Modellierung aus fachlicher Sicht
 - Anwendungsfalldiagramm beschreibt die Leistungen und Schnittstellen des Systems.
 - Aktivitätsdiagramm beschreibt den Ablauf der Reiseunterstützung.

Reiseunterstützung (Anwendungsfalldiagramm)



Reiseunterstützung (Aktivitätsdiagramm)



1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz
2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze
3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- Geschäftsprozess, Geschäftsvorgang, Geschäftsvorfall, Workflow, Prozesskette, betrieblicher Ablauf, ...
- Business Process Reengineering, Business Process Improvement, Business Process Innovation, Business Transformation, Business Engineering, Geschäftsprozessoptimierung, ...

- **Arbeitsdefinition**

Menge von manuellen, teil-automatisierten oder automatisierten betrieblichen Aktivitäten, die nach bestimmten Regeln auf ein bestimmtes Ziel hin ausgeführt werden.

- Aktivitäten hängen bzgl. betroffener Personen, Maschinen, Dokumenten, Ressourcen u.ä. miteinander zusammen.
- Aktivitäten werden von *personellen* und *nicht-personellen (maschinellen) Aufgabenträgern* ausgeführt.
- Aufgaben sind als zu erbringende Leistungen zu verstehen, wobei die Erfüllung einer Aufgabe durch Ausführung einer oder mehrerer Aktivitäten erfolgt.
- Ein Geschäftsprozess erzeugt für Kunden ein Ergebnis von Wert.
- Ein *kooperativer* oder *arbeitsteiliger* Geschäftsprozess ist dadurch gekennzeichnet, dass mindestens zwei Aufgabenträger seine Aktivitäten ausführen.

- Abwicklung eines Schadenfalles bei einer Versicherung
- Bearbeitung eines Kreditantrags in einer Bank
- Planung, Buchung, Abrechnung einer Reise in einem Reisebüro
- Bearbeitung einer Einkommenssteuererklärung in einem Finanzamt
- Bearbeitung eines Bauantrags in der öffentlichen Verwaltung
- Beantragung und Abrechnung einer Dienstreise
- ...

- Geschäftsprozessmodellierung
 - Beschreibung aller relevanten Aspekte eines Geschäftsprozesses in einer Beschreibungssprache
 - Beschreibungssprachen
 - z.B. EPK oder Petri-Netze

- Geschäftsprozess-Modell/Schema
 - Ergebnis der Geschäftsprozessmodellierung

1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

■ Historie

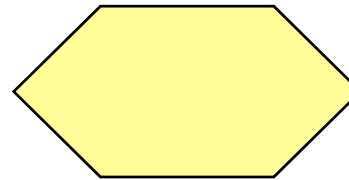
- Semiformale, graphische Beschreibungssprache
- 1992 entwickelt von Prof. Scheer (Uni Saarbrücken) und Mitarbeitern.
- Hoher Verbreitungsgrad in Deutschland in Verbindung mit ARIS-Toolset

■ Einsatz

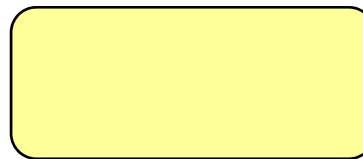
- EPKs beschreiben Prozesse, d.h. zusammenhängende Aktivitäten und Ablaufreihenfolgen
- *Ereignisse* lösen Aktivitäten aus und sind das Ergebnis von Aktivitäten.
- Ein Ereignis ist definiert als das Auftreten eines Objektes oder die Änderung einer bestimmten Objekteigenschaft.
- Ereignisse und Aktivitäten können mit verschiedenen Verknüpfungsoperatoren miteinander verbunden werden:

o *und, oder, exklusives oder*

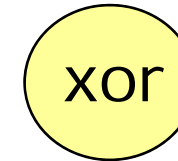
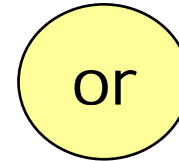
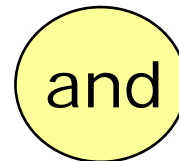
Ereignis



