Technology
Arts Sciences
TH Köln

ST1

Softwaretechnik 1 (ST1) im SoSe 2022
Objektorientierte Modellierung und Entwicklung

Kapitel 4b: UML – Verhaltensmodellierung: Zustandsdiagramm

Lernziel: Nach dieser Vorlesung sollten Sie ...

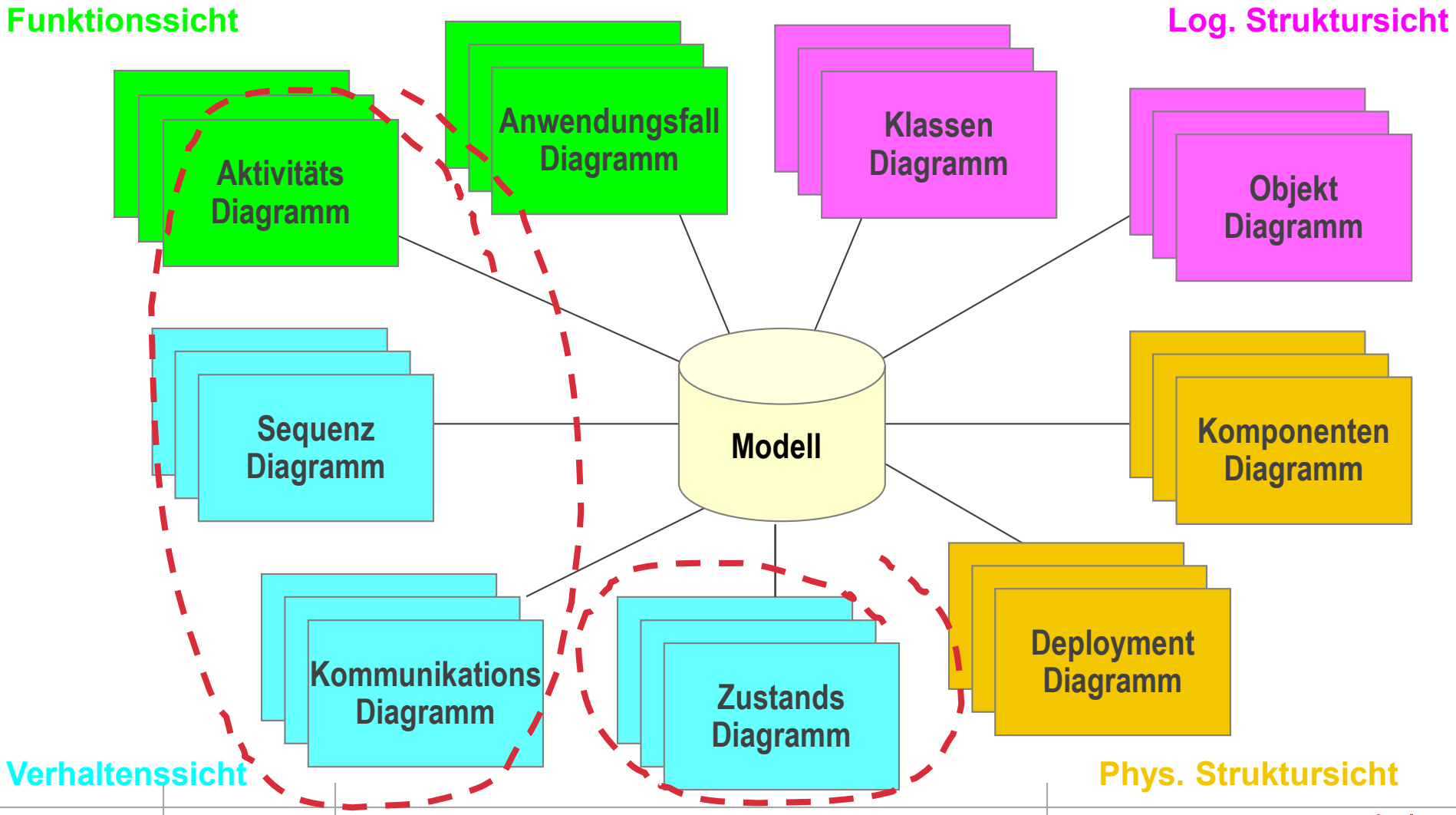
- den Unterschied zwischen interaktivem (ablauforientiertem) Verhalten und reaktivem (zustandsorientiertem) Verhalten verstehen sowie
- Zustände von Objekten als Abstraktion konkreter Attributwerte und Verbindungen und
- Ereignisse als Auslöser von Aktionen und/oder Zustandsänderungen (Transitionen) modellieren können,

um das reaktive, zustandsabhängige Verhalten eines Systems oder aller Instanzen einer Klasse mit einem Zustandsdiagramm zu modellieren.

Inhaltsüberblick

- Interaktives vs. reaktives Verhalten
- Zustände und Ereignisse
- Zustandsübergänge
- Zustandsdiagramm

Verhaltensorientierte Diagramme in der UML



Rückblick: Arten von Verhalten in der Objektorientierung

Interaktives Verhalten (Ablauforientiertes Verhalten)

- Inter-Objektverhalten („System-Intern“ beobachtbares Verhalten)
  - Abläufe von Operationen
  - Objekte i.d.R. mehrerer Klassen arbeiten bei der Ausführung einer im Klassenmodell definierten Operation durch den Austausch von Nachrichten zusammen
  - Determiniert durch den Zustand der Objekte zu Beginn der Operations-ausführung und die aktuellen Parameter des Operationsaufrufs
- Systemverhalten („System-Extern“ beobachtbares Verhalten)
  - Abläufe von Anwendungsfällen
  - Akteure interagieren in einem Szenario eines Anwendungsfalls mit dem Anwendungssystem
  - Determiniert durch den Zustand des Anwendungssystems zu Beginn des Anwendungsfalls und die aktuellen Interaktionsparameter

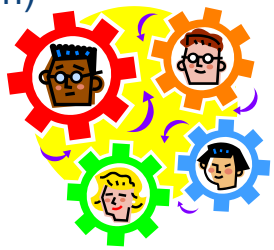
Vgl. auch Aktivitätsmodellierung (Kap. 2)

