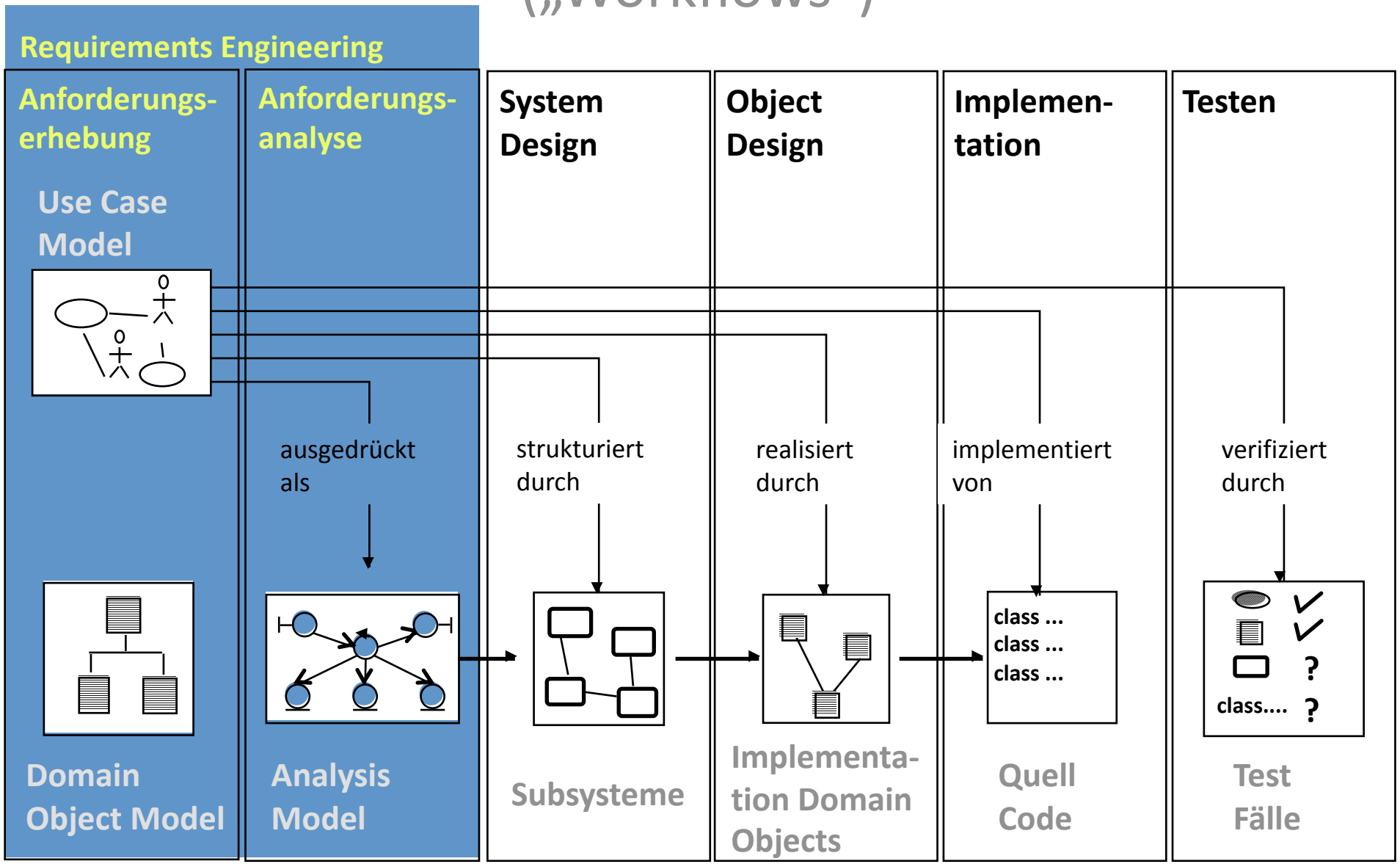
Workflows:
Anforderungserhebung- und -analyse