Aktivität

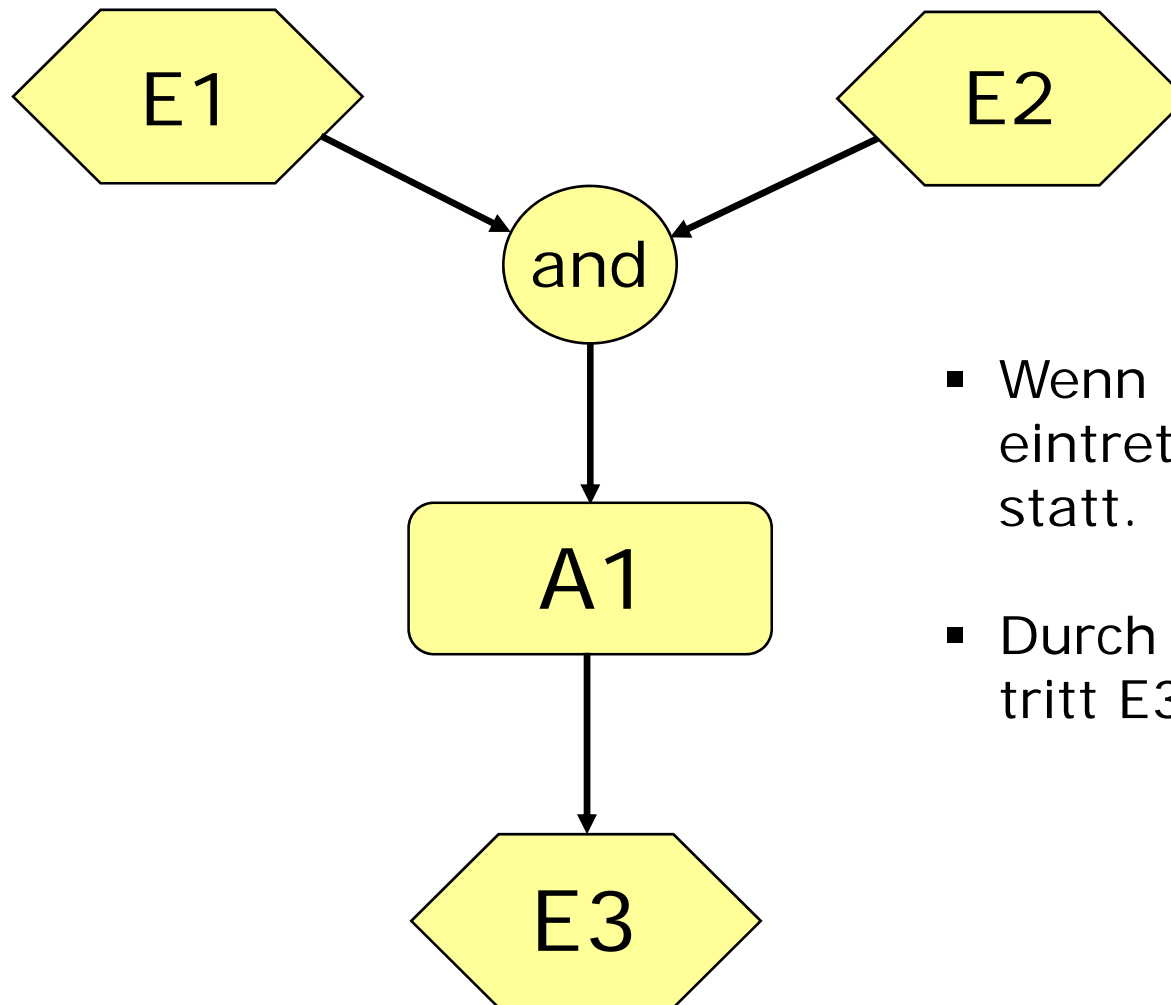


Verknüpfungs-
operatoren

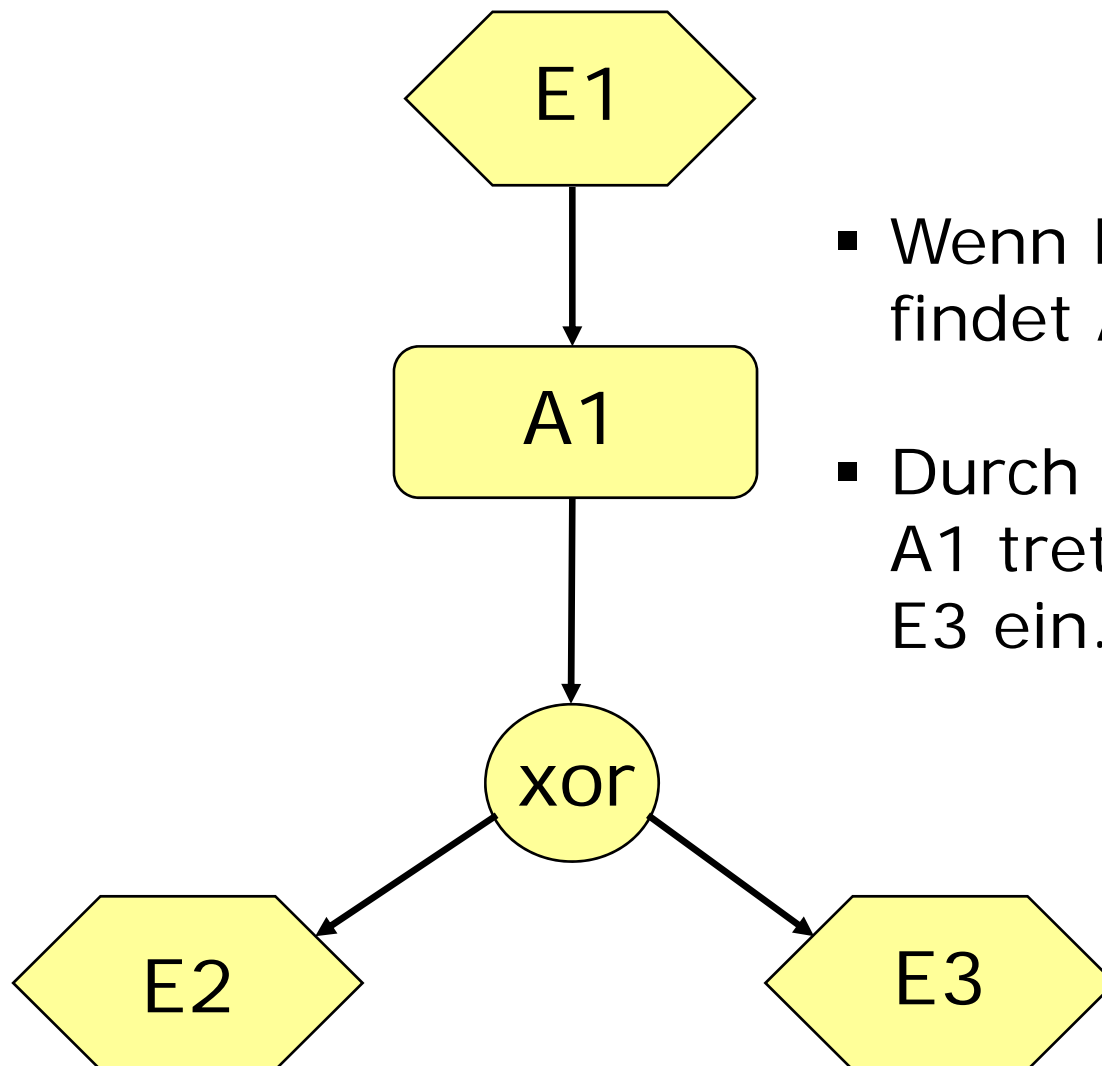


Abhängigkeit zwischen
Ereignis und Funktion



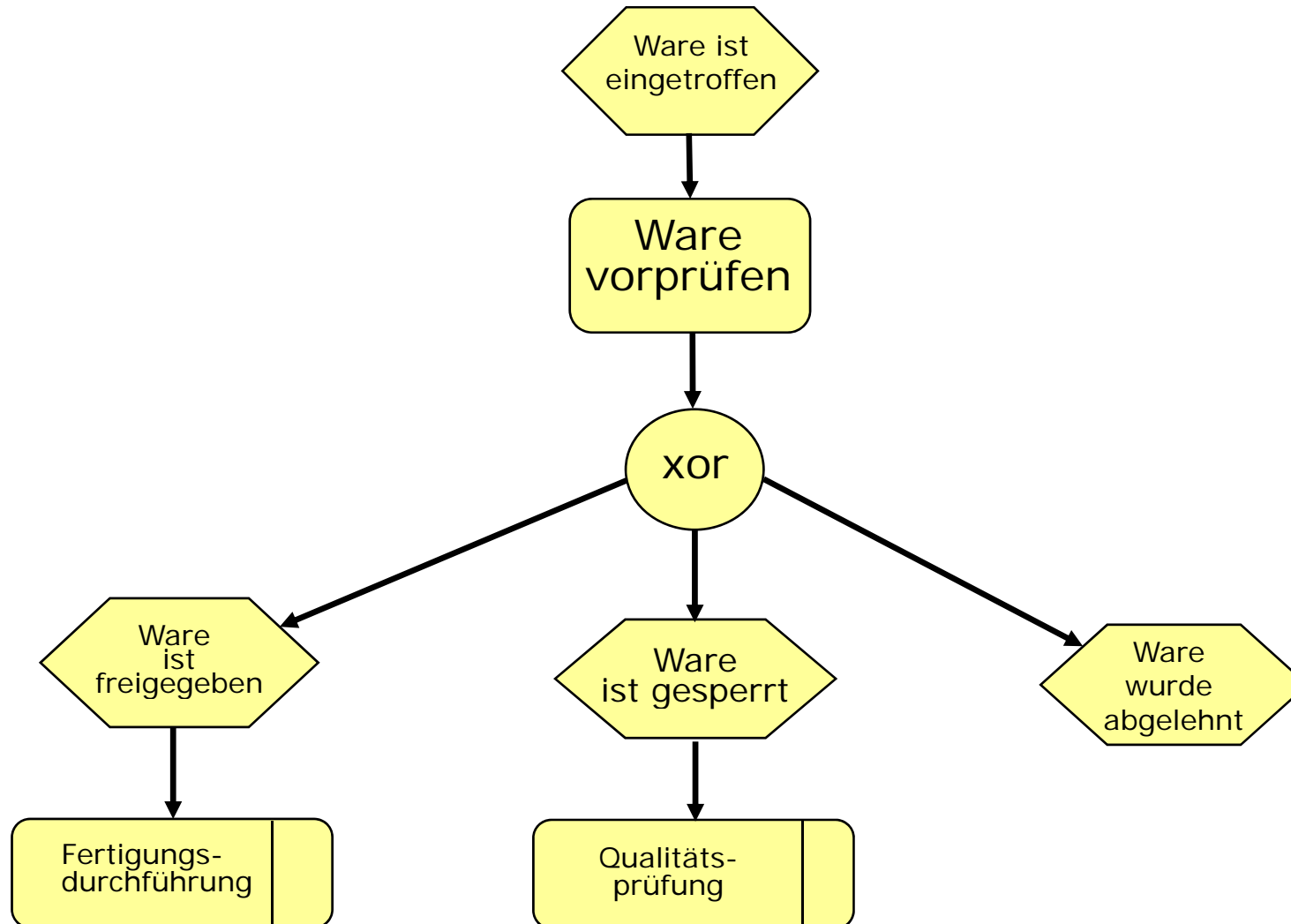


- Wenn Ereignisse E1 und E2 eintreten, findet Aktivität A1 statt.
- Durch das Stattfinden von A1 tritt E3 ein.



- Wenn Ereignis E1 eintritt, findet Aktivität A1 statt.
- Durch das Stattfinden von A1 treten entweder E2 oder E3 ein.

- Jede EPK beginnt mit mindestens einem Ereignis (Startereignis) und wird mit mindestens einem Ereignis (Endereignis) abgeschlossen.
- Ausnahme: Verweis auf andere EPK

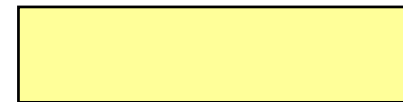


- Aktivitäten können hierarchisch verfeinert werden.

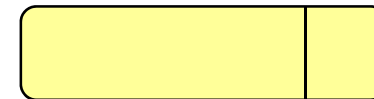
- Aktivitäten können
 - den mit der Ausführung betrauten Organisationseinheiten sowie
 - ein- und ausgehenden Datenobjekten zugeordnet werden.

Aktivitäten können den mit der Ausführung betrauten Organisationseinheiten sowie ein- und ausgehenden Datenobjekten zugeordnet werden:

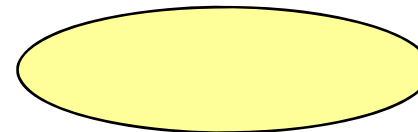
Daten- / Materialobjekt



Aktivität wird durch EPK
verfeinert



Organisationseinheit

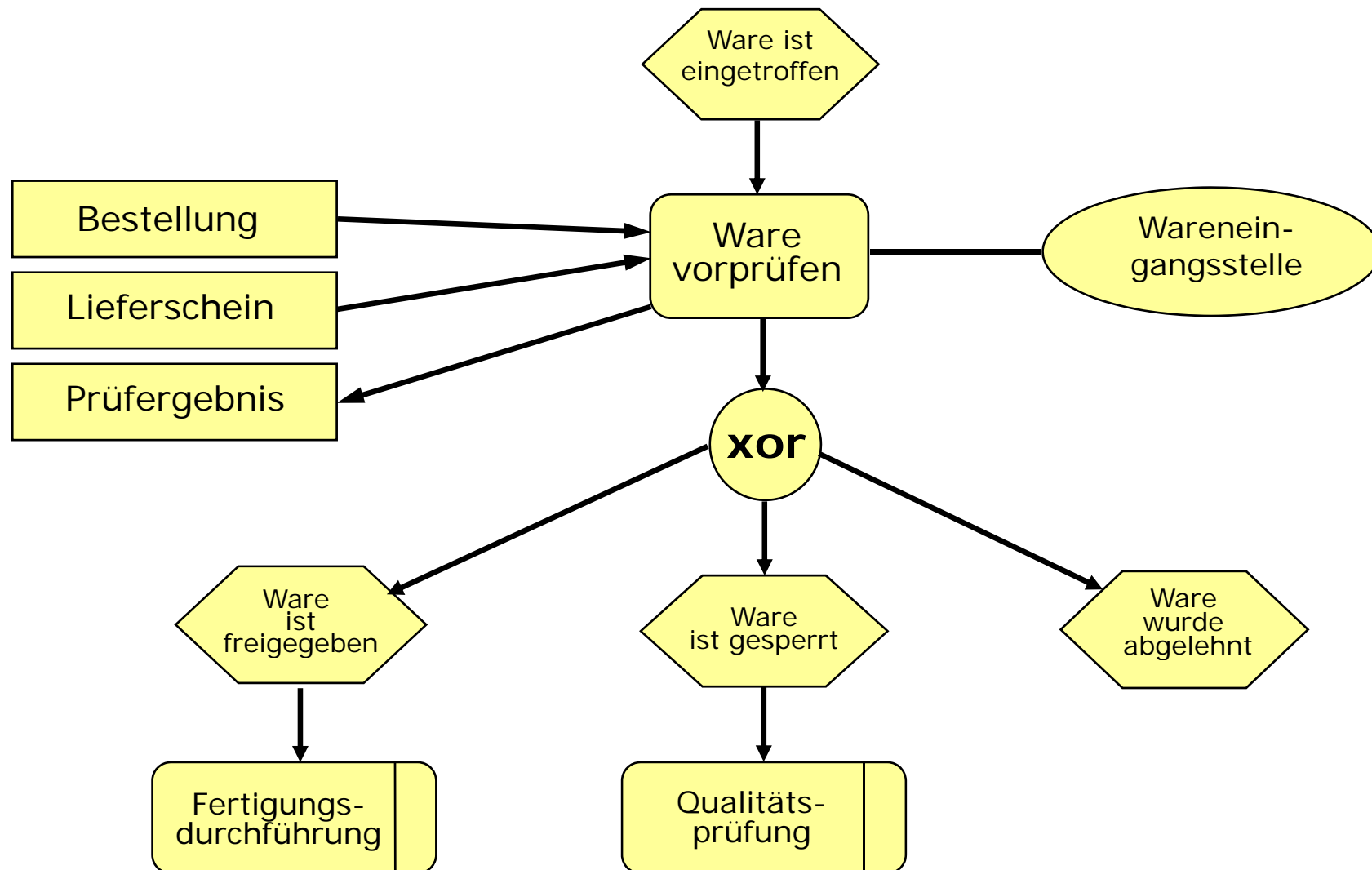


Informations-/Materialfluss

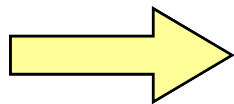


Zuordnung
Organisationseinheit





- Einfache graphische Darstellung
- Keine **präzise** Bedeutung der einzelnen Symbole, daher keine formale Analyse möglich
- Zusammenhänge zur Datenmodellierung unzureichend
- Fehlende Unterscheidung zwischen Typ und Ausprägung eines Ablaufs



keine direkte Ausführbarkeit

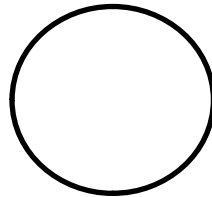
1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

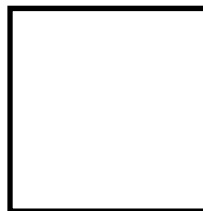
- Gehen zurück auf C.A. Petri (Dissertation 1962)
- Ermöglichen graphisch-formale Beschreibung
 - sequentieller,
 - sich gegenseitig ausschließender und
 - nebenläufiger Aktivitäten.
- Erfordern nur wenige graphische Beschreibungsstrukture
- Unterscheidung zwischen Typ und Ausprägung eines Prozesses

- Stelle:



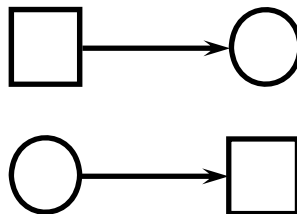
Interpretation:
Objektspeicher

- Transition:



Interpretation:
Aktivität

- Flussrelation:



Interpretation:
Input-/Output-Beziehung

Ein (*Petri-*)Netz ist ein Tupel $N = (S, T, F)$, mit

(i) S, T endliche Mengen

(ii) $S \cap T = \emptyset$

(iii) $S \cup T \neq \emptyset$

(iv) $F \subseteq (S \times T) \cup (T \times S)$

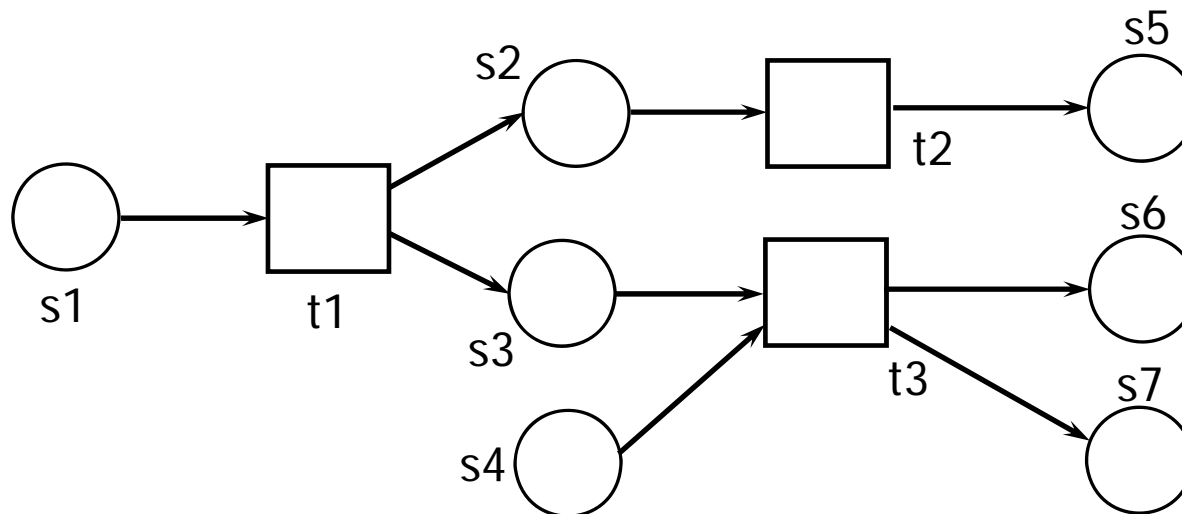
- Die Elemente aus S heißen **Stellen**, die aus T **Transitionen**.
Stellen und Transitionen werden auch als Knoten bezeichnet.
- F ist eine Menge binärer Flussrelationen (Kanten der Netze)

