UML
Unified Modelling Language

Dipl.-Inform. Christian Fuß

Dieses Tutorial basiert auf einem Internet-Tutorial von Prof. Dr.-Ing. R. Dumke an der Universität Magdeburg, es ist zu finden unter der URL http://www-ivs.cs.uni-magdeburg.de/~dumke/UML

RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

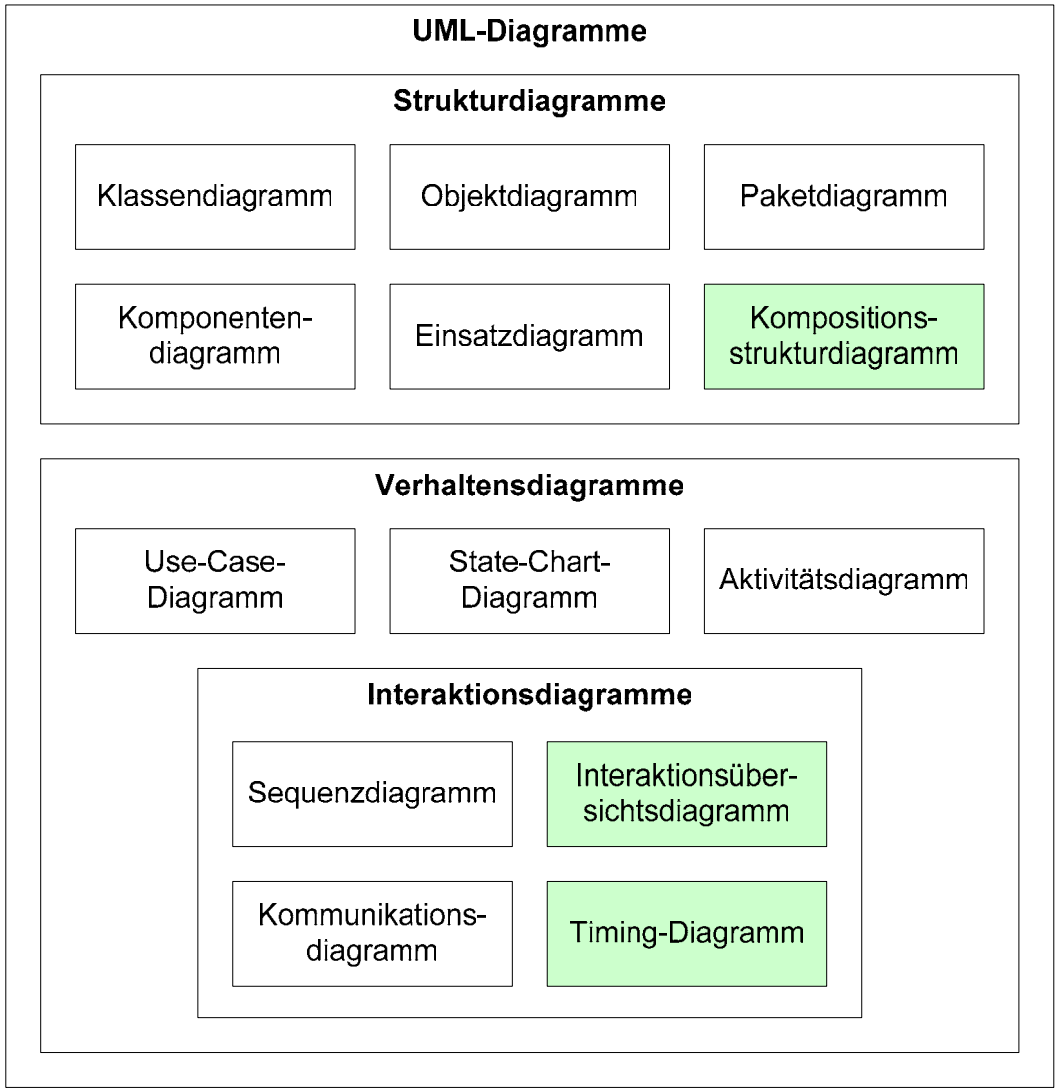
Einführung

Object-Orientation

Defining abstractions of real-world entities known as objects, which contain both data and procedures. Key characteristics of object-oriented systems are encapsulation, inheritance and polymorphism.