Tutorium 4

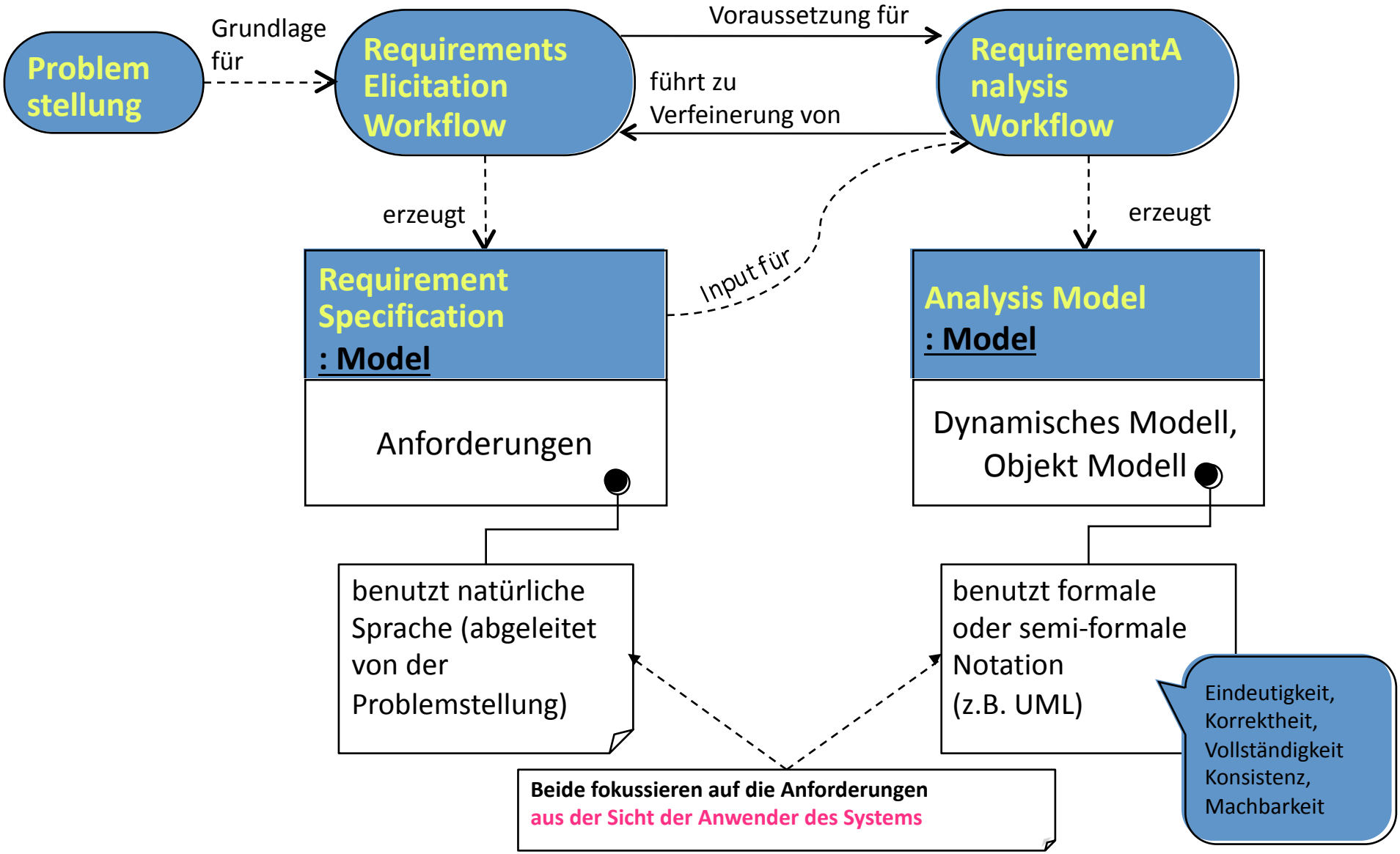
9. März 2009

Aktivitäten bei der Softwareentwicklung („Workflows“)



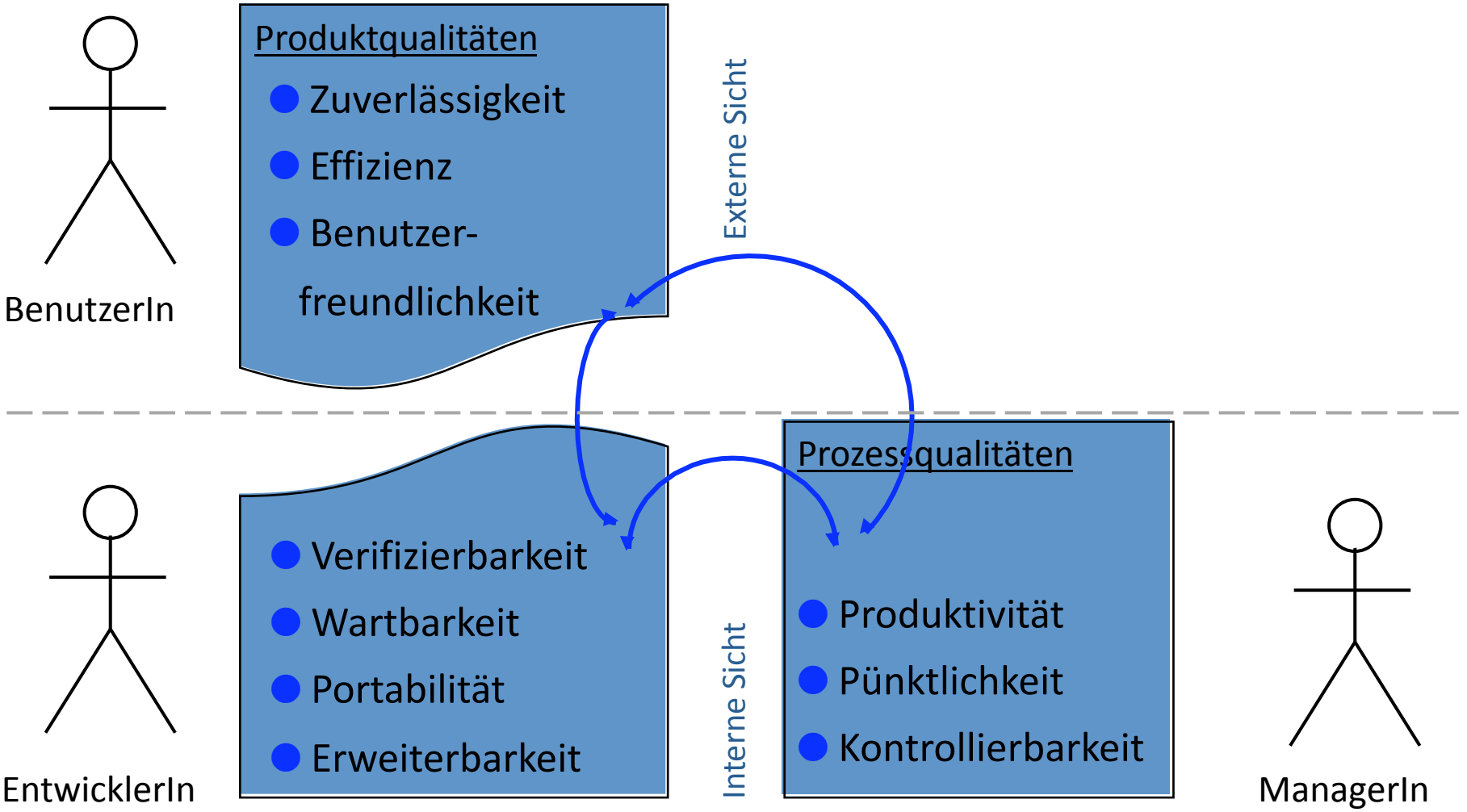
* Die erzeugten dynamischen Modelle (und Andere) sind hier aus Platzgründen nicht explizit dargestellt.