Reaktives Verhalten (Zustandsorientiertes Verhalten)

- Mögliche/erlaubte Reaktionen eines Systems (oder einer Komponente oder der Instanzen einer Klasse) über den gesamten „Lebenszyklus“ hinweg
- Ereignisse / Reaktionen / Zustandsänderungen

Interaktives Verhalten vs. Reaktives Verhalten

- Interaktionsdiagramme betrachten „von außen“, wie Instanzen (i.d.R. mehrerer Klassen) sich durch den Austausch von Nachrichten koordinieren
- **Interaktionsdiagramm = „*Ein Ablauf für einige Objekte i.d.R. mehrerer Klassen*“**



- Oft möchte man modellieren, wie alle Instanzen einer Klasse abhängig von ihrem Zustand auf Ereignisse reagieren – sich also quasi „in ein Objekt hineinversetzen“
- Zustandsdiagramme stellen das komplexe Zusammenspiel von Zuständen (Attributwerten und Verbindungen) und Operationen bei der Reaktion einer Instanz (einer Klasse) oder eines Systems auf äußere Ereignisse dar

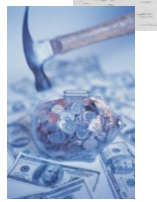
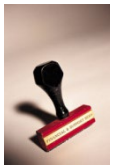
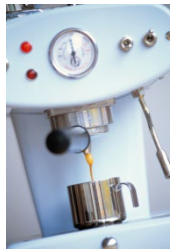
- **Zustandsdiagramm = „*Alle Abläufe für die Instanzen einer Klasse*“**
- Beispiel: Wie reagieren die Instanzen einer Klasse in Abhängigkeit von den Werten ihrer Attribute und ggf. weiteren Bedingungen auf bestimmte Ereignisse wie z. B. den Aufruf einer Operation?
- Eine Reaktion auf das Eintreten eines Ereignisses besteht in der Regel aus der Änderung des Zustandes (Übergang in einen Folgezustand, Änderung von Attributwerten und Verbindungen) und kann die Ausführung weiterer Aktionen und/oder das Auslösen weiterer Ereignisse beinhalten.

Wo sind wir?

- Interaktives vs. reaktives Verhalten
- **Zustände und Ereignisse**
- Zustandsübergänge
- Zustandsdiagramm

Zustände

- Zustände abstrahieren (Teilmengen) möglicher Attributwerte bzw. Attributwertkombinationen …
  - In Abhängigkeit vom Geburtsdatum und dem aktuellen Datum lassen sich bei einer Person die Zustände “Kleinkind“, “Kind“, “Jugendlicher“ und “Erwachsener“ unterscheiden
  - Der Zustand einer Bank kann entweder “Solvent“ oder “Insolvent“ sein, je nachdem, ob ihr Vermögen ihre Verpflichtungen überschreitet oder nicht
  - Der Zustand des Druckbehälters einer Espresso-Maschine kann sein „ Unterdruck“ ([0..8[), „ Normaldruck“ ([8-12[) und „ Überdruck“([12-15[)
  - Der Zustand einer Bestellung kann sein “Eingegangen“, “Offen“, “Ausgeliefert“ und “Bezahlt“
- … und mögliche (Kombinationen) konkreter Objektverbindungen
  - In Abhängigkeit von der Existenz einer (beliebigen) Arbeitgeber-Verbindung ist eine Person in einem der beiden Zustände “Beschäftigt“ und “Arbeitslos“
  - Bei einer Zugmaschine mit Auflieger unterscheidet man in Abhängigkeit von der Existenz einer (beliebigen) Verbindung zwischen der Zugmaschine und einem Auflieger die beiden Zustände “Beladen“ und “Entladen“

Zustandsmodellierung

- In welchen Zuständen kann ein Objekt sein? Was kann bzw. darf es dann alles tun?
- Abstraktionsgrad der Zustände wird vom Abstraktionsgrad der Systemmodellierung bestimmt
- Diejenigen Eigenschaften außer acht lassen, die das Verhalten nicht beeinflussen
  - Beim Telefonieren ist für den Zustand “Wählend“ die aktuell gewählte Nummer irrelevant
  - Für einen anderen Sachverhalt wie z. B. die Operation “anrufen“ ist die Nummer jedoch wichtig und muss als Parameter berücksichtigt werden
- Zustände werden in der UML als abgerundete Rechtecke dargestellt, in denen der Bezeichner des Zustands angegeben wird
  - Existieren in einem Zustandsdiagramm mehrere Zustandssymbole ohne Bezeichner, repräsentieren diese (per Definition) unterschiedliche Zustände
  - Zwei Zustandssymbole mit demselben Bezeichner repräsentieren (per Definition) auch denselben Zustand
  - Aus Gründen der Übersichtlichkeit und einfacheren Konsistenzerhaltung auf die mehrfache Darstellung von Zuständen in einem Diagramm verzichten

Ereignisse

- Objekte ändern ihre Attributwerte durch die Ausführung von Operationen
- Bisher Nachrichten (und insbesondere Operationsaufrufe als ihre spezielle Variante) als Auslöser solcher Zustandsänderungen
- Verallgemeinerung: Empfang einer Nachricht ist ein spezielles Ereignis
- Ereignis: Eintreten eines bestimmten Sachverhalts zu einem bestimmten Zeitpunkt, das im Kontext der Modellierung eine vernachlässigbare Zeitdauer besitzt
- Ereignisse werden mit Ereignisnamen versehen
- Nachrichtsempfangs- (Aufruf-) Ereignis (call event)
  - Empfang einer synchronen oder asynchronen Nachricht, welche die Ausführung einer Operation veranlasst
- Änderungsereignis (change event)
  - Änderung, die für das zustandsorientierte Verhaltens relevant ist, z.B. Instanziieren oder Löschen eines Objekts, Beginn oder Ende eines Vorgangs, Wahrwerden einer bestimmten Bedingung
- Zeitereignis (time event)
  - Ablauf einer bestimmten Zeitperiode oder Eintritt eines bestimmten Zeitpunktes