Die folgende Abbildung zeigt die graphische Darstellung eines Netzes $N = (S, T, F)$ mit

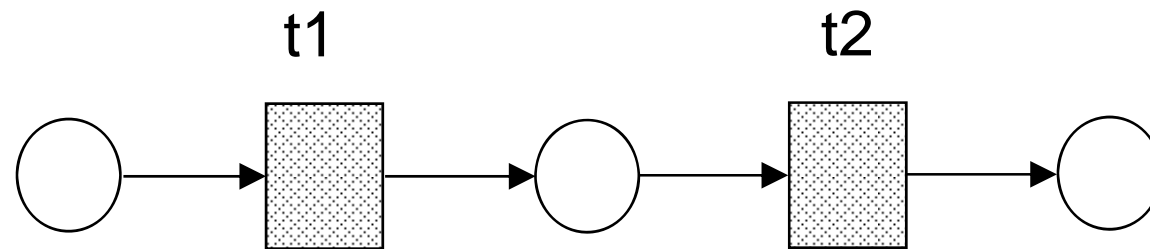
$S = \{s1, s2, s3, s4, s5, s6, s7\}$,

$T = \{t1, t2, t3\}$ und

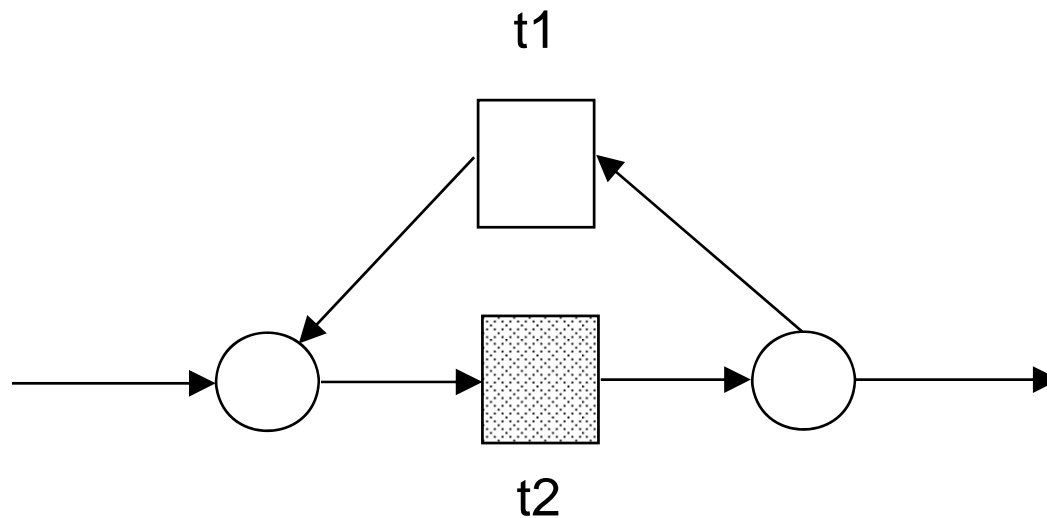
$F = \{(s1, t1), (t1, s2), (t1, s3), (s2, t2), (t2, s5), (s3, t3), (s4, t3), (t3, s6), (t3, s7)\}$



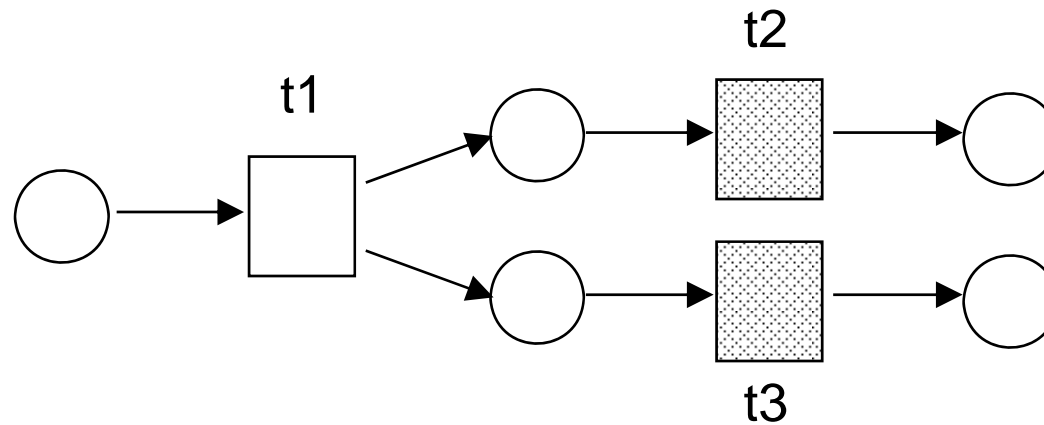
Sequenz



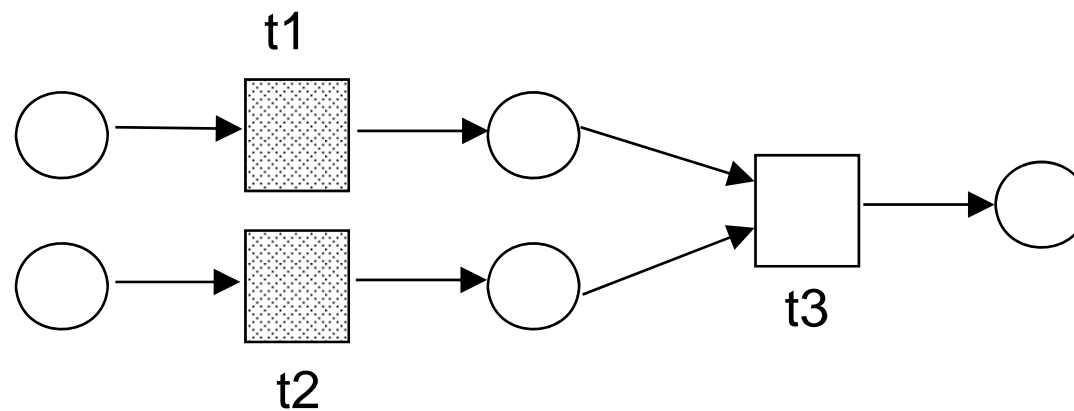
Iteration



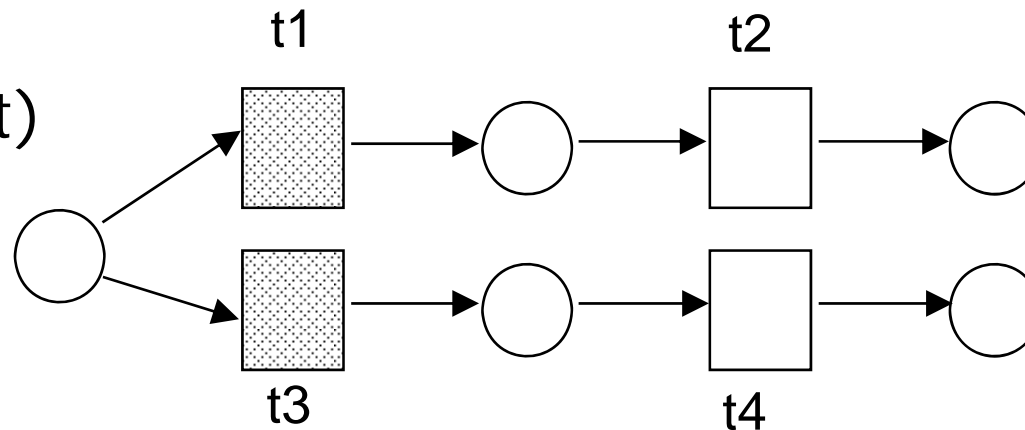
Nebenläufigkeit



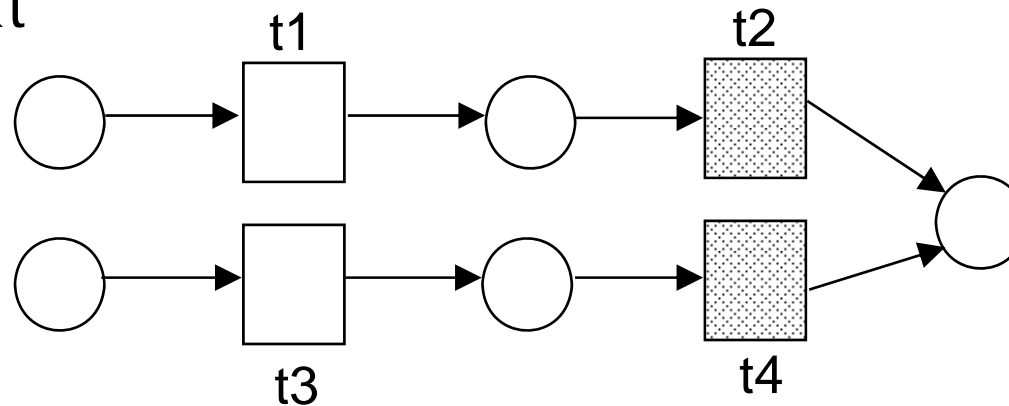
Synchronisation



Auswahl
(Vorwärtskonflikt)



Rückwärtskonflikt



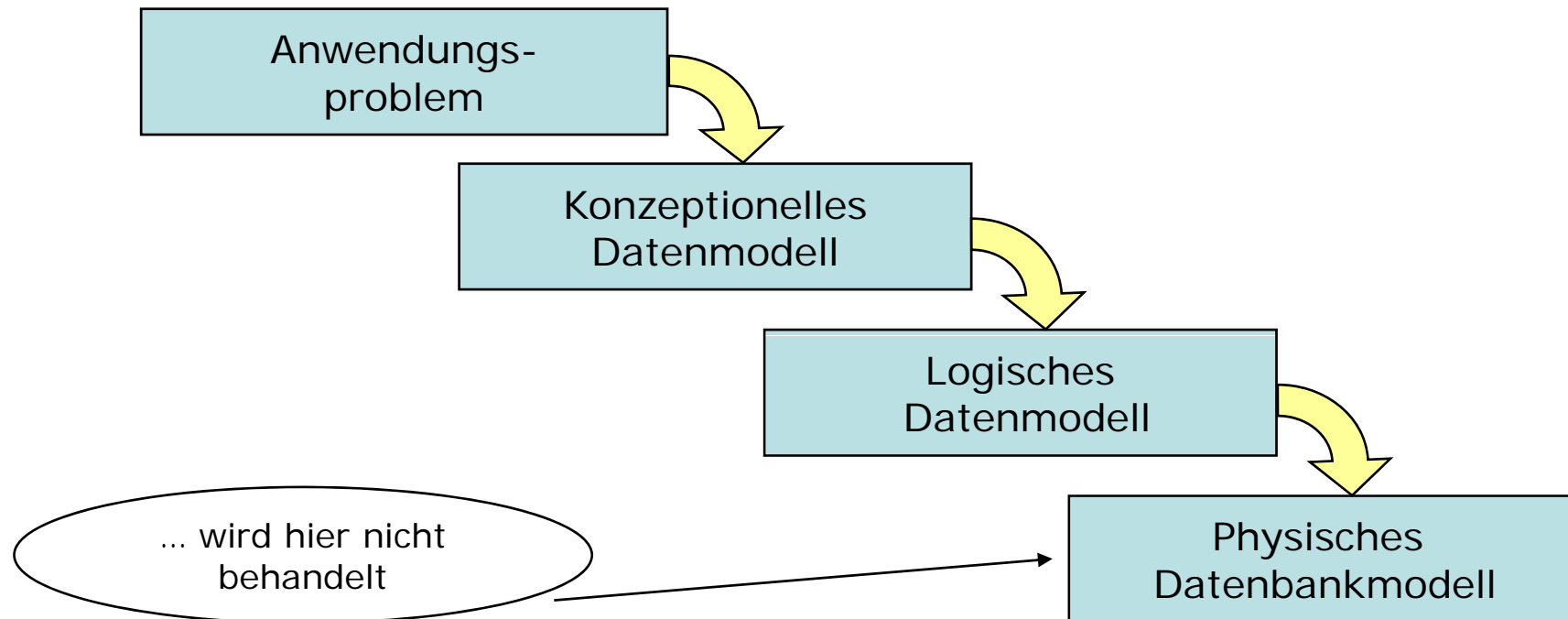
- Integrieren objektbezogene Aspekte
- Sind direkt ausführbar (Simulation)
- Erlauben schrittweise Formalisierung
- Sind mathematisch fundiert, formal analysierbar