Der Requirements Engineering Prozess



Anforderungen

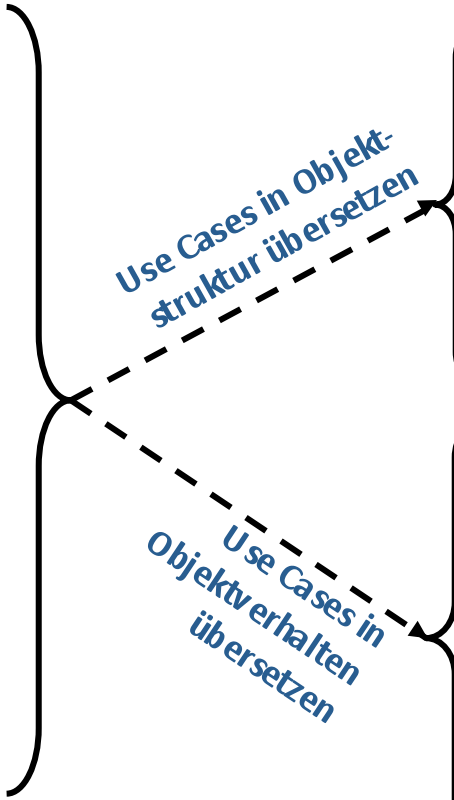
Verschiedene Sichten



Aktivitäten bei der Anforderungserhebung und -analyse

Anforderungserhebung

- Identifiziere Akteure
- Identifiziere Szenarien
- Identifiziere Use Cases
- Verfeinere Use Cases
- Identif. Beziehungen zwischen Use Cases
- Identif. nichtfunktionale Anforderungen
- Identifiziere Neben-bedingungen
- Identifiziere beteiligte Objekte
- Erstelle Glossar