Ereignismodellierung: Abstraktion und Attribute

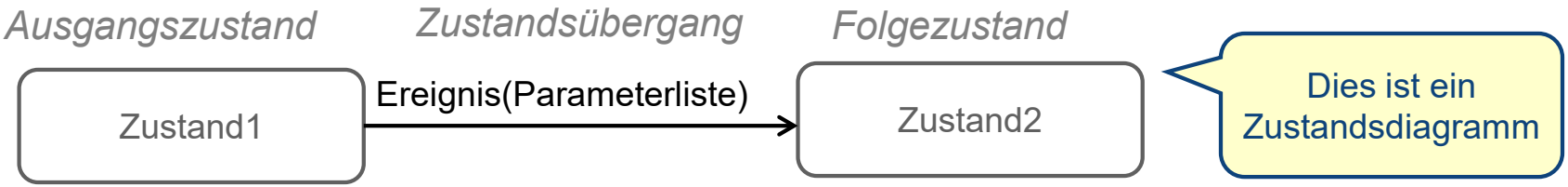
- Abstraktionsgrad hängt vom Abstraktionsgrad ab, mit welchem das betrachtete System modelliert wird
  - Ein Reiseveranstalter z. B. behandelt beim Planen einer Reiseroute jeden Eintritt in eine neue Etappe als Ereignis
  - Eine Fluganzeigetafel eines Flughafens unterscheidet die Ereignisse “Ankunft“ und “Abflug“
  - Ein System zur Überwachung des Flugverkehrs unterteilt den Luftraum in Bereiche und betrachtet jedes Überfliegen einer Bereichsgrenze als relevantes Ereignis
- Unterscheiden sich mehrere Ereignisse lediglich in Details kann ein (Aufruf-) Ereignis Ereignisattribute haben, so dass seine “Instanzen“ anhand ihrer konkreten Werte voneinander unterscheidbar sind
  - Ohne Ereignisattribute müssen beispielsweise bei einem Flugüberwachungssystem die Ereignisse „Flug 123 der KLM fliegt in Chicago ab“, „Flug 456 der Lufthansa fliegt in Rom ab“ usw. namentlich unterschieden werden
  - Bei der Verwendung von Ereignisattributen sind diese gleichartigen Ereignisse in der Form „Abflug (Fluglinie, Flugnummer, Stadt)“ angebbar und unterscheidbar
  - Ein konkretes Ereignis (Ereignis-Instanz) wäre dann z. B. „Abflug(KLM, 123, Chicago)“

Wo sind wir?

- Interaktives vs. reaktives Verhalten
- Zustände und Ereignisse
- **Zustandsübergänge**
- Zustandsdiagramm

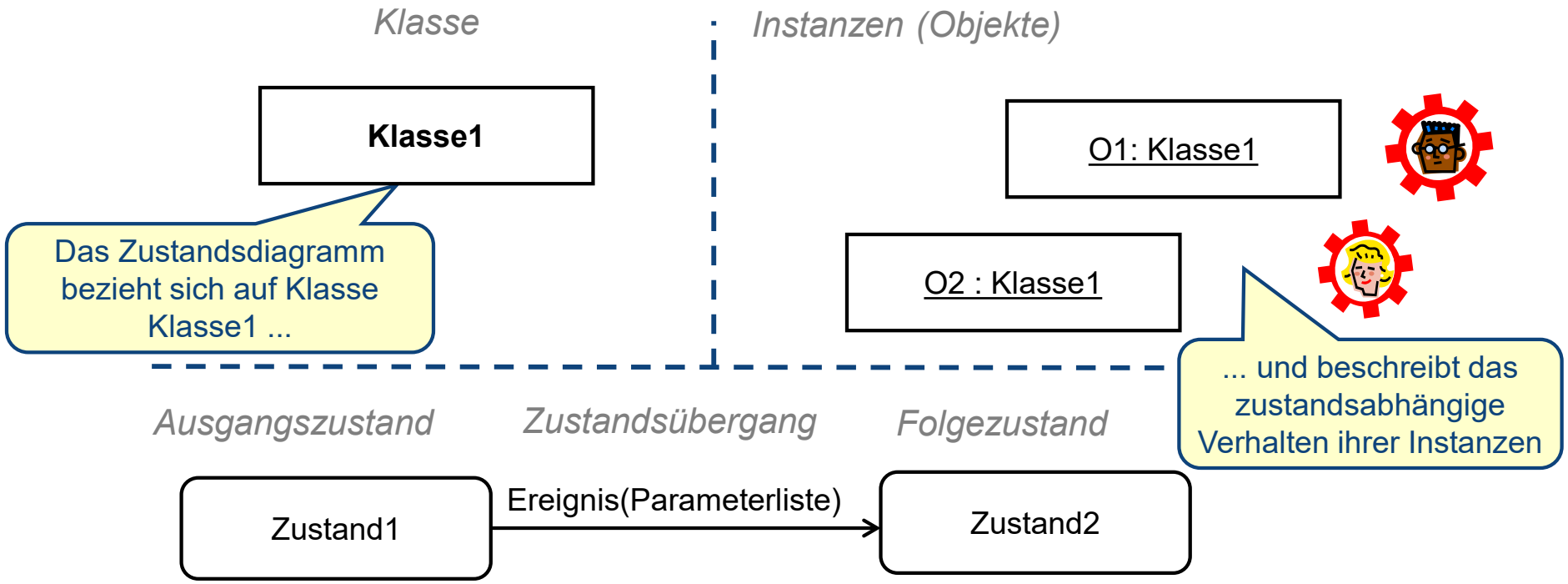
Zustandsübergänge (Transitionen)

- Das Verhalten eines Objekts variiert je nach aktuellem Objektzustand und eingetretenem Ereignis (trigger)
- Entsprechend muss das zustandsabhängige Verhalten für jede Kombination aus Zuständen und Ereignissen spezifiziert werden
- Die Reaktion auf des Eintreten eines Ereignisses wird als Zustandsübergang (transition) modelliert
  - Ein Objekt in einem bestimmten (Ausgangs-)Zustand geht bei Eintritt eines bestimmten Ereignisses in einen bestimmten (Folge-)Zustand über
  - Sind Ausgangs- und Folgezustand eines Zustandsübergangs identisch, spricht man auch von einem reflexiven Zustandsübergang.
- Ein Zustandsübergang wird in der UML als Pfeil mit offener Spitze vom Ausgangs- zum Folgezustand dargestellt, der mit dem Namen des den Zustandsübergang auslösenden Ereignisses beschriftet werden kann