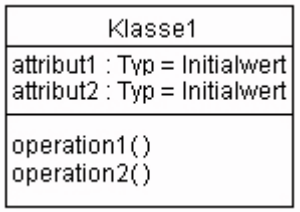
- UML ist eine Sammlung von Diagrammsprachen für den objektorientierten Entwurf
- Spezifikation, Visualisierung, Konstruktion und Dokumentation von Modellen
- Softwaresysteme, Geschäftsmodelle und andere Nicht-Softwaresysteme
- OMG Standard seit 1998, derzeit aktuell UML 2
- Entwickelt von Grady Boch, Ivar Jacobsen und Jim Rumbaugh

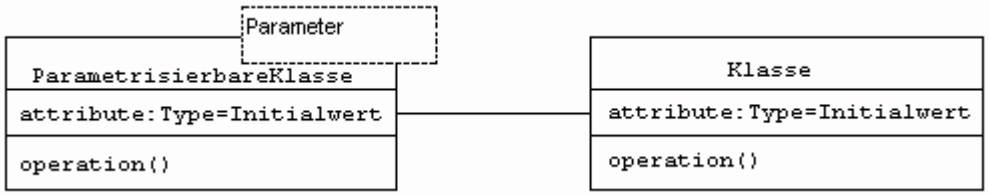
Überblick


UML-Diagramme
Strukturdiagramme
Klassendiagramm
Objektdiagramm
Paketdiagramm
Komponenten-diagramm
Einsatzdiagramm
Kompositions-strukturdiagramm
Verhaltensdiagramme
Use-Case-Diagramm
State-Chart-Diagramm
Aktivitätsdiagramm
Interaktionsdiagramme
Sequenzdiagramm
Interaktionsüber-sichtsdiagramm
Kommunikations-diagramm
Timing-Diagramm

Klassendiagramm – Klassen

- Klasse ist Konstruktionsbeschreibung für Objekte
  - Attribute
  - Operationen (Methode=Implementierung der Operation)
- Abstrakte Klassen
  - Grundlage für Unterklassen
  - nicht instanzierbar
- Parametrisierbare Klassen
  - Schablone zur Erzeugung von Klassen
  - Erzeugte Klasse muss mit <<bind>>-Assoziation mit Schablone verbunden werden





RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Klassendiagramm – Attribute, Operationen

`attribut : Klasse = Initialwert {Merkmal}{Zusicherung}`

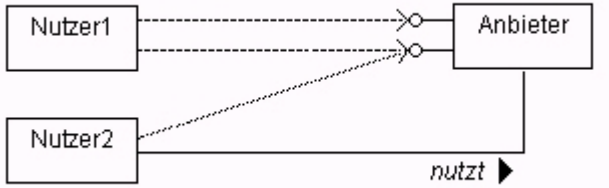
- Datenelement jedes Objekts einer Klasse
  - /abgeleitetes Attribut (nach Berechnungsvorschrift)
  - klassenattribut
  - +publicAttribut
  - #protectedAttribut
  - -privateAttribut
- Klasse beschreibt Typ des Attributs
- Merkmal beschreibt weitere Eigenschaften, z.B. read-only
- Zusicherung schränkt Inhalte, Zustände oder Semantik eines Modellelements ein

`operation(argument:Argumenttyp=Standardwert,...) : Rückgabetyp {Merkmal} {Zusicherung}`

- vergleichbar den Attributen
- Operationen sind Dienstleistungen, die aufgerufen werden können

Klassendiagramm – Schnittstellen

- Schnittstellen spezifizieren Teil des äußerlich sichtbaren Verhaltens von Modellelementen
- beinhalten eine Menge von Signaturen für Operationen
- symbolisiert durch nicht ausgefüllten Kreis, der durch Linie mit anbietender Klasse verbunden ist
- Name der Schnittstelle entspricht dem Namen der Schnittstellenklasse
- Nutzung durch Abhängigkeitsbeziehung