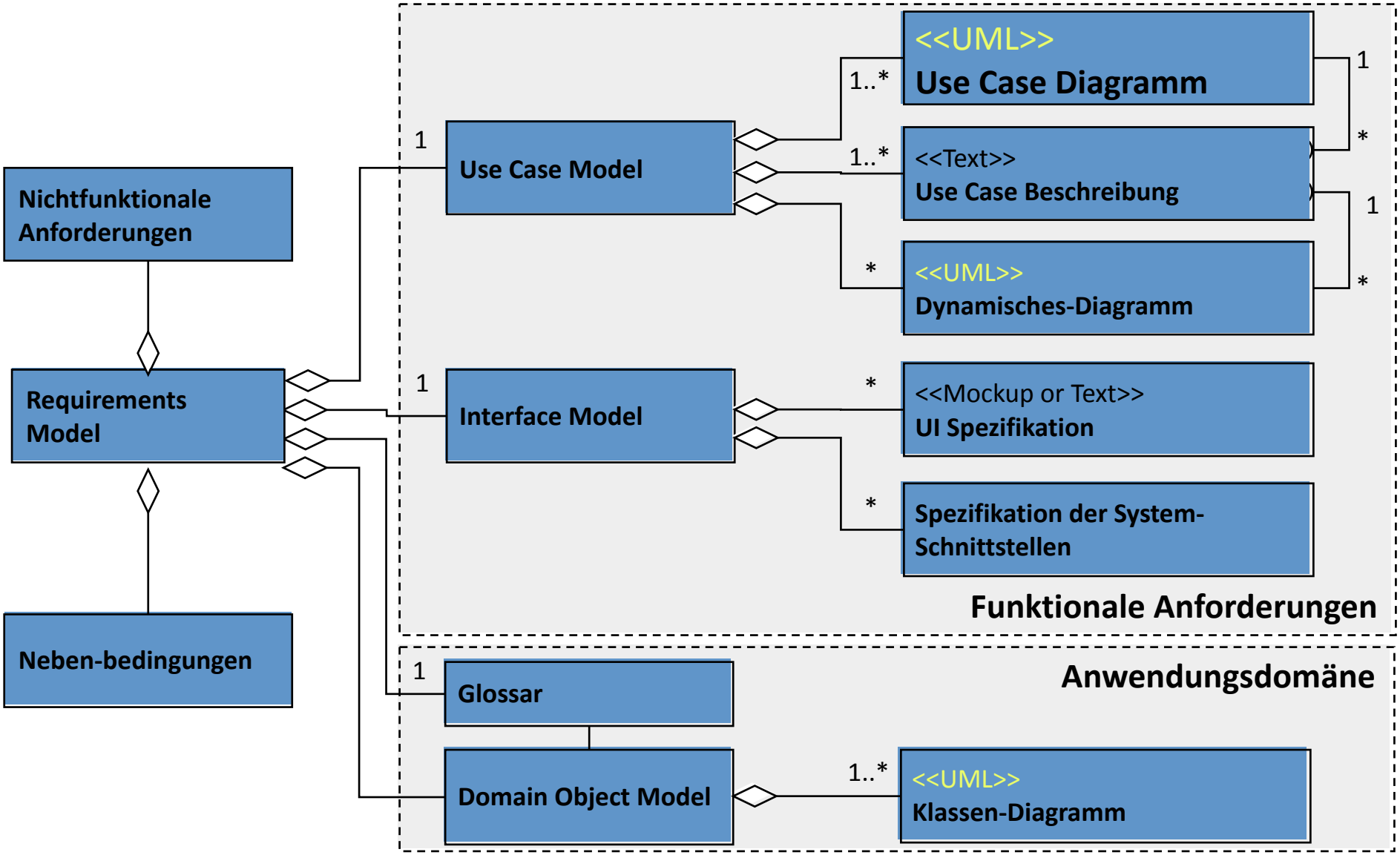
Anforderungsanalyse

- Identifiziere Objekte/Klassen
  - boundary, controller, entity
- Identifiziere Operationen und Methoden
- Identifiziere Assoziationen
- Identifiziere Attribute

- Modelliere Objektinteraktionen
  - Kollaborationsdiagramme
  - Sequenzdiagramme
- Modelliere nichttriviales internes Verhalten
  - Zustandsdiagramme

Von beobachteter
Mehrdeutigkeit, Unkorrektheit, Unvollständigkeit, Inkonsistenz, Undurchführbarkeit
zur Verfeinerung der Anforderungen

Produkte der Anforderungserhebung



Anforderungen

Typen

- Funktionale Anforderungen
- Nichtfunktionale Anforderungen
- Nebenbedingungen (“Pseudo requirements”)

Anforderungserhebung

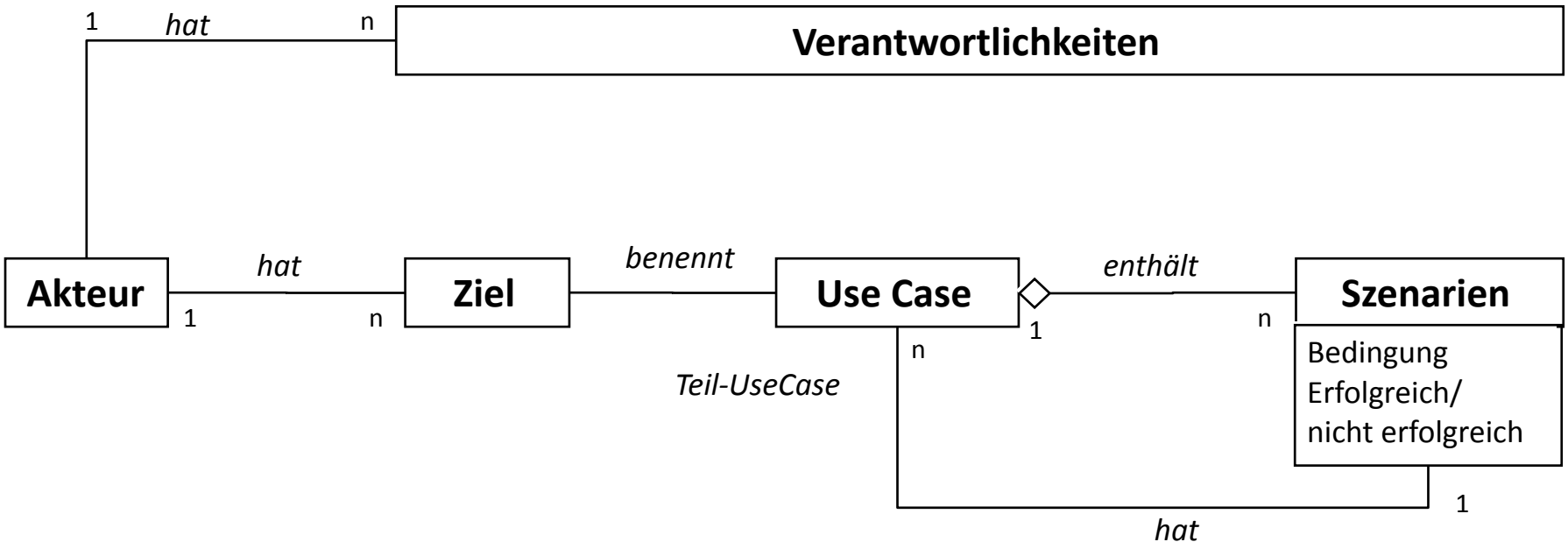
Arten

- Greenfield Engineering („Planung auf der grünen Wiese“)
- Reengineering
- Interface Engineering

Akteur – Szenario - Use Case

Beziehungen

- Ein Akteur hat Ziele
- Use Cases sind nach den Zielen benannt
- Jeder Use Case fasst eine Menge von Szenarien zusammen
- Ein Szenario kann sich auf Teil-Use Cases beziehen.