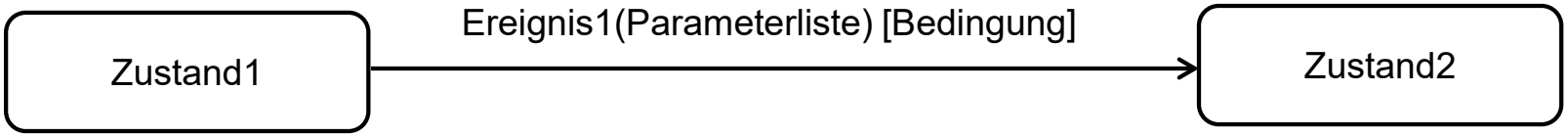
Zustandsübergänge modellieren Objektverhalten

- Zustandsübergänge modellieren, wie die Instanzen der Klasse abhängig von ihrem Zustand auf das Eintreten von Ereignissen reagieren
- Dies nennt man reaktives oder zustandsabhängiges Verhalten



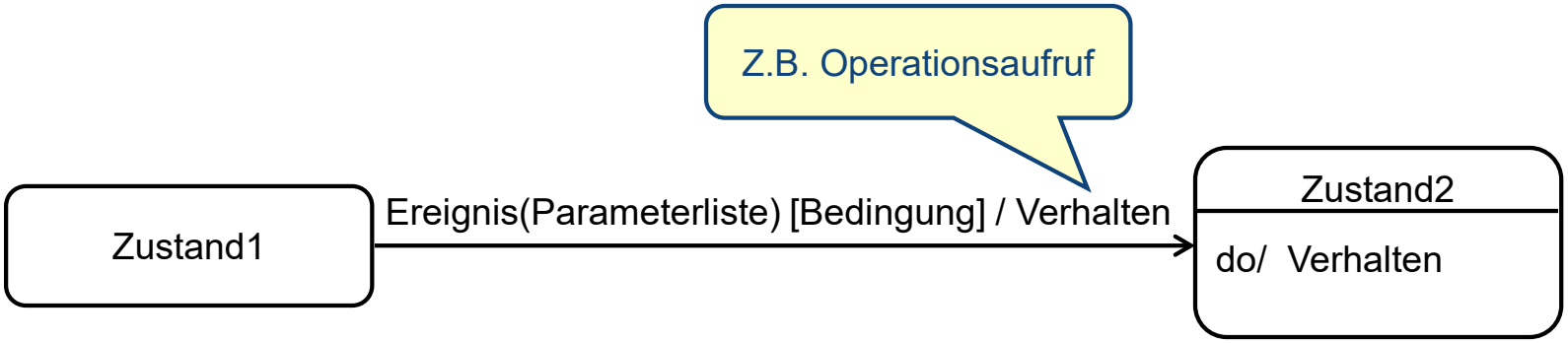
Zustandsübergänge mit Bedingungen

- Ob ein Zustandsübergang ausgelöst werden soll hängt manchmal z. B. von den konkreten Werten der Ereignisattribute ab
- Zur Modellierung solcher Sachverhalte können Zustandsübergänge mit Bedingungen versehen werden
- Eine Bedingung (Wächterbedingung, guard) bezieht sich u. A. auf Parameter des auslösenden Ereignisses sowie auf Attribute und Verbindungen des betroffenen Objekts
- Die Bedingung kann als “Wächter“ des Zustandsübergangs betrachtet werden: Der “bewachte“ Zustandsübergang wird nur dann ausgelöst, wenn das zugehörige Ereignis eintritt und die Wächterbedingung erfüllt ist
- Die Wächterbedingung eines Zustandsübergangs wird im Zustands-diagramm in eckigen Klammern hinter dem Ereignisnamen angegeben



Verhalten bei Zustandsübergängen (1)

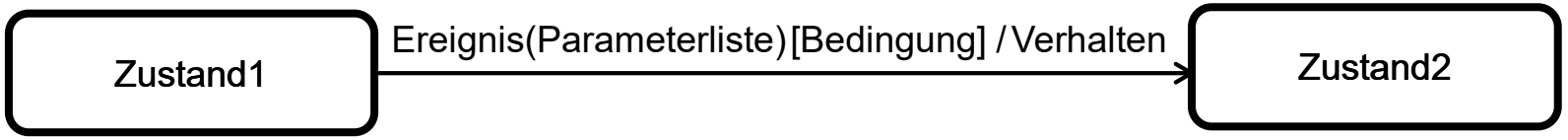
- Objekte, die nur ihren Zustand wechseln können, sind meist uninteressant
- Bei einem Zustandsübergang wird oft ein bestimmtes Verhalten ausgeführt und dabei ggf. weitere Ereignisse ausgelöst
  - Z.B. Ausführung einer bestimmten Operation oder Aktion, bei der ggf. weitere Nachrichten gesendet bzw. Ereignisse ausgelöst werden können
  - Oder Zuweisungen an „lokale“ Variablen und Berechnungen aus Variablen und Ereignisparametern
- Achtung: Die Ausführung des Verhaltens wird im Modell als „zeitlos“ angesehen – keine „Langläufer“ verwenden!
- Für zeitbehaftetes Verhalten: **do-Aktivität** in Zuständen modellieren



Verhalten von Zustandsübergängen (2)