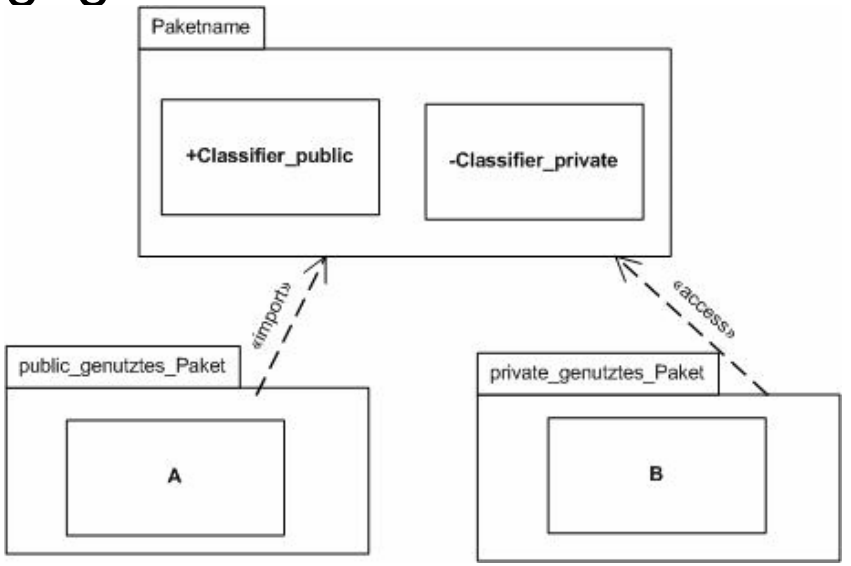


Klassendiagramm – Stereotypen

- erweitern vorhandene Modellelemente des UML-Metamodells
- Element wird direkt durch definierte Semantik beeinflusst
- besitzen keine Typsemantik
- sollten projekt-, unternehmens- oder methodenspezifisch vergeben werden

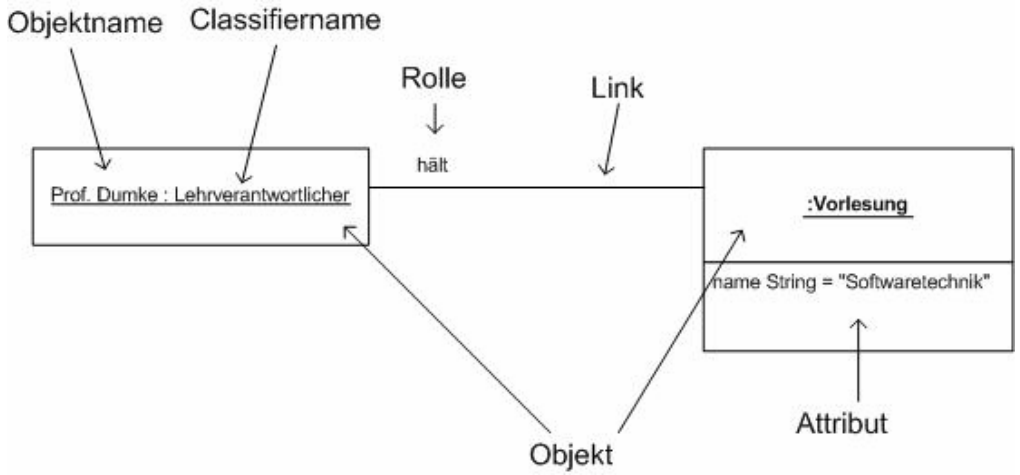
Klassendiagramm – Pakete

- Ansammlungen von Modellelementen
- Gesamtmodell in kleine überschaubare Einheiten untergliedern
- definieren keine Modellsemantik
- jedes Modellelement gehört zu genau einem Paket
- können hierarchisch gegliedert werden



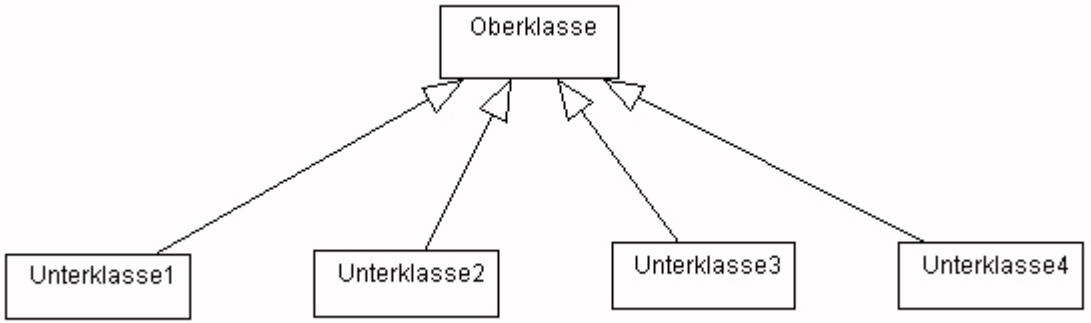
Klassendiagramm – Objekte

- interessierende Objekte und Beziehungen in einer Art Momentaufnahme
- z.B. Datensatz für den Test zu definieren Systemablauf während des Debugging erwünschte oder unerwünschte Zustände
- Objektdiagramm bildet konkrete Instanz des abstrakt beschriebenen Modells