Use Case

Bestandteile

- Name
- Akteure
  - Beschreibung der am Use Case beteiligten Akteure
- Anfangsbedingung / Vorbedingung
  - Nutze einen syntaktischen Ausdruck wie “Dieser Use Case beginnt, wenn…”
- Ereignisfluss
  - Formlos, informelle natürliche Sprache
- Endbedingung / Nachbedingung
  - Beginnt mit “Dieser Use Case endet, wenn…”
- Ausnahmen
  - Beschreibe was passiert, wenn etwas schief geht inkl. der entsprechenden Nachbedingungen.
- Spezielle Anforderungen
  - Nichtfunktionale Anforderungen und Nebenbedingungen

Funktionale und Dynamische Modellierung

Beschreibung von Use Cases

- strukturierter Text
- Szenarien
    - Konzentration auf einen wesentlichen Ablauf (ein Szenario)
    - Sequenz- oder Kollaborationsdiagramm, Zustandsdiagramm
- Use Case Modell
    - Use-Case Diagramm
- Dynamisches Modell
    - Aktivitätsdiagramm

Anwendungsfalldiagramm

(use-case diagram)

Alternative dt. Bez.: *Use-Case Diagramm*

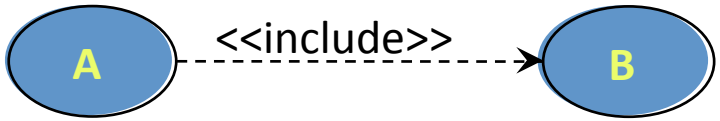
Funktionalität des zu entwickelten Softwaresystems aus Benutzersicht

Assoziationstypen:

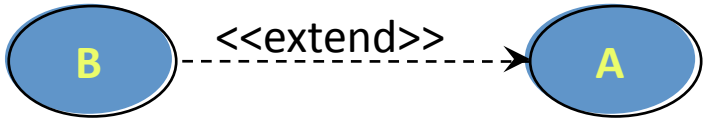
- Include
- Extend
- Generalisierung

Use-Case Diagramm

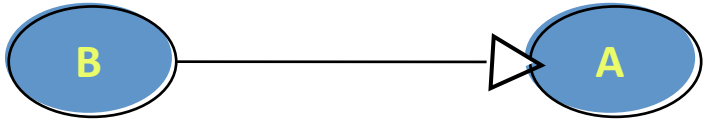
- **Include-Beziehung**
  - Wenn immer A ausgeführt wird, muss auch B ausgeführt werden
  - B ist unbedingt nötig, um A auszuführen
  - Für Verhalten, das von mehreren genutzt wird



- **Extend-Beziehung**
  - Wenn A ausgeführt wird, kann auch B ausgeführt werden
  - B ist nicht zwingend nötig, um A auszuführen
  - Für Sonderfälle, optionales oder seltenes Verhalten