1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- Vorgehen bei der datenorientierten Modellierung



Auszug aus der Realität, welcher
in einem Modell abzubilden ist

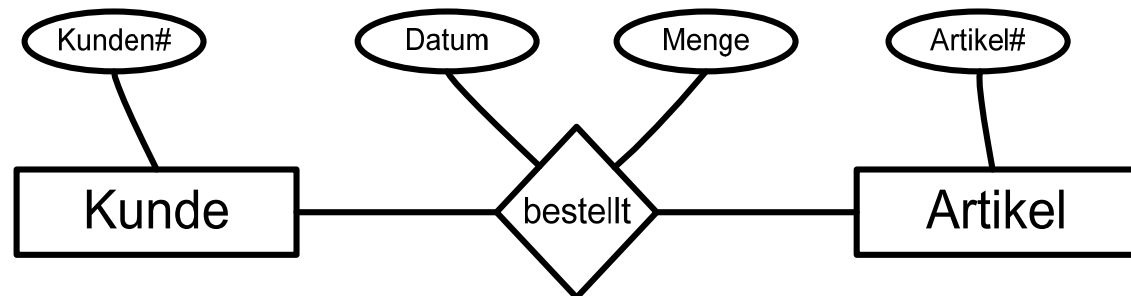
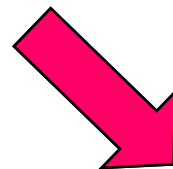


Z.B. „Kunde bestellt einen Artikel.“

- Modellierung eines Anwendungsproblems aus fachlicher Sicht
- Abstrahierung von der technischen Implementierung
- Verschiedene Modellierungsansätze (ERM, SERM, ...)



Kunde bestellt
einen Artikel

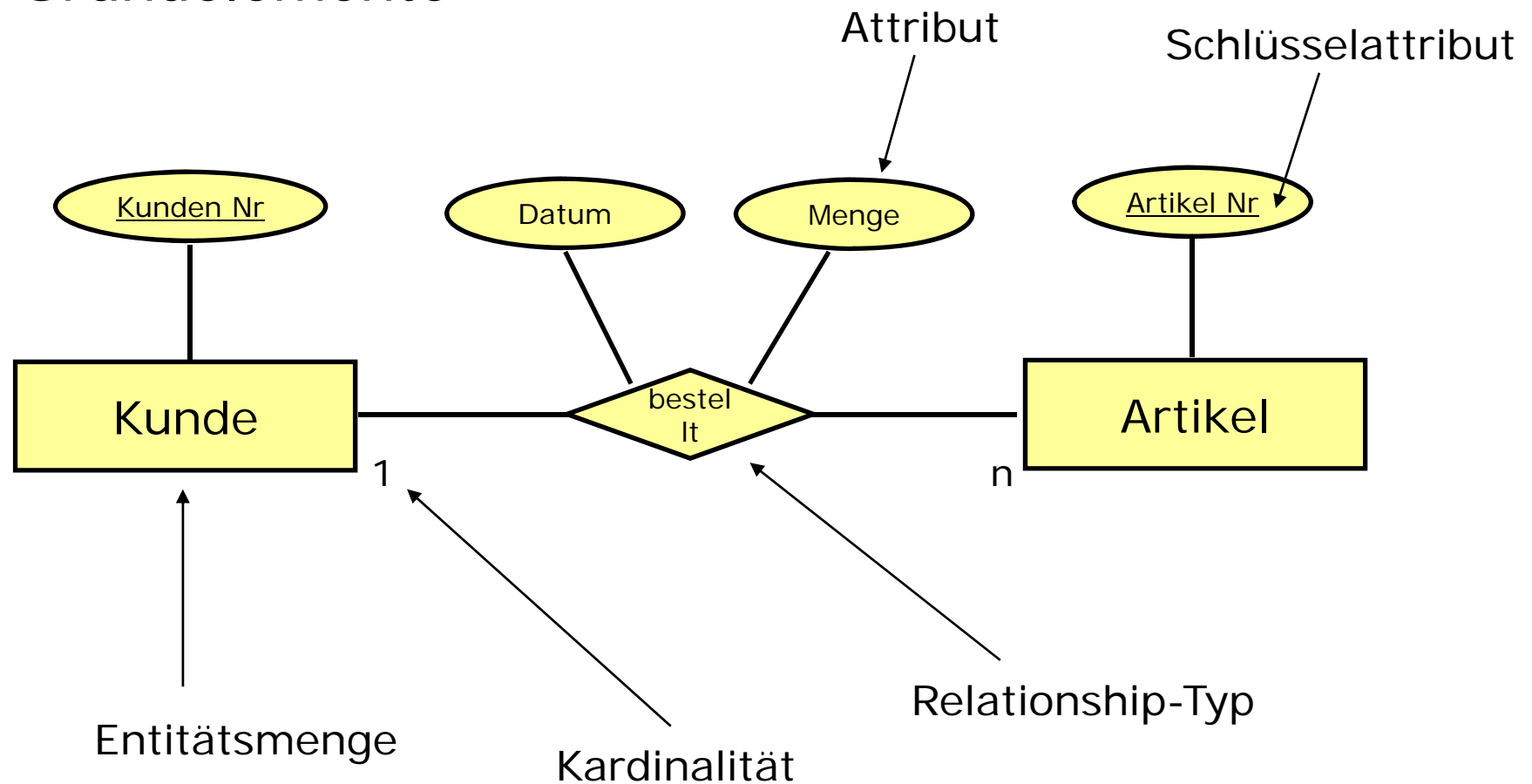


1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

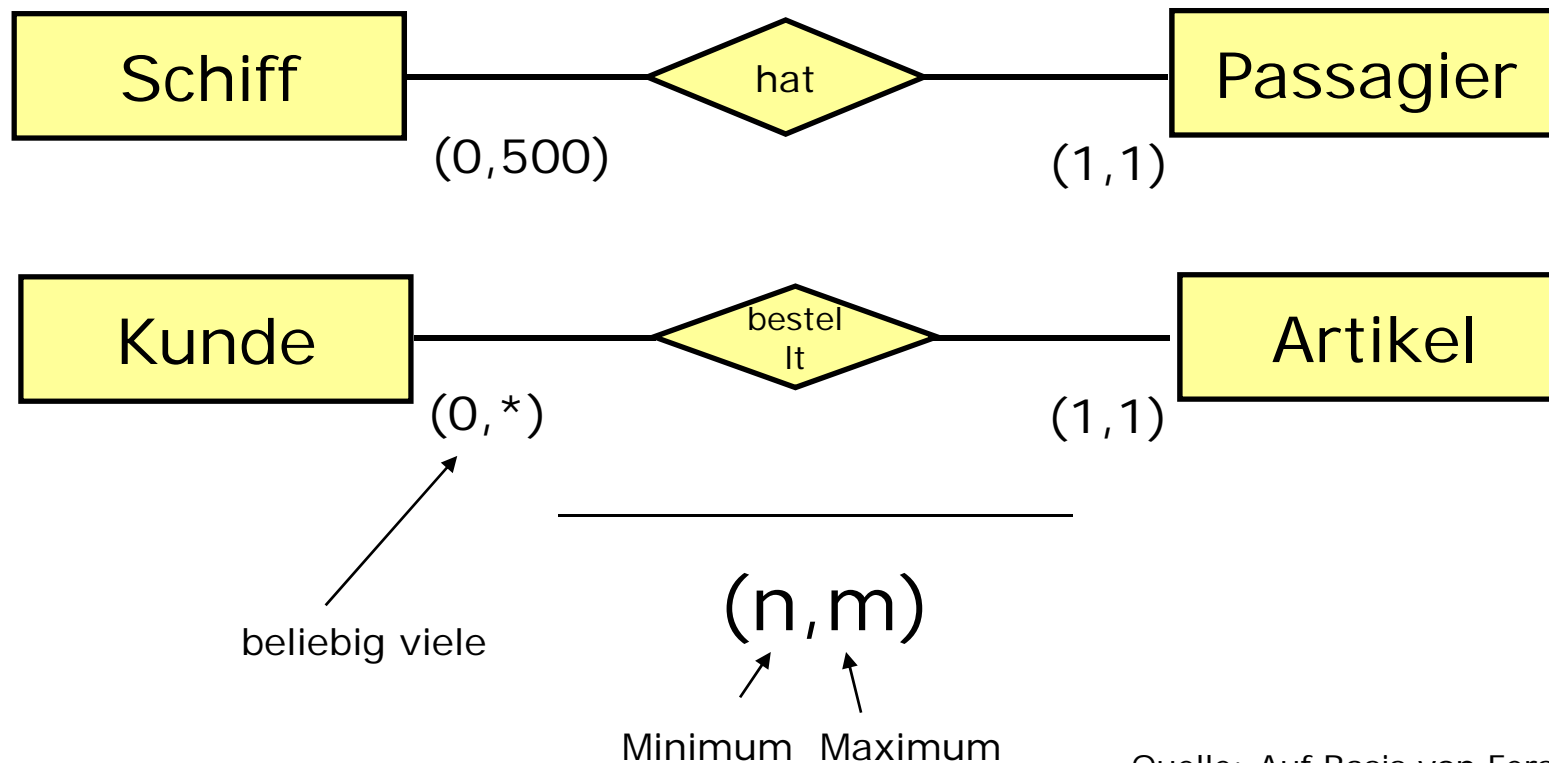
3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell