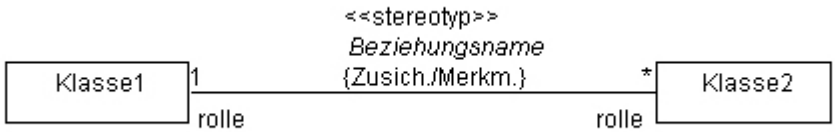
Klassendiagramm – Vererbung

- Programmiersprachenkonzept für Relation zwischen Ober- und Unterklasse
- Attribute und Operationen der Oberklasse sind auch Unterklassen zugänglich
- Abstraktionsprinzip zur hierarchischen Gliederung der Semantik eines Modells
- Unterscheidung eines Diskriminators: z.B. Antriebsart für PKWs



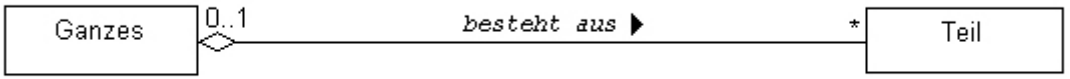
Klassendiagramm – Assoziation

- Assoziationen beschreiben Verbindungen zwischen Klassen
- Objekte können nur dann miteinander kommunizieren
- konkrete Beziehung (Instanz einer Assoziation) wird Objektverbindung (*Link*) genannt
- an den Enden kann Multiplizität angegeben werden (als einzelne Zahl oder Wertebereich)
- eine abgeleitete Assoziation wird nicht gespeichert, sondern bei Bedarf berechnet



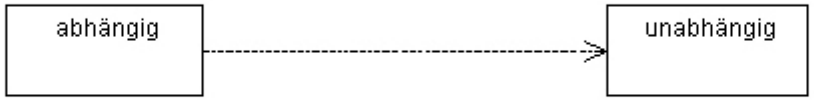
Klassendiagramm – Aggregation, Komposition

- Aggregation ist Zusammensetzung eines Objektes aus einer Menge von Einzelteilen
- Aggregat nimmt stellvertretend für Teile Aufgaben wahr
- Aggregat kann Nachrichten an seine Teile weiterleiten
- die beteiligten Klassen führen keine gleichberechtigte Beziehung
- Teil kann zu mehreren Aggregationen gehören
- Komposition ist existenzabhängige Aggregatbeziehung



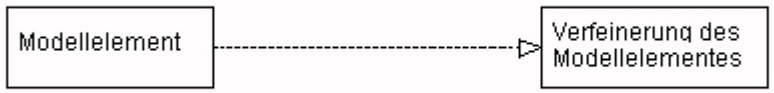
Klassendiagramm – Abhängigkeit

- Beziehung zwischen Modellelementen, die zeigt, dass Änderung des einen (unabhängigen) Elements Änderung im anderen (abhängigen) Element bewirkt
- bezieht sich dabei auf Modellelemente selbst, nicht auf Instanzen



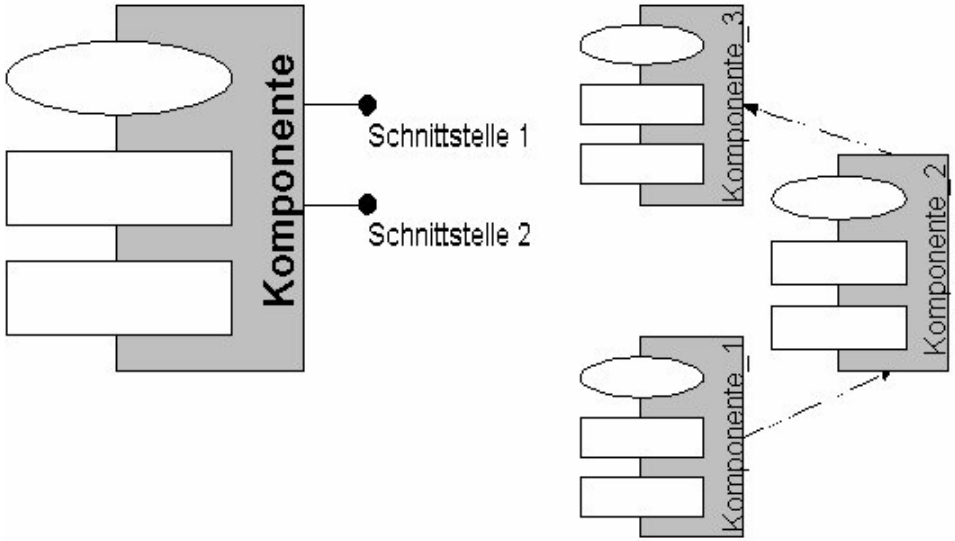
Klassendiagramm – Verfeinerung

- Verfeinerungen sind Beziehungen zwischen gleichartigen Elementen unterschiedlichen Detaillierungsgrades
- Beispiele
  - Beziehung von Analyse- und Designversion
  - Beziehung von sauberer und optimierter Variante
  - Beziehung zwischen zwei unterschiedlich granulierten Elementen
  - Beziehung zwischen Schnittstellenklasse und umsetzender Klasse