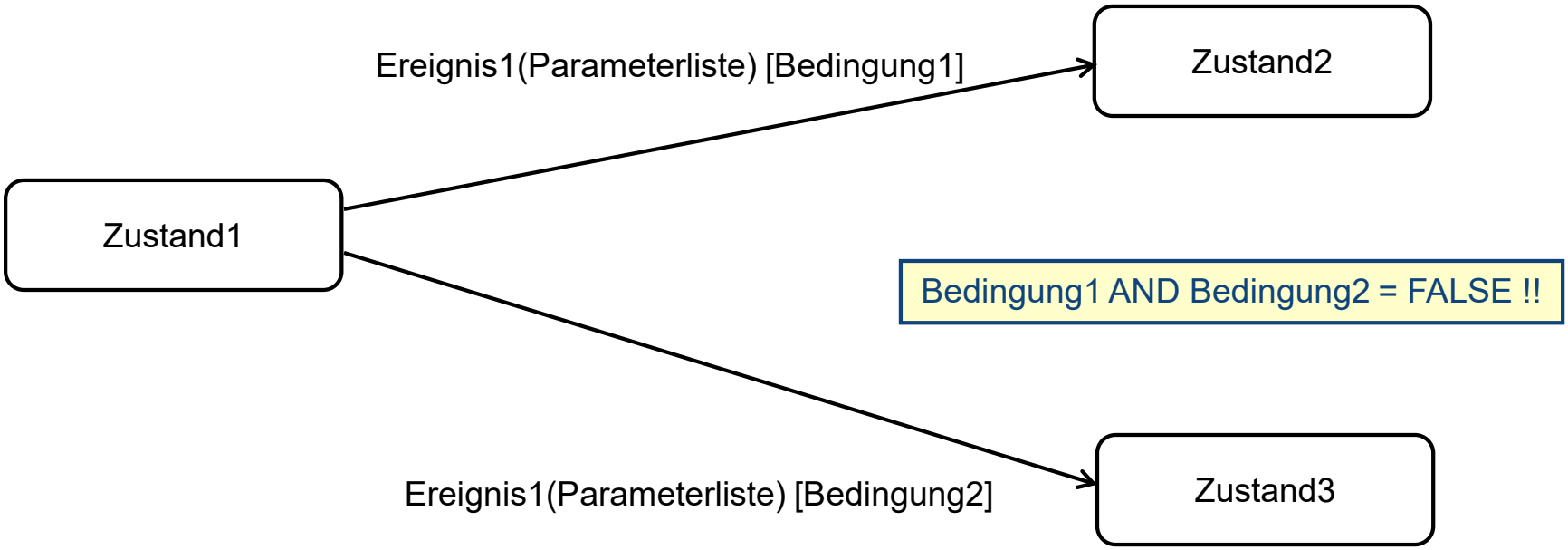
Drei Teile, die alle optional sind:

- Ereignis triggert die Transition, kann Parameter beinhalten. An einer Transition können auch mehrere Ereignisse angegeben werden, durch Komma getrennt
- Wächterbedingung steuert anhand der Ereignisparameter, des Ausgangszustandes sowie weiterer „Attributwerte“, ob die Transition durchgeführt wird („feuert“)
- Verhalten wird ausgeführt, wenn die Transition durchgeführt wird, z.B. Operationsaufruf mit Parameterangaben



Konkurrierende Zustandsübergänge

- Manchmal kann ein und dasselbe Ereignis für denselben Objektzustand unterschiedliche Zustandsübergänge auslösen
- Damit das so modellierte zustandsorientierte Verhalten deterministisch ist, müssen die Bedingungen aller Zustandsübergänge, die durch dasselbe Ereignis ausgelöst werden und von ein- und demselben Zustand ausgehen, sich gegenseitig ausschließen



entry- und exit-Verhalten in Zuständen, interne Transitionen

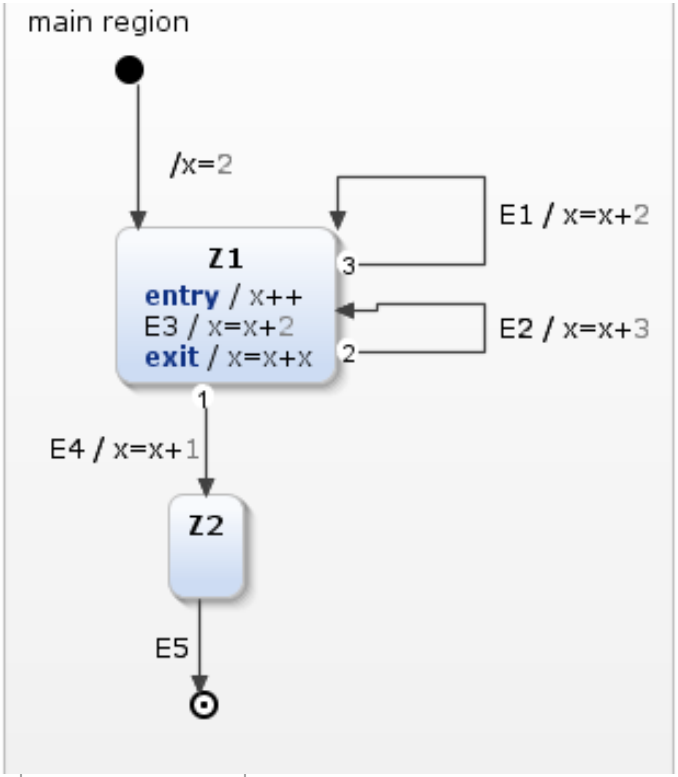
- entry-Verhalten: Wird ausgeführt, nachdem ein eingehender Zustands-übergang getriggert und durchgeführt wurde
- exit-Verhalten: Wird ausgeführt, wenn ein ausgehender Zustandsübergang getriggert und durchgeführt wird (vor der Aktion des ausgehenden Zustandsübergangs)

- Aber: Manchmal soll ein Objekt bei bestimmten Ereignissen seinen Zustand behalten, so dass exit- und entry-Verhalten nicht ausgeführt wird
- Diese Ereignisse als Trigger interner Transitionen modellieren

Beispiel: „Rechnendes“ Zustandsdiagramm

YAKINDU Statechart Tools

- „Papiersimulation“ der Ereignisfolge <create, E1, E3, E4> mit erweiterter Zustands-Übergangstabelle
- „Zustandsdiagramm-lokale“ Variable x wird durch Aktionen gesetzt



Z	E	Z‘	A	x
Initial	create (entry)	 Z1	x=2 x++	2 3
Z1	E1 (exit) (entry)	 Z1	 x=x+x x=x+2 x++	 6 8 9
Z1	E3	 Z1	x=x+2	11
Z1	E4 (exit)	 Z2	 x=x+x x=x+1	 22 23
Z2	destroy	Terminal		