- Grundelemente



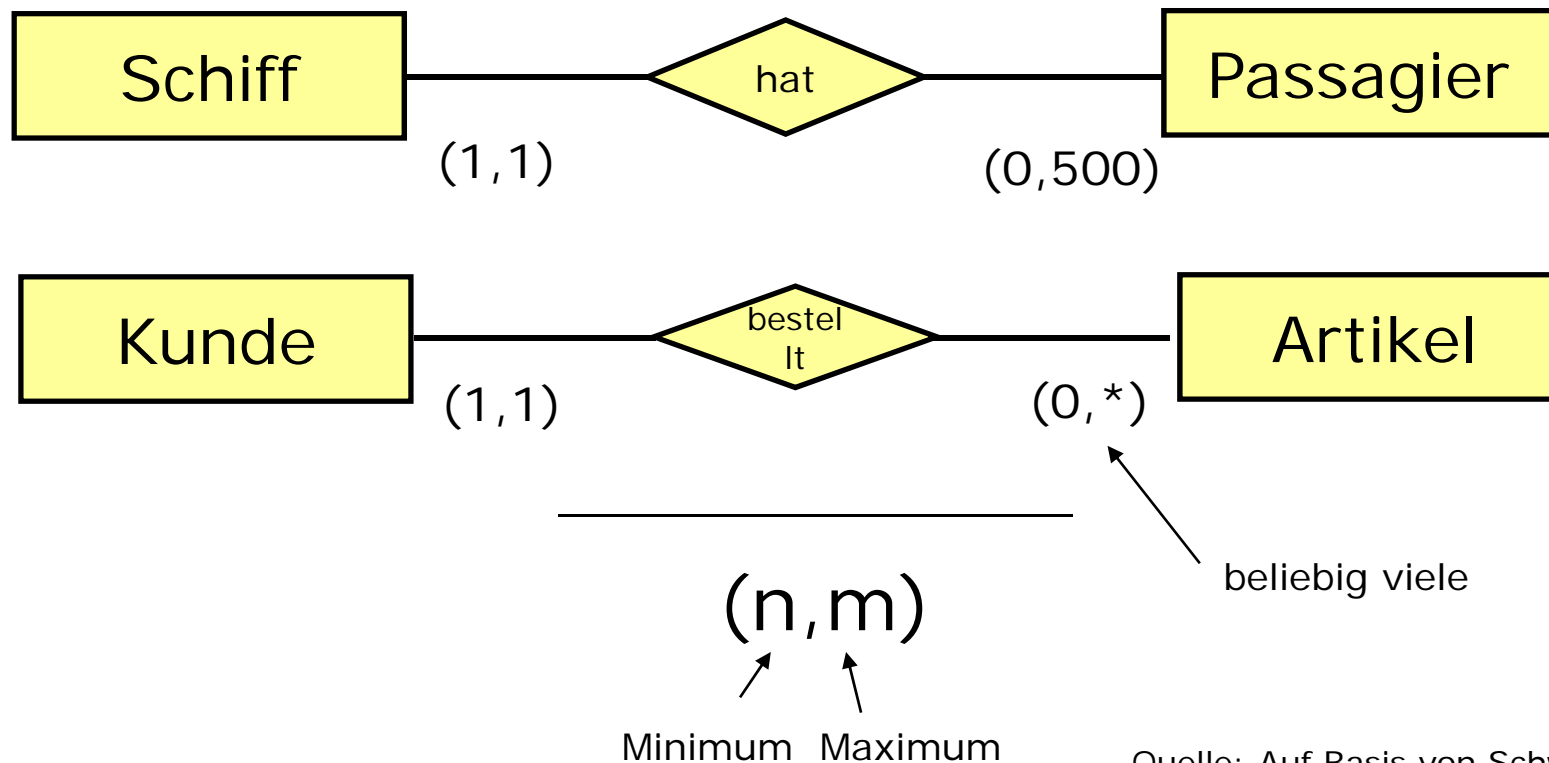
- Fortgeschrittene Elemente
 - Intervallangaben
 - Aggregation
 - Generalisierung
 - Rekursive Relationship-Typen
 - Schwache Entitäten

- Kardinalitäten werden mittels Intervallangaben genauer spezifiziert.

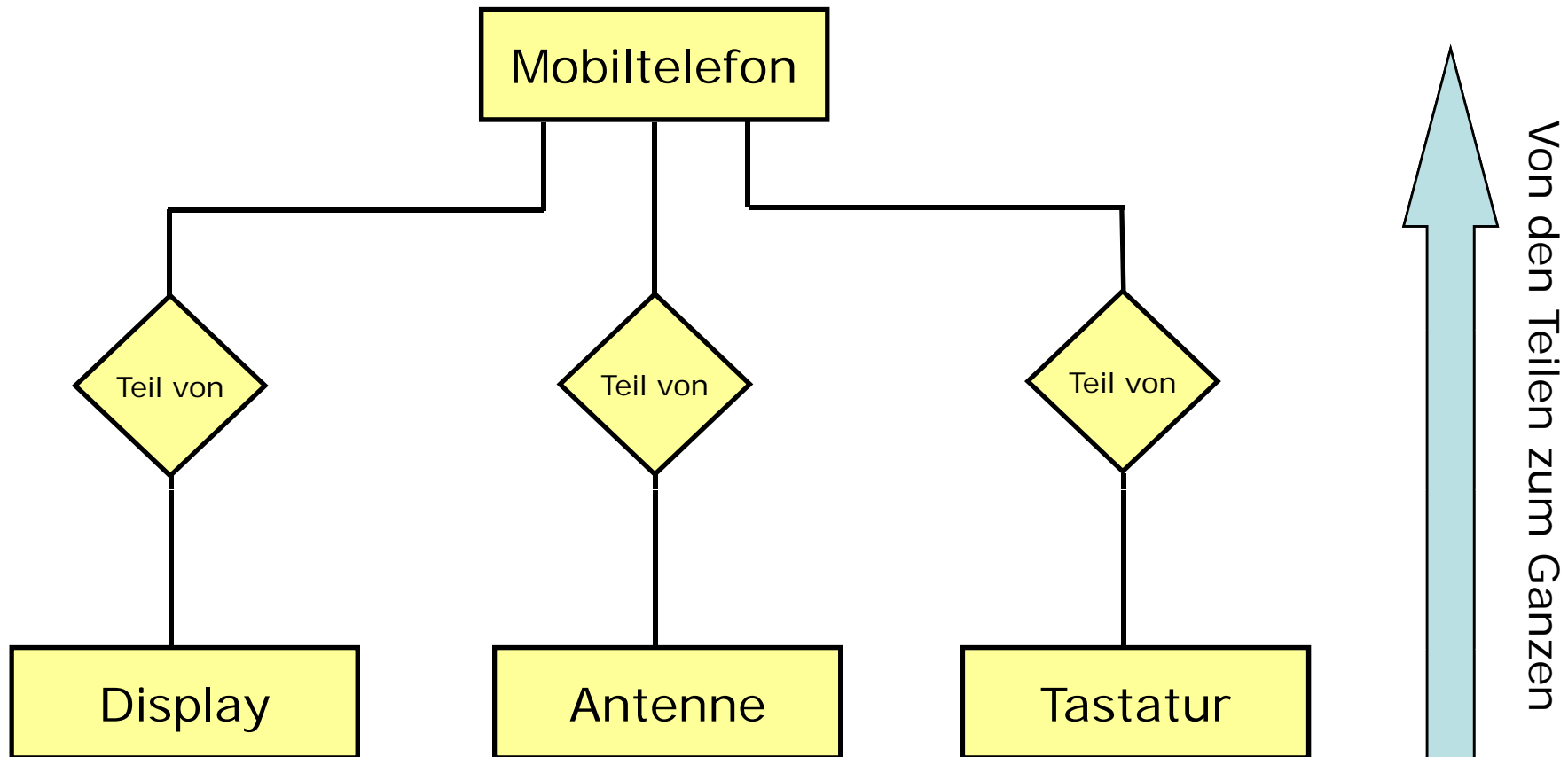


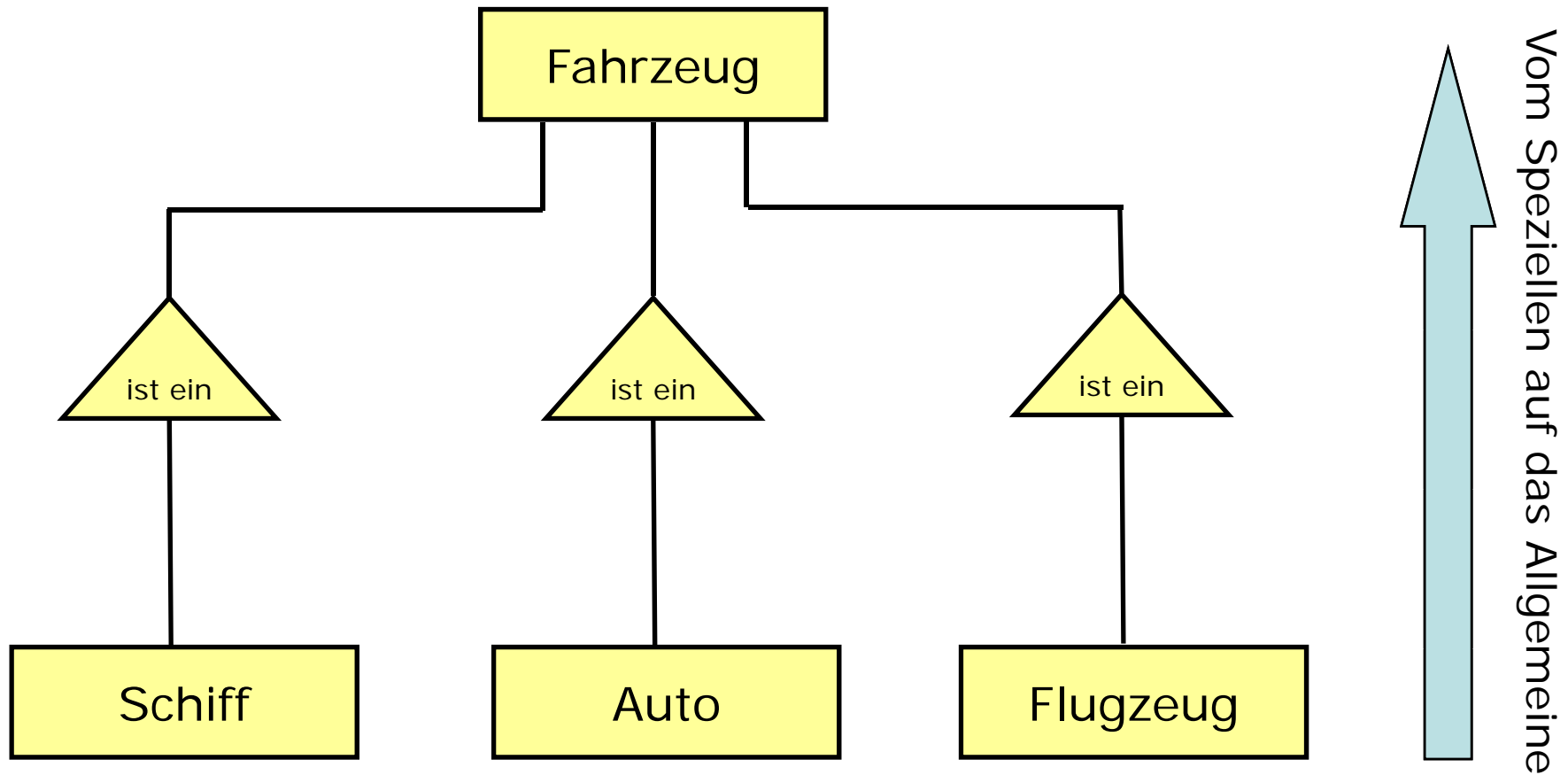
Quelle: Auf Basis von Ferstl; Sinz, 2001

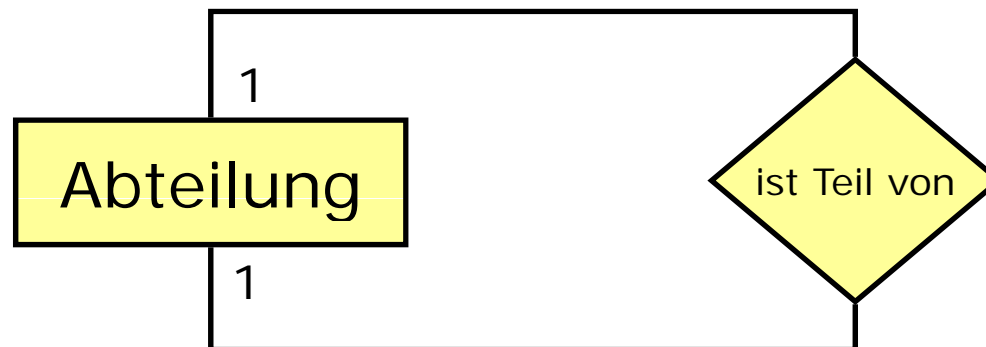
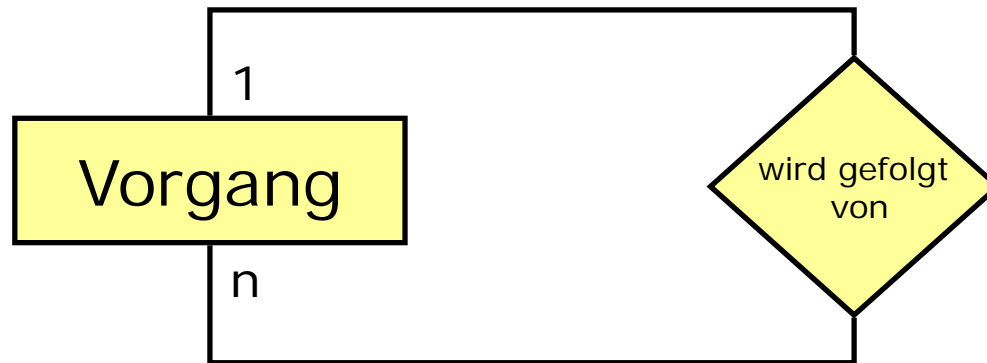
- Kardinalitäten werden mittels Intervallangaben genauer spezifiziert.