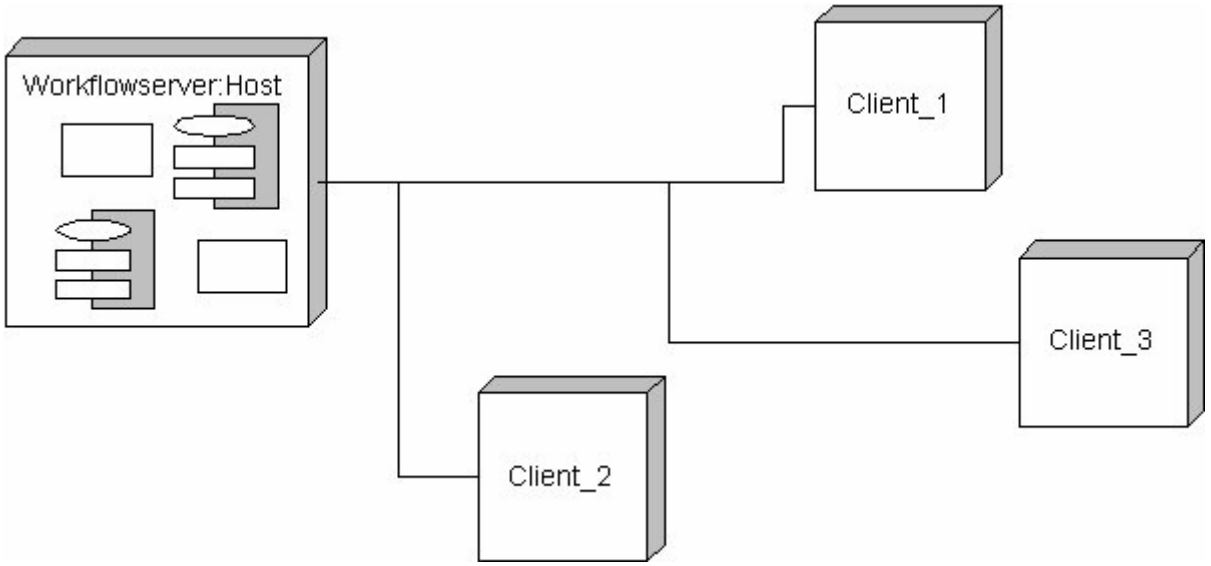
Komponentendiagramm

- Komponente stellt physisches Stück Programmcode dar
- Komponente kann Elemente (Objekte, Komponenten, Knoten) enthalten und Schnittstellen besitzen
- Komponentendiagramm zeigt Abhängigkeiten zwischen Komponenten



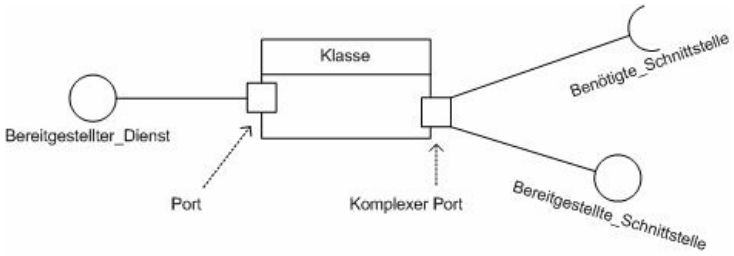
Einsatzdiagramm

- Knoten stellen Laufzeitumgebungen dar
- Knoten werden als Quader visualisiert
- In den Knoten werden dort installierte Komponenten und Objekte dargestellt
- Knoten die miteinander kommunizieren werden durch Linien miteinander verbunden

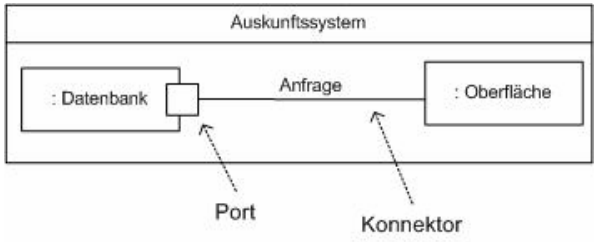


Kompositionsstrukturdiagramm UML2

- dient der internen hierarchischen Strukturierung von Komponenten (Klassen)
- Interne Strukturierung geschieht auf Instanzebene (Zusammenfassung zur Laufzeit)
- Externe Struktur kann durch Ports verfeinert werden
- Ports bündeln geforderte und gelieferte Schnittstellen mit logischen Interaktionen