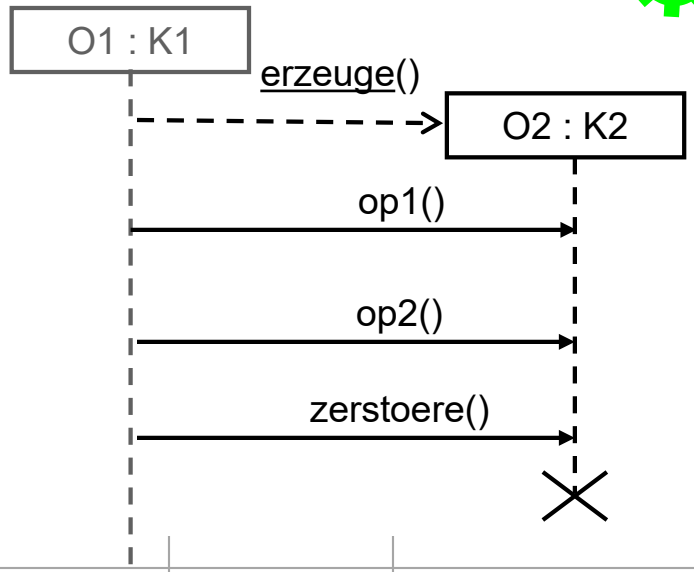
Wo sind wir?

- Interaktives vs. reaktives Verhalten
- Zustände und Ereignisse
- Zustandsübergänge
- **Zustandsdiagramm**

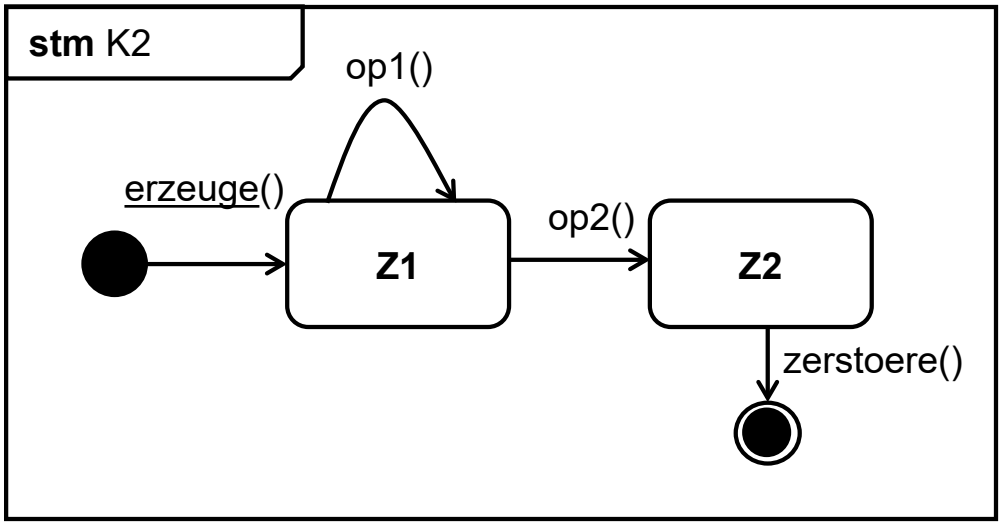
Der Objekt-Lebenszyklus interaktiv und reaktiv modelliert

- Ein Objekt wird irgendwann erzeugt (instanziiert) ...
- ... ändert durch Aufrufe der (in seiner Klasse definierten) Operationen seinen Zustand ...
- ... und wird irgendwann zerstört

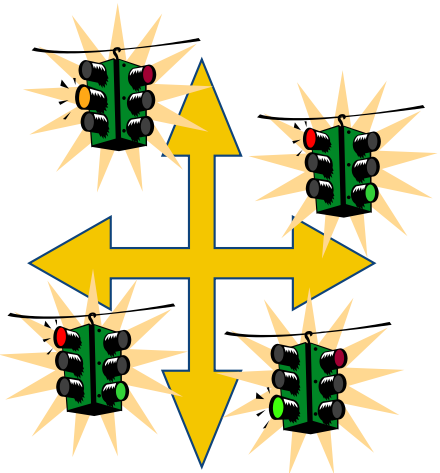
Interaktives Verhalten



Reaktives Verhalten



Zustandsdiagramm einer Ampelsteuerung



Trigger

Wächter-Bedingung

Takt [Auto in N/S-Linksabbiegespur]

Nord/Süd gerade

Nord/Süd links

Takt [Kein Auto in N/S-Linksabbiegespur]

Takt

Takt

Takt [Kein Auto in O/W-Linksabbiegespur]

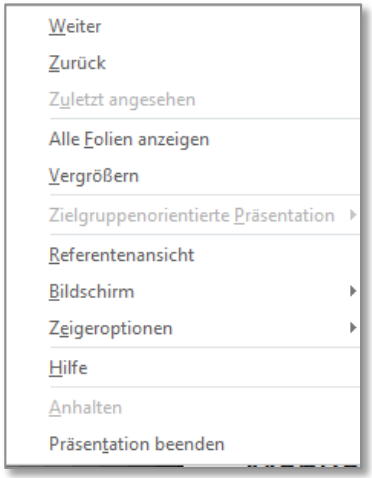
Takt [Auto in O/W-Linksabbiegespur]