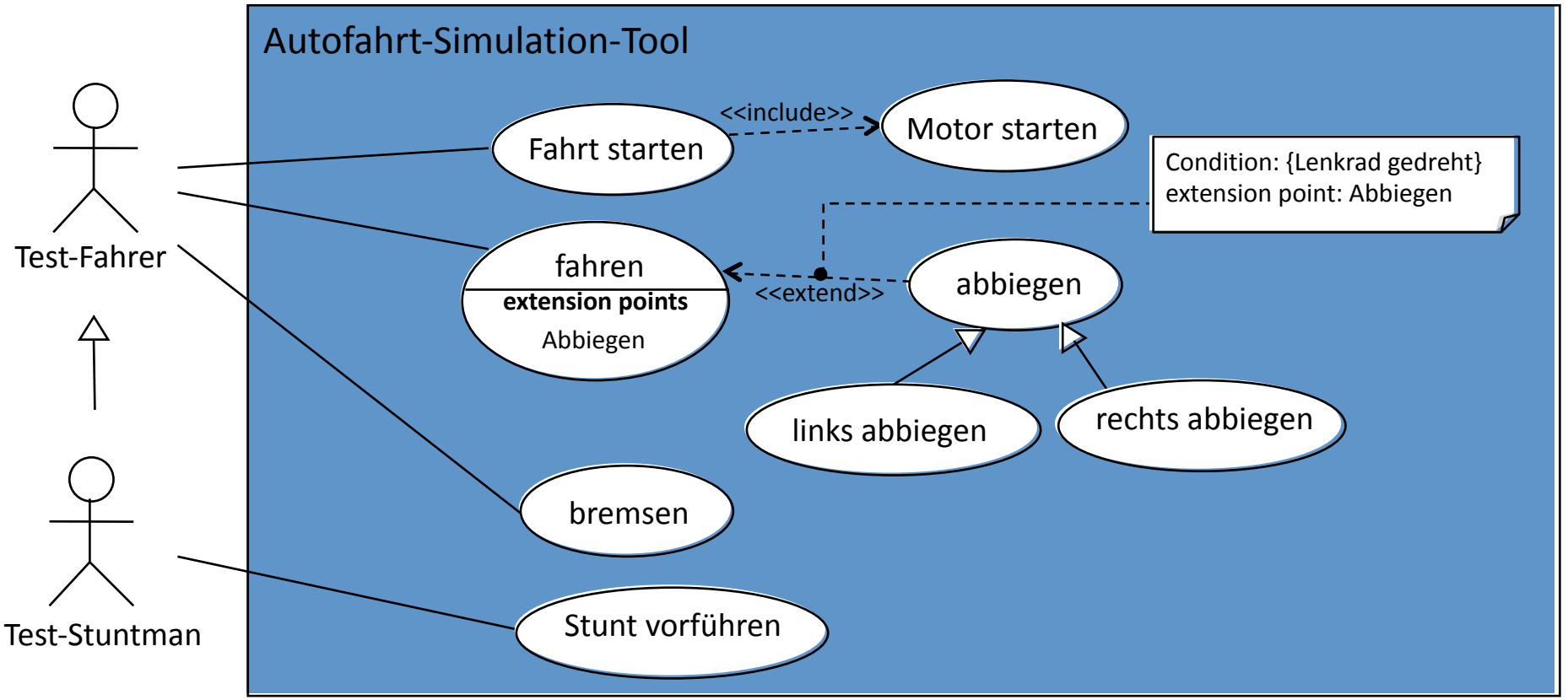


- **Generalisierungsbeziehung**
  - B erbt das ganze Verhalten von A
  - Zur Modellierung von Vererbung



Use-Case Diagramm

Beispiel



Objektmodellierung

Domain Object Model (DOM)

- Klassendiagramm für Konzepte der Anwendungsdomäne
  - Klassen
  - Attribute
  - Beziehungen
  - (wenige Operationen)

- Vorgehensweise
  - Dialog mit Benutzer
  - Hauptwörter-Erfassung und –Filterung
    → Verfahren von Abbott

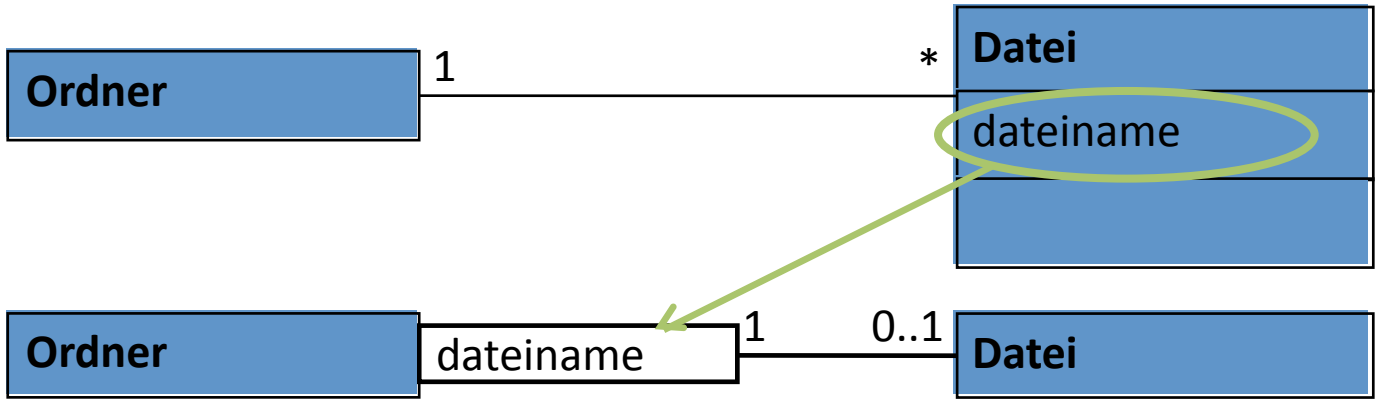
Objektmodellierung im Analyse-Workflow

Qualifikation

- Ein Qualifier präzisiert die Multiplizitätsangaben einer Assoziation zwischen Klassen.
- Er wird benutzt, um 1-n Multiplizitäten auf 1-1 Multiplizitäten zu reduzieren

Ohne Qualifikation:

Ein Verzeichnis enthält viele Dateien. Eine Datei gehört zu genau einem Verzeichnis.



Mit Qualifikation:

Ein Verzeichnis enthält viele Dateien, jede mit einem einmaligen Namen

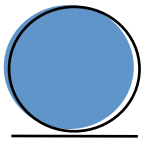
Objektmodellierung im Analyse-Workflow

Analyseklassen

Eine *Analyse-Klasse* ist stets einer der drei folgenden Stereotypen:

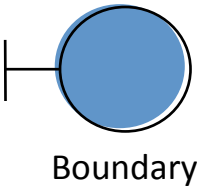
Entity Objekte

- Repräsentierendie für das Anwendungsgebiet relevante Konzepte („Business objects“)
- entsprechen of den persistenten Informationen der Anwendung.