Externe Portanwendung

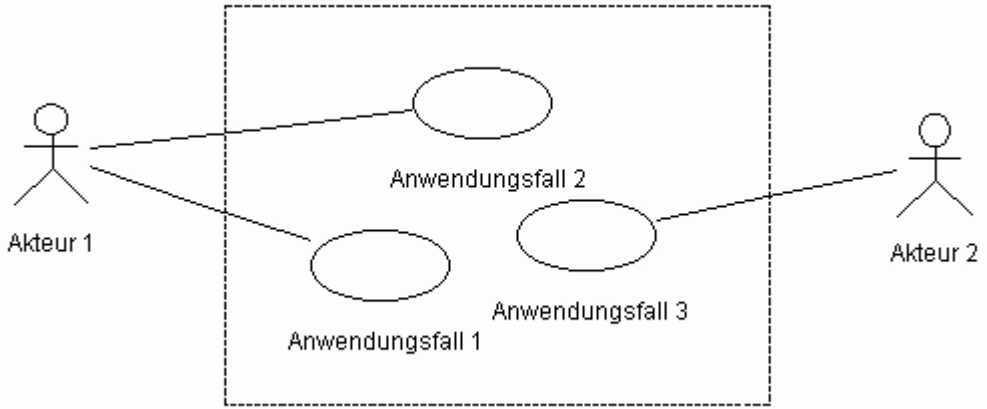


Interne Portanwendung

RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Use-Case-Diagramm

- stellt Beziehungen zwischen Akteuren und Anwendungsfällen dar
- Anwendungsfall beschreibt äußerlich sichtbares Systemverhalten, kann hierarchisch geschachtelt werden
- Akteur befindet sich außerhalb der Systemgrenze
- Systemgrenze wird durch Rahmen symbolisiert