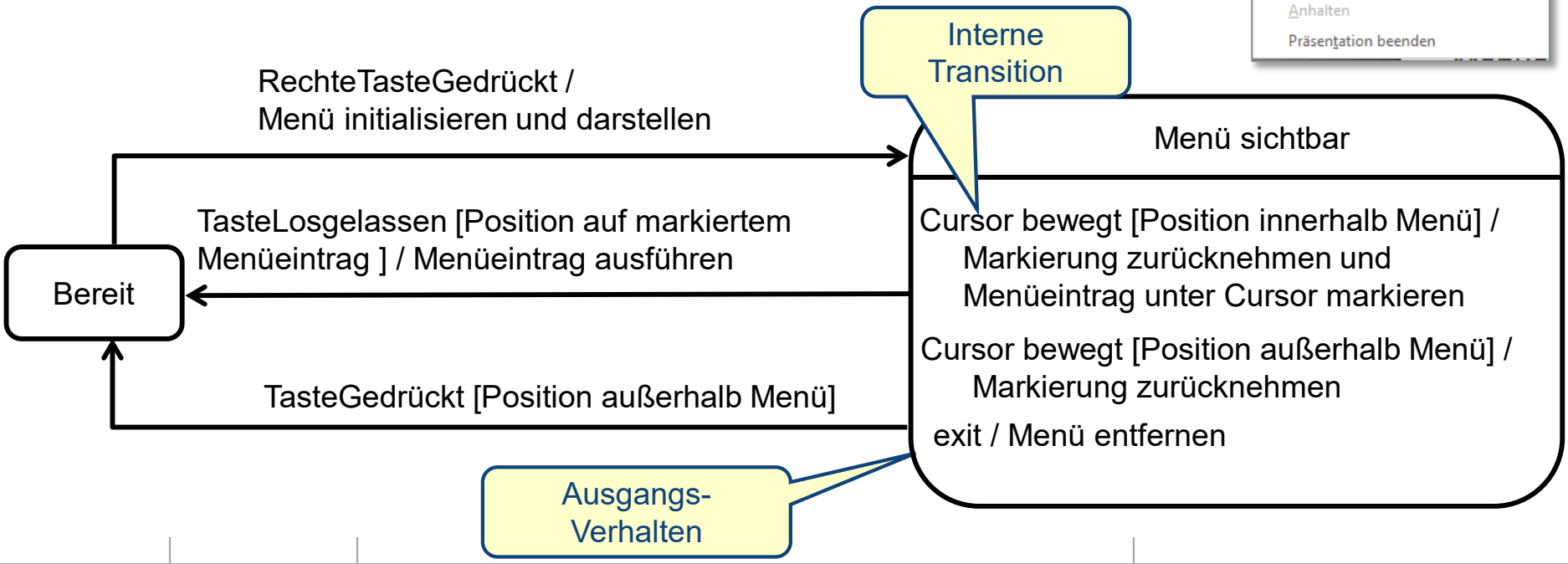
Ost/West links

Ost/West gerade

Zustandsdiagramm eines Pop-Up Menüs

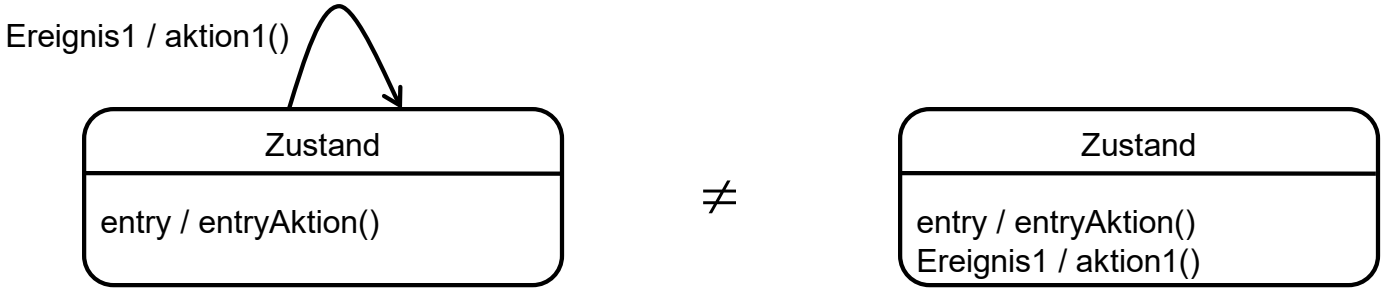
- Interne Transitionen sowie die Eingangs- und Ausgangsaktion werden im unteren, mit einer waagerechten Linie abgetrennten Teil des Zustandssymbols aufgeführt
- Eingangs- und Ausgangsaktionen werden nach einem Schrägstrich hinter den Schlüsselworten **`entry`** bzw. **`exit`** angegeben





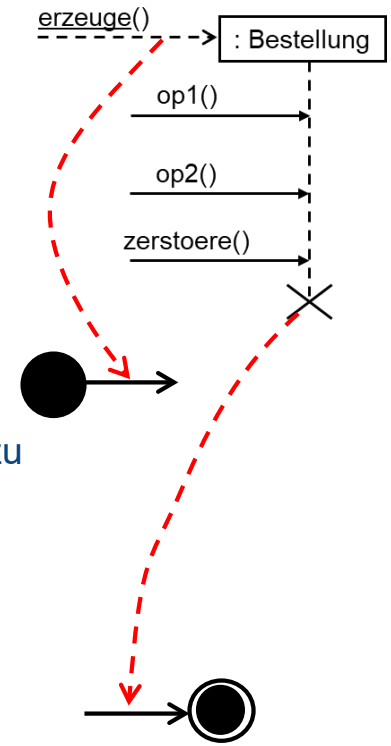
Reflexive Zustandsübergänge und interne Transitionen

- Entry- und Exit-Verhalten führen zu einer unterschiedlichen Semantik für reflexive Zustandsübergänge und interne Transitionen
  - Bei Eintritt des einen reflexiven Zustandsübergang auslösenden Ereignisses wird zunächst das Exit-Verhalten des Zustands ausgeführt, dann das Verhalten des reflexiven Zustandsübergangs und zum Schluss das Entry-Verhalten des Zustands
  - Im Falle einer internen Transition wird dagegen nur das ihr zugeordnete Verhalten, nicht aber das Entry- und Exit-Verhalten des Zustands ausgeführt