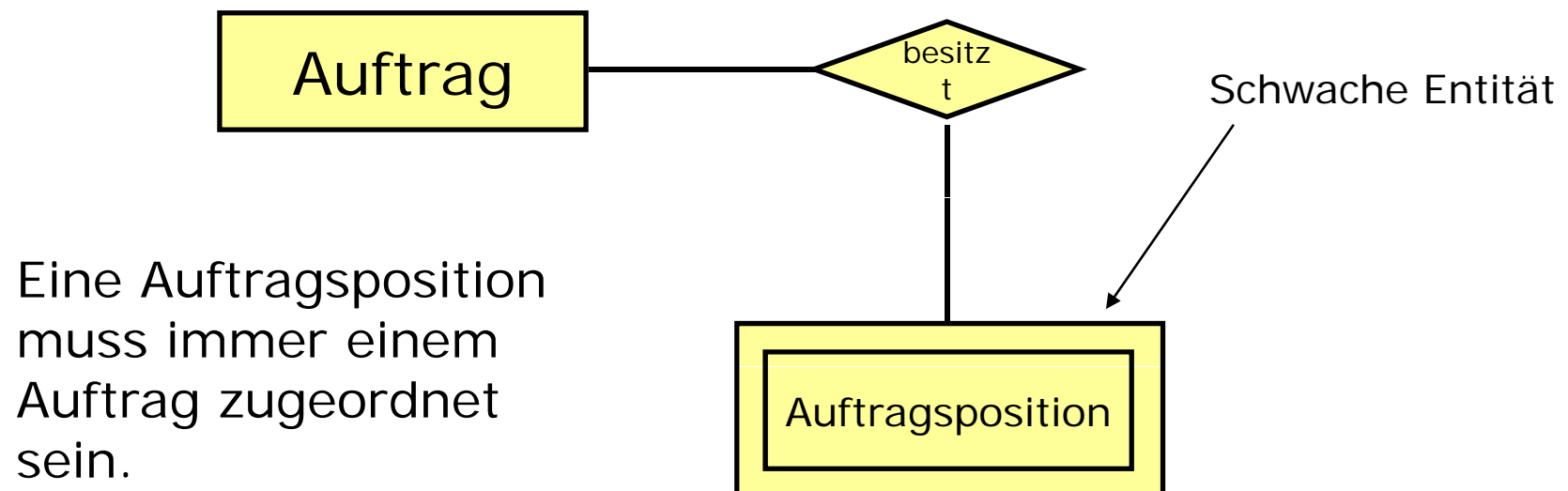
Quelle: Auf Basis von Schwickert, 2004



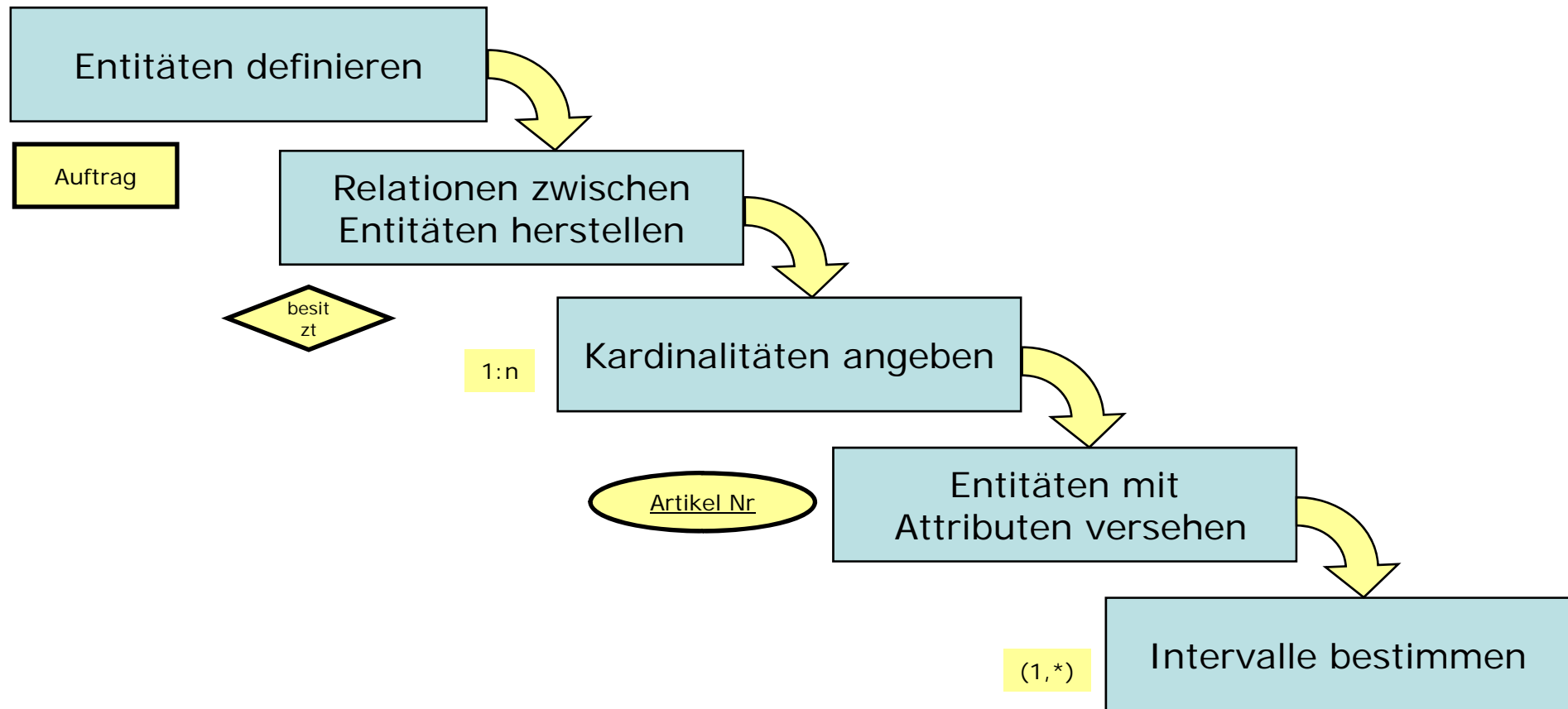




Schwache Entitäten sind abhängig von mindestens einer weiteren Entität und können somit nicht alleine existieren.



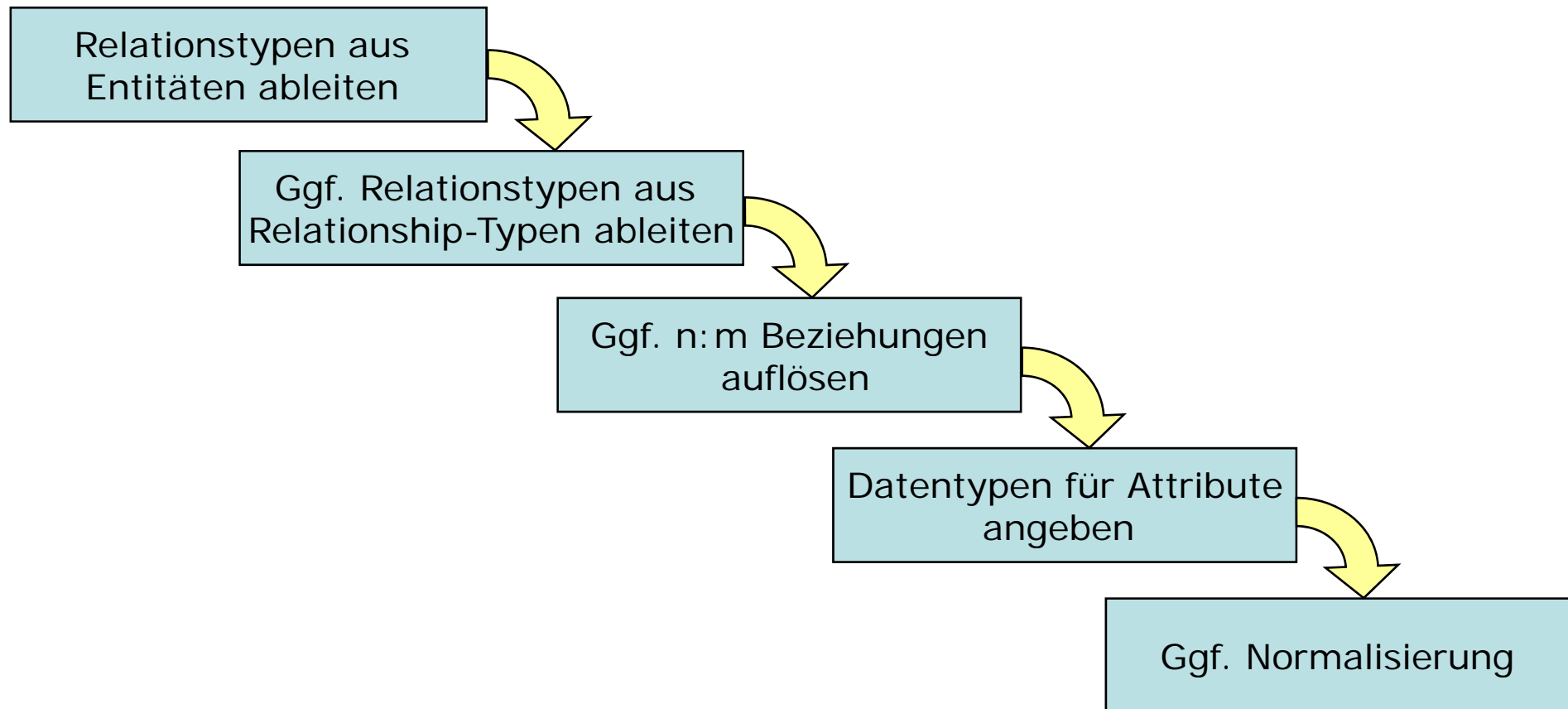
Vorgehen zur Erstellung eines ER-Modells (Bsp.)



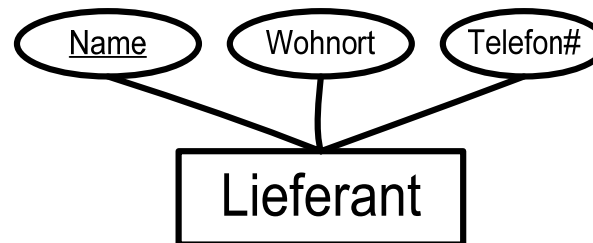
1. Objektorientierte Modellierung
 - 1.1 Grundkonzepte der Objektorientierung
 - 1.2 Unified Modeling Language (UML)
 - 1.3 Beschreibungselemente der UML
 - 1.4 UML im Einsatz

2. Prozessorientierte Modellierung
 - 2.1 Einführung in die Geschäftsprozessmodellierung
 - 2.2 Ereignis-gesteuerte Prozessketten
 - 2.2 Petri-Netze

3. Datenorientierte Modellierung
 - 3.1 Ansatz der datenorientierten Modellierung
 - 3.2 Fortgeschrittene ER-Modellierung
 - 3.3 Von ER-Modell zum Relationen-Modell



- Aus Entity-Typ wird Relationstyp mit den entsprechenden Attributen



Lieferant	<u>Name</u>	Wohnort	Telefon#

- Aus n:m-Relationship-Typ wird ein zusätzlicher Relationstyp.
 - Relation enthält die Schlüssel der beteiligten Entity-Typen als Attribute und zusätzlich die Attribute des Relationship-Typs.