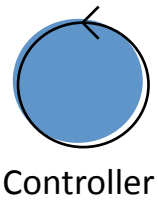
Boundary Objekte

- Repräsentieren die Interaktion zwischen Akteuren und System

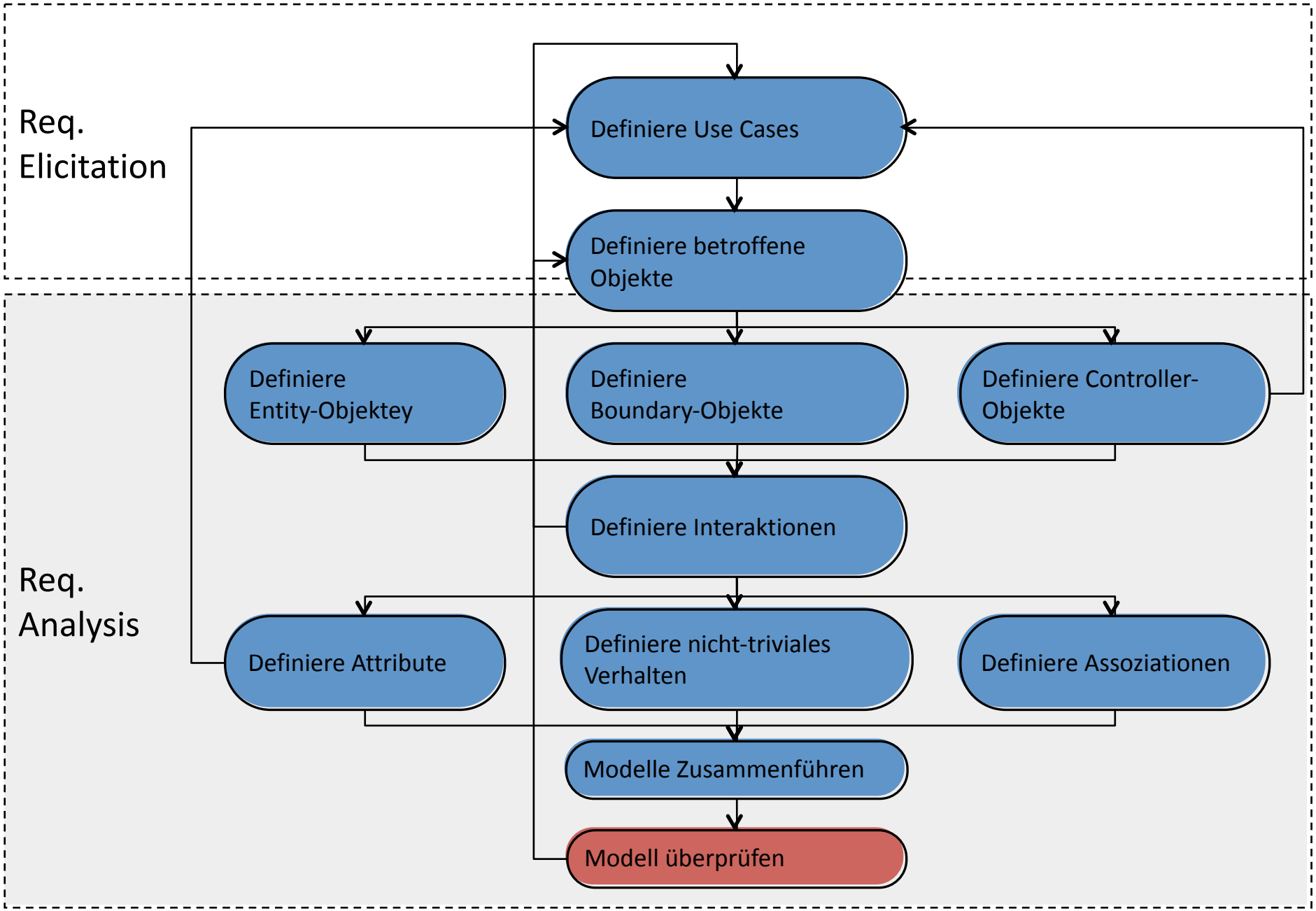


Control Objekte

- Repräsentieren die vom System ausgeführten Kontrollaufgaben / Abläufe / Use Cases



Aktivitätsdiagramm des Analyse-Workflow



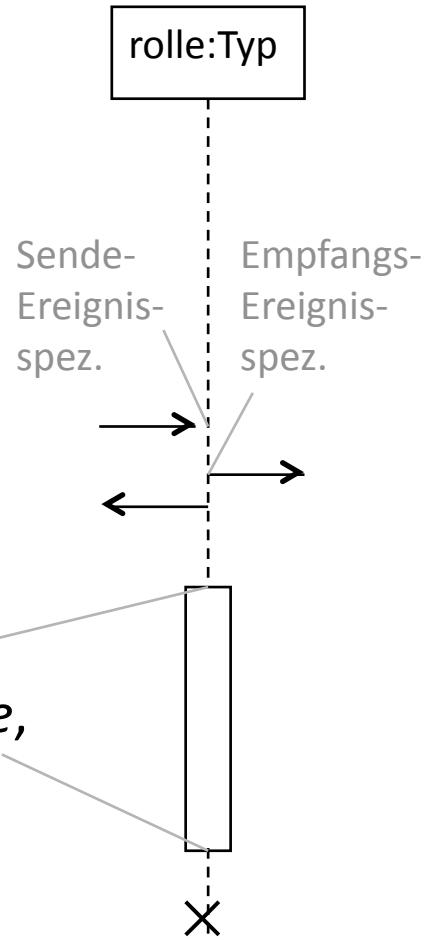
Dynamische Modellierung

UML

- Interaktionsdiagramm
  - beschreibt das Zusammenspiel von Objekten
  - Sequenzdiagramm
    - Verhalten einer Menge von Objekten in zeitlicher Abfolge
    - Gut für die Spezifikation von Echtzeitsystemen
  - Kollaborationsdiagramm
    - Zeigt auch die Beziehungen zwischen Objekten.
    - Zeit wird implizit über Nummerierung der Nachrichten dargestellt
- Zustandsdiagramm
  - beschreibt das Verhalten eines einzelnen Objektes
  - Ein endlicher Automat zur Beschreibung der Reaktion eines Objektes auf äußere Einwirkung (Events).
- Aktivitätsdiagramm
  - beschreibt beliebige Abläufe incl. Fallunterscheidungen, Nebenläufigkeit, Synchronisation

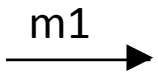
Sequenzdiagramm (UML Wdh.)