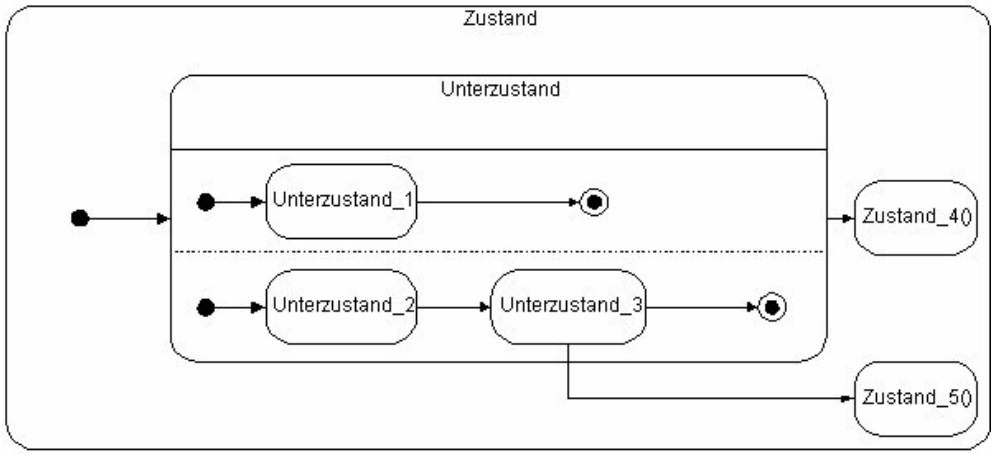


State-Chart-Diagramm

- Zustandsmaschine
- ein Anfangszustand
- endliche Menge Zustände

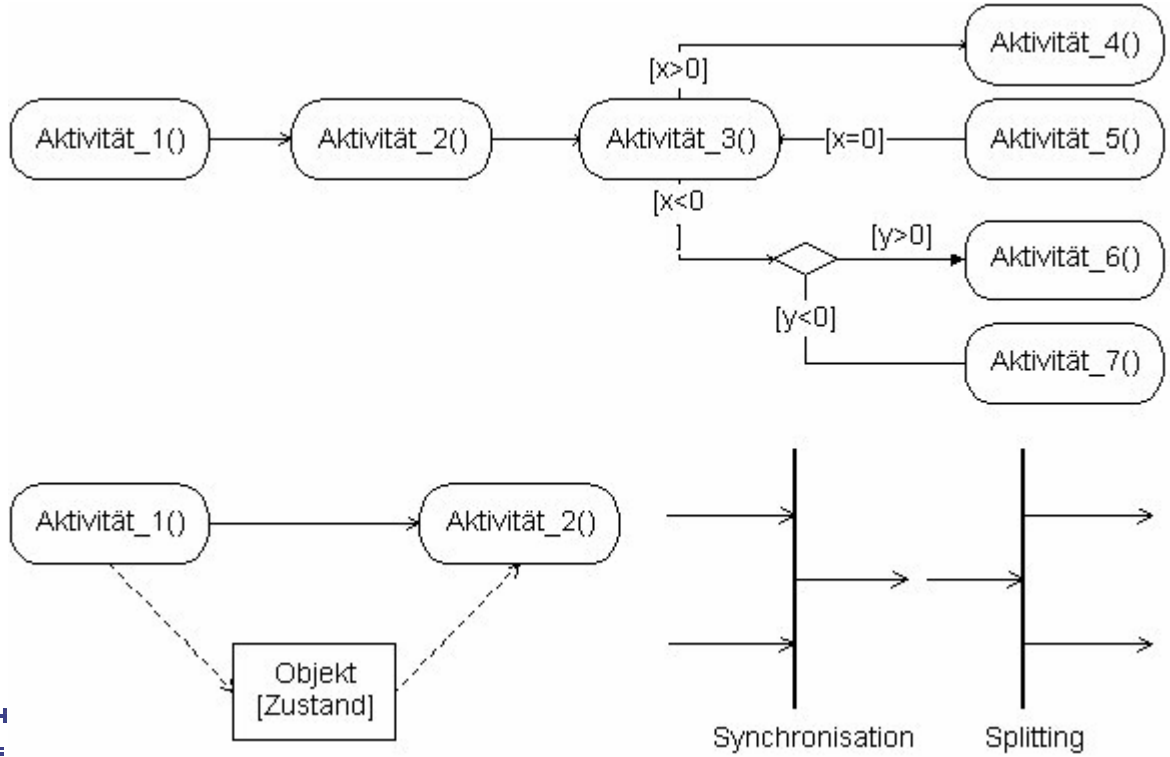
  Unterzustände mit Anfangszustand möglich
- endliche Menge Transitionen durch Ereignisse

  ereignis(argumente) [bedingung] / operation(argumente) ^zielobjekte.gesendetesEreignis(argumente)
- endliche Menge Endzustände



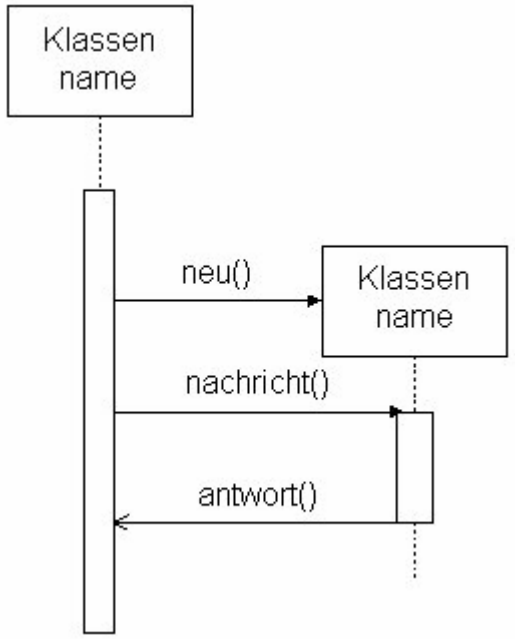
Aktivitätsdiagramm

- Sonderform des State-Chart-Diagramms
- ähnelt prozeduralem Flussdiagramm
- beschreibt Aktivitäten von **Objekten**
- Aktivität ist Zustand mit interner Aktion