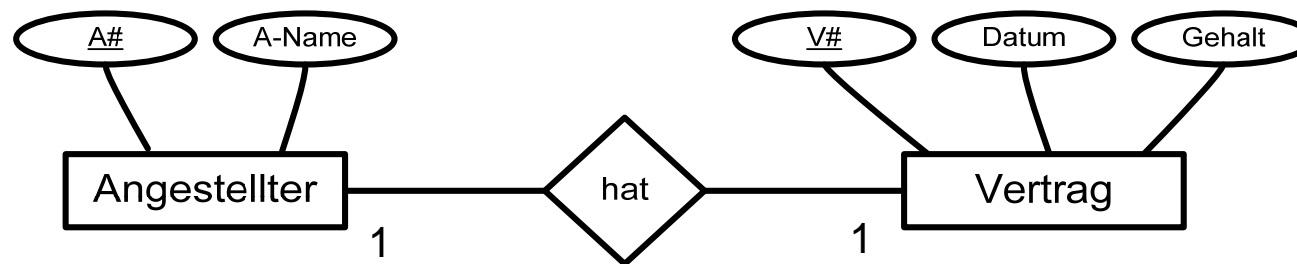
Beispiel:



bestellt	Kunden#	Artikel#	Datum	Menge

- Ein 1:1-Relationship-Typ wird i. allg. nicht zu einer eigenen Relation.
- Information wird an eine der beiden den betroffenen Entity-Typen entsprechenden Relationen „angehängt“.

Beispiel:



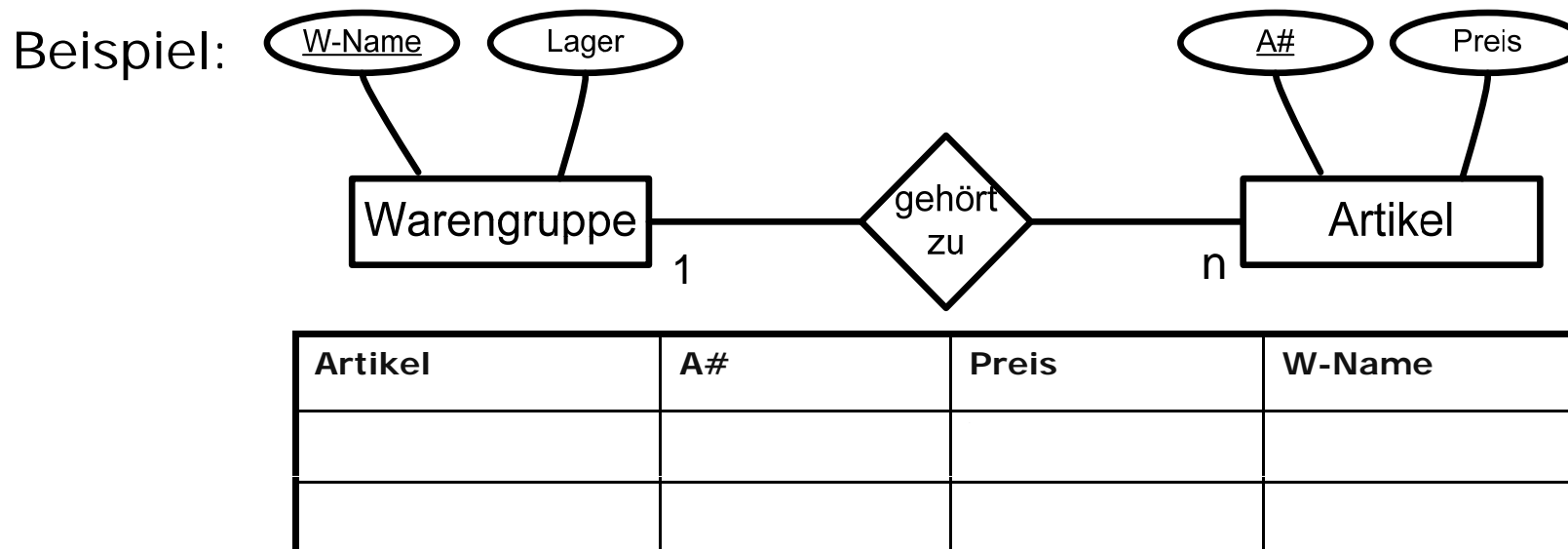
Alternative 1:

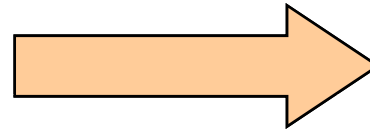
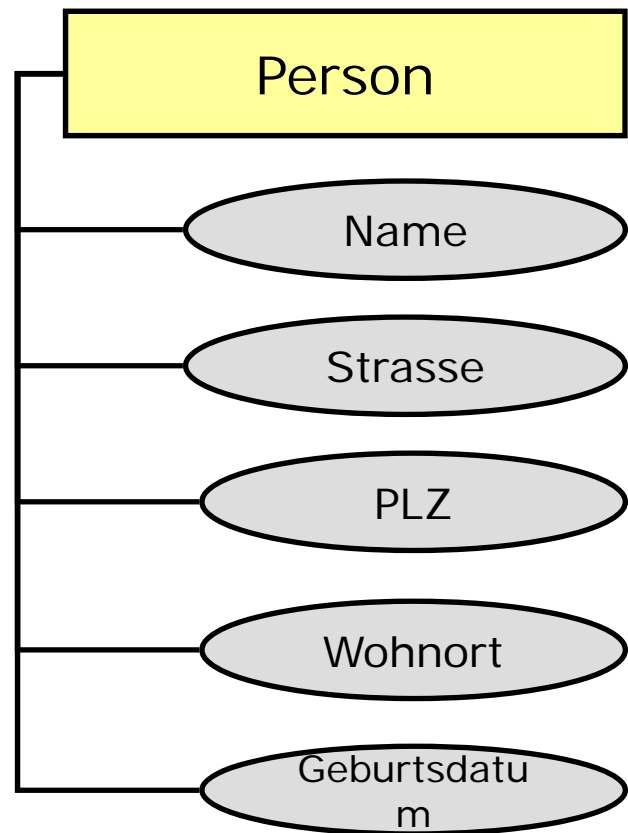
Angestellter	A#	A-Name	V#

Alternative 2:

Vertrag	V#	Datum	Gehalt	A#

- Ein 1:n-Relationship-Typ wird i. allg. nicht zu einer eigenen Relation.
- Information wird an die Relation "angehängt", die dem Entity-Typen an der mit n beschrifteten Kante entspricht.





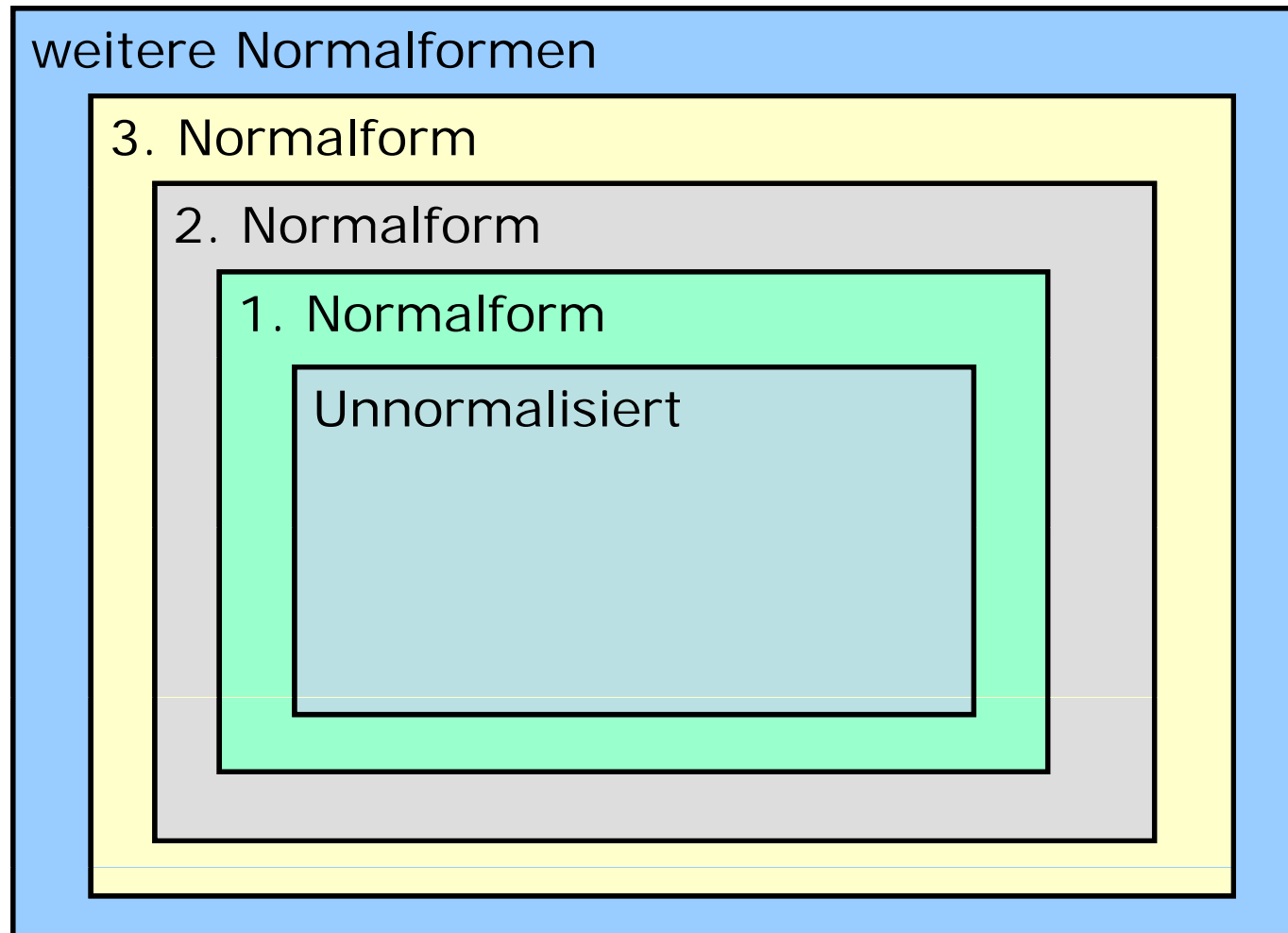
Relation „Person“

- Name: CHAR(30)
- Strasse: CHAR(40)
- PLZ: INT
- Wohnort: CHAR(30)
- Geburtsdatum: DATUM

- Ziele
 - Jede Relation enthält nur Daten einer Bedeutung.
 - Redundanzfreie Speicherung von Daten
 - Vermeidung von Anomalien verursacht durch Redundanzen und Inkonsistenzen

- Normalisierungsformen
 - Unnormalisierte Relationen
 - Normalisierte Relationen in der 1. - 3. Normalform (NF)
 - Weitere Normalformen (z.B. 4.+5. Normalform oder Boyce Codd)

- Vorgehen zur Normalisierung
 - Unnormalisiert -> 1. NF -> 2. NF -> 3. NF
 - Relationen abgeleitet aus ER-Modell sind bereits in 2. NF



Sign	Verfasser	Titel	Jahr	SID	Sachgebiet	Schlagwort
QH1	Ferstl, Sinz	Wirtschafts informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Ferstl	Wirtschafts -informatik	1993	GRD	Grundlagen	Objektmodellierung
QH1	Sinz	Wirtschafts -informatik	1993	GRD	Grundlagen	Objektmodellierung
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Algorithmus, Modul
QH3	Wirth	Pascal	1990	PRG	Programmierung	Algorithmus
QH3	Wirth	Pascal	1990	PRG	Programmierung	Datenstruktur

Quelle: Ferstl, Sinz 2001

Eine Relation ist unnormalisiert (0. NF), wenn Attribute mit nicht-elementaren Wertebereichen vorhanden sind.