- Lebenslinie
  - stellt Interaktionspartner dar
  - erstreckt sich über die gesamte »Lebenszeit« des zur Laufzeit an die Rolle gebundenen Objekts
- Interaktionen
  - als Abfolge von *Ereignisspezifikationen*
- Ausführung eines Verhaltens innerhalb einer Interaktion wird
  - durch 2 Ereignisspezifikationen: *Start* und *Ende*, auf der gleichen Lebenslinie definiert
  - *Ausführungsspezifikation* genannt
    - In UML 1: *Aktivierung (focus of control)*

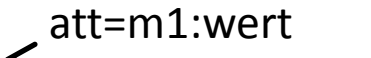


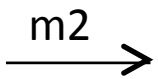
Sequenzdiagramm (UML Wdh.)

Nachrichten

- Ausführung einer Nachricht
- Versenden einer Nachricht
  - synchron 

    - Antwortnachricht 

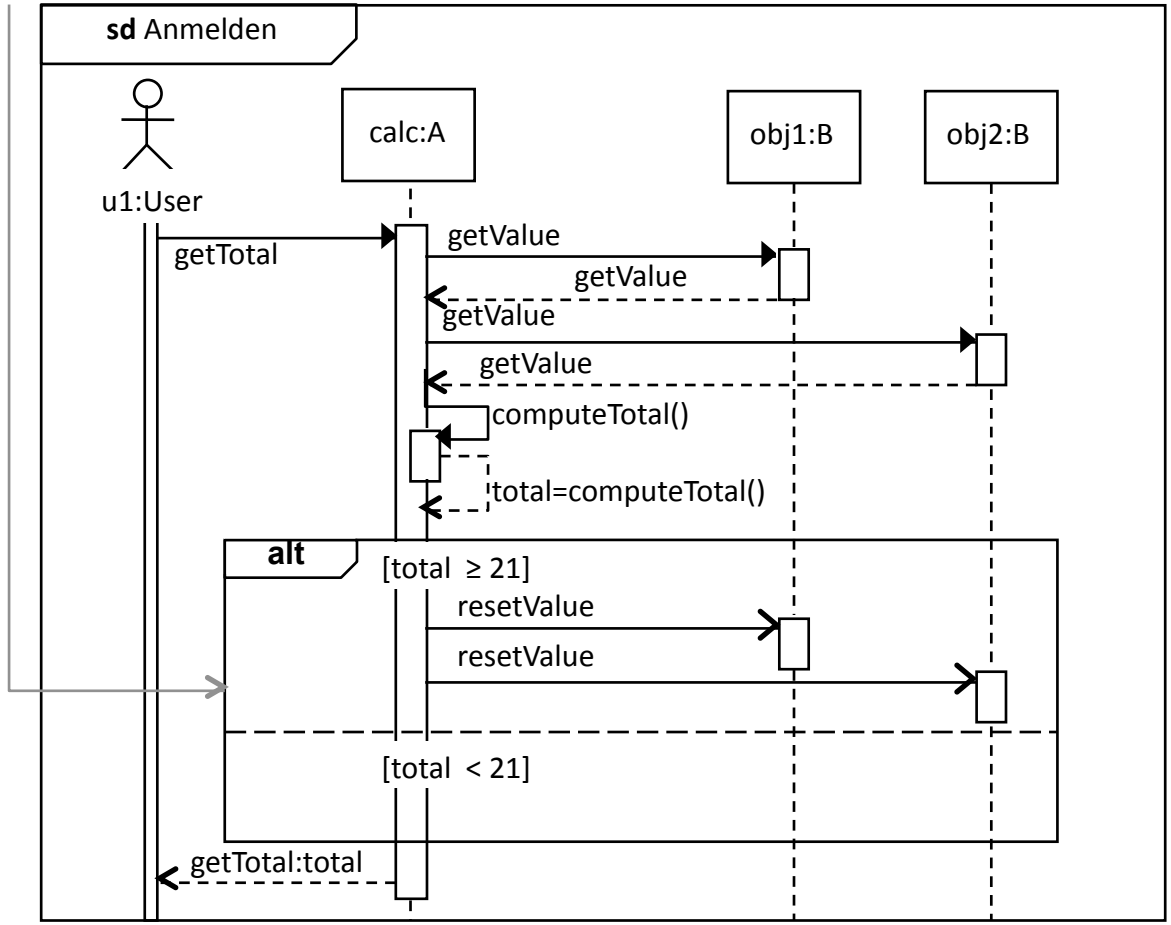
    - Rückgabewerte 

  - asynchron 

Sequenzdiagramm (UML Wdh.)

Kontrollflusssteuerung

Kombinierte Fragmente



Beispiele:
Verzweigung (alt, opt, break)
Schleife (loop)

Quellenangabe

- Dr. Kniesel, *Vorlesungsfolien zu Softwaretechnologie*, WS 2008.
- Bernd Bügge, Allen H. Dutoit, *Objektorientierte Softwaretechnik mit UML, Entwurfsmustern und Java*, Pearson Studium, 2004.
- Douglas Bell, *Software Engineering for Students. A Programming Approach*, 4th Edition, Pearson Education Limited, 2005.