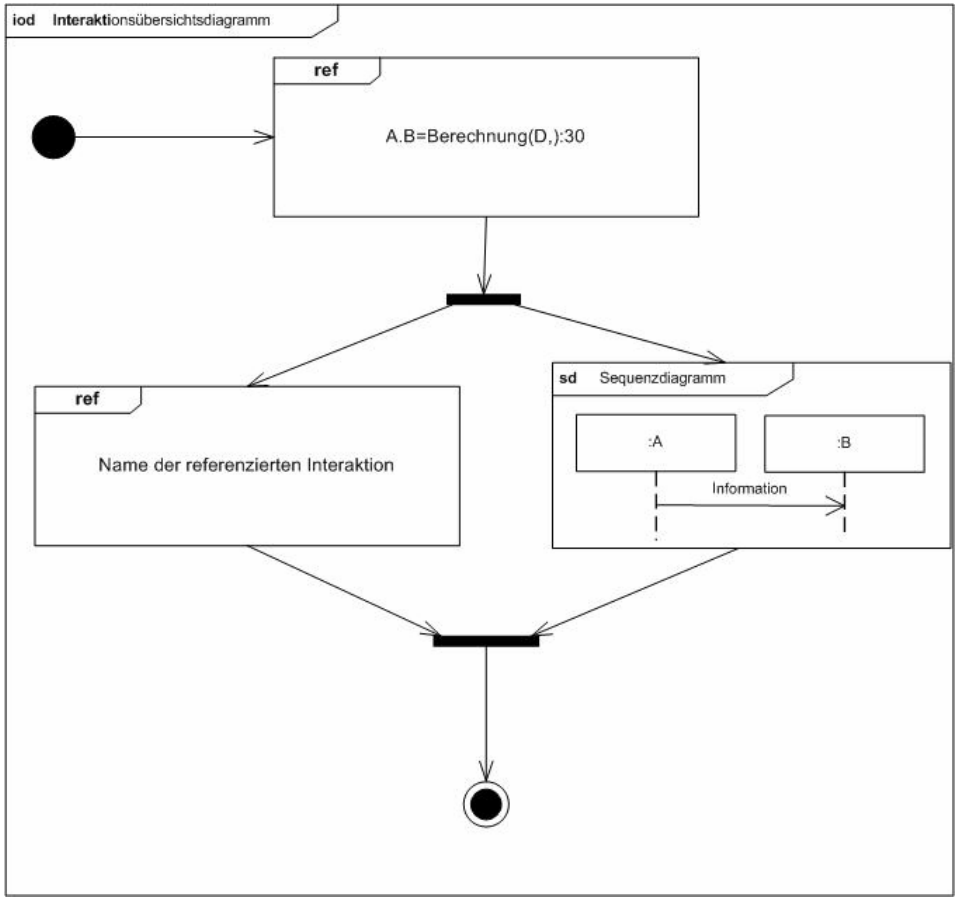
Sequenzdiagramm

- beschreibt zeitliche Abfolge von Interaktionen zwischen **Objekten**
- Zeitlinie senkrecht von oben nach unten
- Objekte durch senkrechte Lebenslinien beschrieben
- gesendete Nachrichten waagerecht entsprechend zeitlichen Auftretens



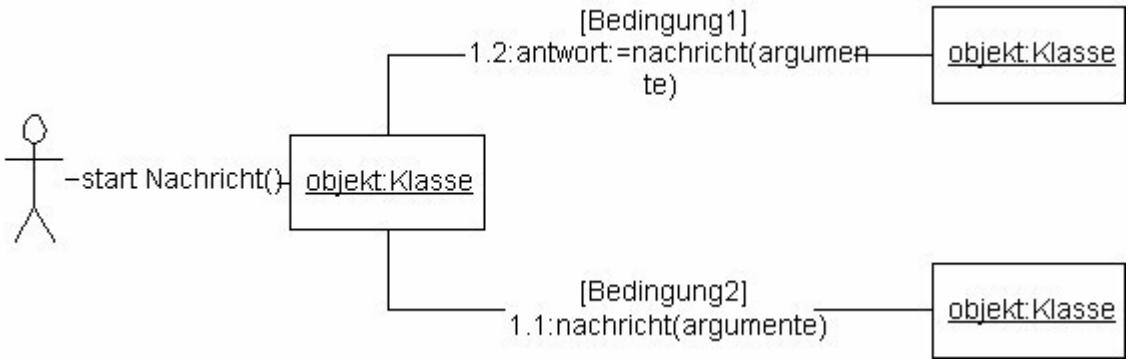
Interaktionsübersichtsdiagramm UML2

- Vermischung aus Aktivitäts- und Sequenzdiagramm
- Aktivitätsbeschreibung in Form von Sequenzdiagrammen



Kommunikationsdiagramm

- *Kollaborationsdiagramm* in UML 1
- Interaktionen für begrenzten Kontext, unter besonderer Beachtung der Beziehungen der Objekte und ihrer Topographie
  - Objekte durch Assoziationslinien verbunden
  - Nachrichten (zeitliche Nummerierung, Namen, Antwort und Argumenten) entlang der Assoziationslinien
  - Startnachricht erhält noch keine Nummer
- Aussagekraft wie Sequenzdiagramm



Timing-Diagramm

UML2

- dienen Darstellung von Zeitbeschränkungen für Zustandswechsel von ein oder mehreren Objekten