- Redundanz
 - Z.B. Mehrfacheinträge für die Zuordnung des Buches „Wirtschaftsinformatik“ zu der Signatur „QH1“.

- Einfügeanomalie
 - Z.B. das Sachgebiet kann nur aufgenommen werden, wenn ein Buch hierfür vorliegt.

- Änderungsanomalie
 - Z.B. Änderungen am Titel eines Buches sind mehrfach durchzuführen.

- Löschanomalie
 - Z.B. mit dem Löschen des letzten Buches wird auch das Sachgebiet selbst gelöscht.

1. Normalform (1. NF)

Sign	Verfasser	Titel	Jahr	SID	Sachgebiet	Schlagwort
QH1	Ferstl	Wirtschafts-informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Sinz	Wirtschafts-informatik	1993	GRD	Grundlagen	Informationssysteme
QH1	Ferstl	Wirtschafts-informatik	1993	GRD	Grundlagen	Objektmodellierung
QH1	Sinz	Wirtschafts-informatik	1993	GRD	Grundlagen	Objektmodellierung
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Algorithmus
QH2	Wirth	Modula-2	1989	PRG	Programmierung	Modul
QH3	Wirth	Pascal	1990	PRG	Programmierung	Algorithmus
QH3	Wirth	Pascal	1990	PRG	Programmierung	Datenstruktur

Quelle: Ferstl; Sinz, 2001

Eine Relation ist in 1. NF, wenn die Wertebereiche aller Attribute elementar sind.

- Vereinbarungen von Bezeichnungen
 - $R(A_1, A_2, \dots, A_n)$ Relationstyp
 - A_1, A_2, \dots, A_n Attribute
 - $X, Y, Z \subseteq \{A_1, A_2, \dots, A_n\}$

- Funktionale Abhängigkeit
 - Y heißt *funktional abhängig* von X in R (X bestimmt Y funktional) $X \rightarrow Y$, wenn es in keiner Relation zwei Attribute gibt, die in ihrem Wert zu X , aber nicht in ihrem Wert zu Y übereinstimmen.

- Voll funktionale Abhängigkeit
 - Y heißt *voll funktional abhängig* von X in R (X bestimmt Y voll funktional) $X \bullet \rightarrow Y$, wenn $X \rightarrow Y$ gilt und X minimal ist, d.h. wenn es keine Attributmenge $Z \subset X$ gibt, so dass $Z \rightarrow Y$.

- Schlüsselkandidat
 - X heißt Schlüsselkandidat von R , falls $X \bullet \rightarrow A_1, A_2, \dots, A_n$.
 - Ein Schlüsselattribut ist ein Bestandteil eines Schlüsselkandidaten.

Quelle: Auf Basis von Ferstl; Sinz, 2001

- Funktionale Abhängigkeit (siehe Abbildung 1. Normalform)
 - Beispiele
 - Sign, Verfasser, Schlagwort → Titel, Jahr, SID, Sachgebiet
 - Sign, Verfasser → Titel, Jahr, SID, Sachgebiet
 - Bei Kenntnis von „Sign, Verfasser, Schlagwort“ bzw. „Sign, Verfasser“ lassen sich die zugehörigen Werte „Titel, Jahr, SID, Sachgebiet“ eindeutig bestimmen.

- Voll funktionale Abhängigkeit (siehe Abbildung 1. Normalform)
 - Beispiele
 - Sign •→ Titel, Jahr, SID, Sachgebiet
 - SID •→ Sachgebiet
 - „Sign“ ist die minimale Attributmenge, um „Titel, Jahr, SID, Sachgebiet“ eindeutig zu bestimmen. „Verfasser“ oder „Schlagwort“ allein genommen würden nicht ausreichen, um „Titel, Jahr, SID, Sachgebiet“ eindeutig zu bestimmen.
 - „SID“ ist die minimale Attributmenge, um „Sachgebiet“ eindeutig zu bestimmen.

- Schlüsselkandidaten (siehe Abbildung 1. Normalform)
 - Schlüsselkandidaten sind Attributkombinationen, von denen alle Attribute der Relation voll funktional abhängig sind.
 - Beispiel
 - Sign, Verfasser, Schlagwort •→ Sign, Verfasser, Titel, Jahr, SID, Sachgebiet, Schlagwort
 - Bei Kenntnis von „Sign, Verfasser, Schlagwort“ kann man **alle** Werte der Relation „Titel, Jahr, SID, Sachgebiet“ eindeutig bestimmen.

- Schlüsselattribute (siehe Abbildung 1. Normalform)
 - Ein Attribut, das Bestandteil eines Schlüsselkandidaten ist, heißt Schlüsselattribut.
 - Beispiel
 - Sign, Verfasser, Schlagwort sind Schlüsselattribute
 - Titel, Jahr, SID, Sachgebiet sind Nichtschlüsselattribute

- Aufspaltung der Relation in 1. NF in zwei neue Relationen in 2. NF
 - Die Relation in 1. NF ist so aufzuteilen, dass
 - alle Attribute aus der Relation in 1. NF auch in den Relationen in 2. NF vertreten sind;
 - in jeder der Relationen in 2. NF alle Nichtschlüsselattribute voll funktional vom Schlüsselkandidaten abhängig sind.
 - Es kann dabei Relationen geben, die nur aus Schlüsselattributen bestehen.
 - Die Relationen sind über ein Schlüsselattribut, welches in beiden Relationen vorhanden ist (Fremdschlüssel) miteinander zu verknüpfen.
 - Beispiel
 - Sign •→ Titel, Jahr, SID, Sachgebiet
 - Sign, Verfasser, Schlagwort •→ Sign, Verfasser, Schlagwort
 - „Sign“ ist der Fremdschlüssel.

2. Normalform (2. NF)

- Die Relation R1 ist in 2. Normalform, wenn sie in 1. NF ist und jedes Nichtschlüsselattribut (Titel, Jahr, SID, Sachgebiet) von dem Schlüsselkandidaten („Sign“) voll funktional abhängig ist.
- Die Relation R2 ist in 2. Normalform, wenn sie in 1. NF ist und jedes Nichtschlüsselattribut (-) von dem Schlüsselkandidaten („Sign, Verfasser, Schlagwort“) voll funktional abhängig ist.

- R1 -

<u>Sign</u>	Titel	Jahr	SID	Sachgebiet
QH1	Wirtschaftsinformatik	1993	GRD	Grundlagen
QH2	Modula-2	1989	PRG	Programmierung
QH3	Pascal	1990	PRG	Programmierung

- R2 -

<u>Sign</u>	<u>Verfasser</u>	<u>Schlagwort</u>
QH1	Ferstl	Informationssysteme
QH1	Sinz	Informationssysteme
QH1	Ferstl	Objektmodellierung
QH1	Sinz	Objektmodellierung
QH2	Wirth	Algorithmus
QH2	Wirth	Modul
QH3	Wirth	Algorithmus
QH3	Wirth	Datenstruktur

Anmerkung:
Schlüsselattribute sind unterstrichen.

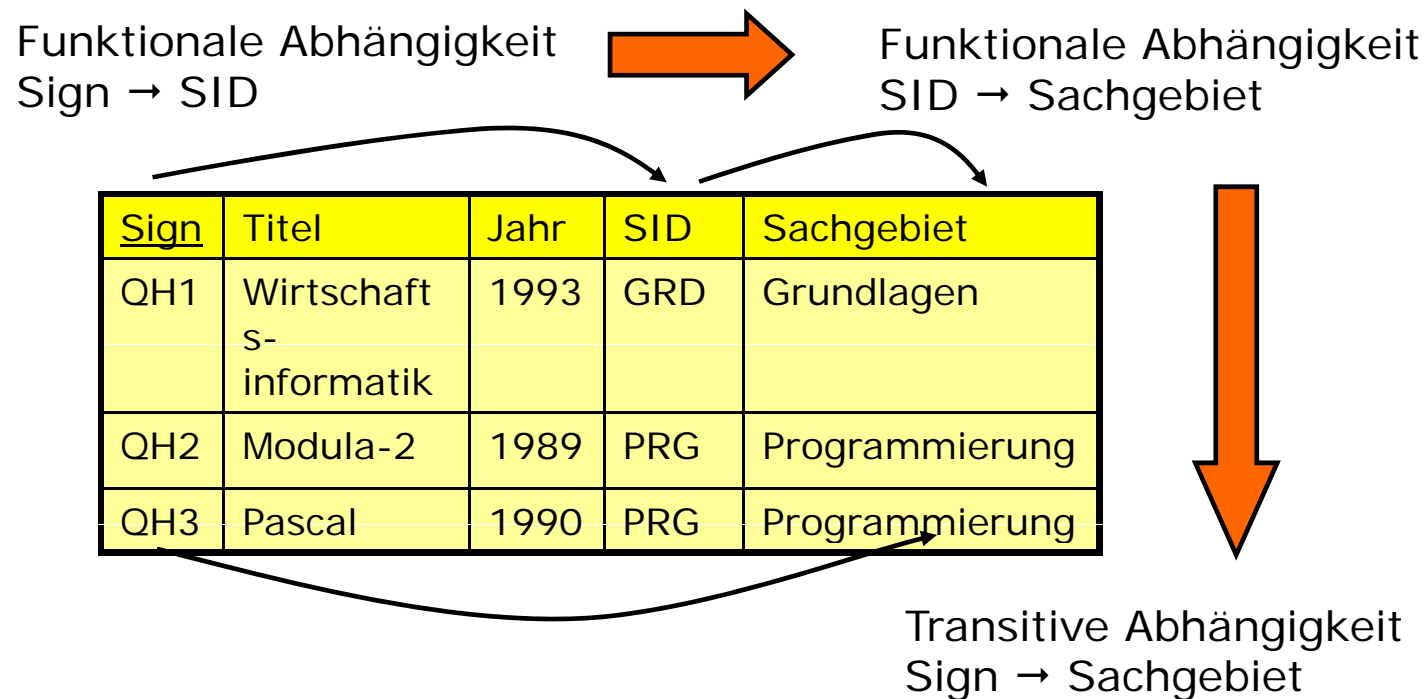
Quelle: Auf Basis von Ferstl; Sinz, 2001

3. Normalform

Eine Relation R ist in 3. Normalform, wenn sie in 2. NF ist und kein Nichtschlüsselattribut transitiv von einem Schlüsselkandidaten abhängt.

Quelle: Auf Basis von Ferstl; Sinz, 2001

Relation in 2. Normalform, da ein Nichtschlüsselattribut (**Sachgebiet**) transitiv von einem Schlüsselkandidaten (**Sign**) abhängt.



Quelle: Auf Basis von Ferstl; Sinz, 2001

3. Normalform

<u>Sign</u>	Titel	Jahr	SID
QH1	Wirtschafts-informatik	1993	GRD
QH2	Modula-2	1989	PRG
QH3	Pascal	1990	PRG

<u>SID</u>	Sachgebiet
GRD	Grundlagen
PRG	Programmierung

Daher Aufteilung der Relationstypen in

- Buch (Sign, Titel, Jahr, SID)
- Sachgebiet (SID, Sachgebiet)

Quelle: Auf Basis von Ferstl; Sinz, 2001

- In der Praxis wird i.d.R. nur die 1.-3. Normalform verwendet.
- Anzahl der Tabellen und die Komplexität eines Relationenmodells steigen mit höherer Normalform.
- Höhere Normalform senkt die Performance eines Datenbanksystems.
- Im praktischen Einsatz wird ein Trade-Off zwischen Normalform und Performance eines Datenbanksystems angestrebt.

- Ferstl, O. K.; Sinz, E. J. (2001) "Grundlagen der Wirtschaftsinformatik", 4. Auflage, München.
- Schwickert, A. (2004) "Modellierung von IuK-Systemen", Universität Gießen.