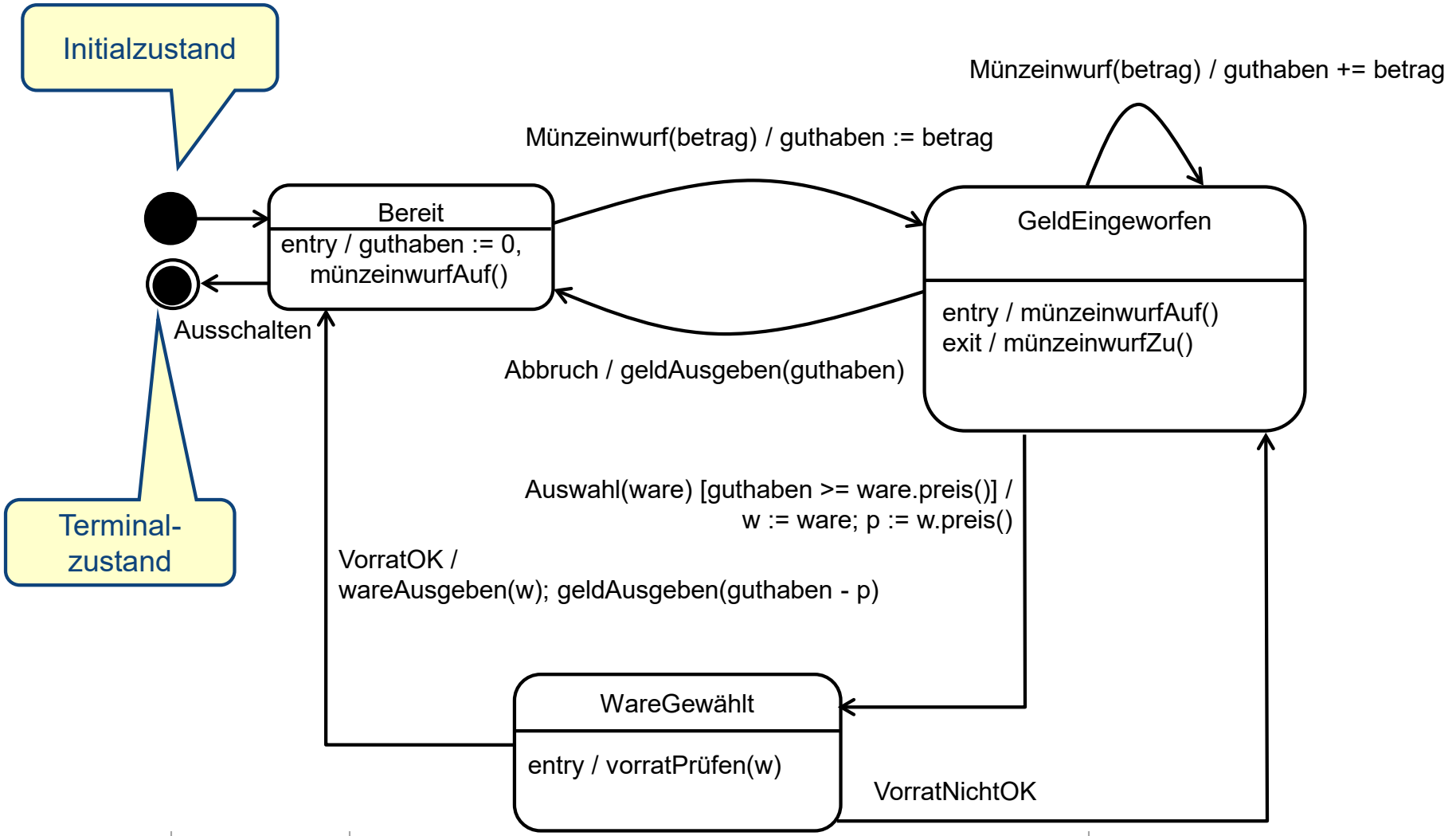
- Zustandsübergange ohne Ereignis und Wächterbedingung werden unmittelbar ausgeführt, sobald ihr Ausgangszustand erreicht wird
- Solche Übergänge sind nur dann sinnvoll, wenn der Ausgangszustand Exit-Verhalten oder der Folgezustand Entry-Verhalten besitzt

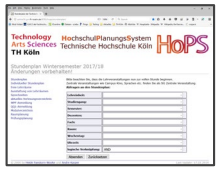
Initialzustand und Terminalzustände



- Zur Erinnerung: Der Lebenszyklus eines Objekts beginnt mit seiner Erzeugung und endet mit seiner Zerstörung
- Im Zustandsdiagramm kann dies mit genau einem Initialzustand und einer Menge von Terminalzuständen modelliert werden
  - Initialzustand: schwarz gefüllter Kreis
  - Terminalzustand: schwarz gefüllter Doppelkreis („bulls eye“)
  - Ggfs. mehrere Terminalzustände verwenden, um ein besseres Layout zu erhalten
- Ist ein Initialzustand angegeben, so muss von ihm aus genau ein Zustandsübergang ohne Ereignis und Wächterbedingung zum Ausgangszustand des Objekts führen – dieser Zustandsübergang erzeugt das Objekt
- Initialzustand und Terminalzustand sind sogenannte Pseudozustände, da kein Objekt in einem solchen Zustand beobachtet werden kann:
  - Übergang aus Initialzustand entspricht Erzeugung eines Objekts
  - Übergang in einen Terminalzustand entspricht Zerstörung des Objekts

Zustandsdiagramm eines Getränkeautomaten

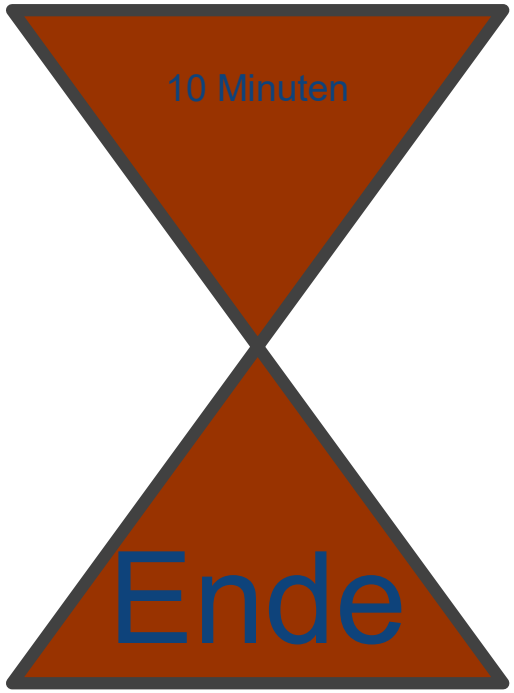


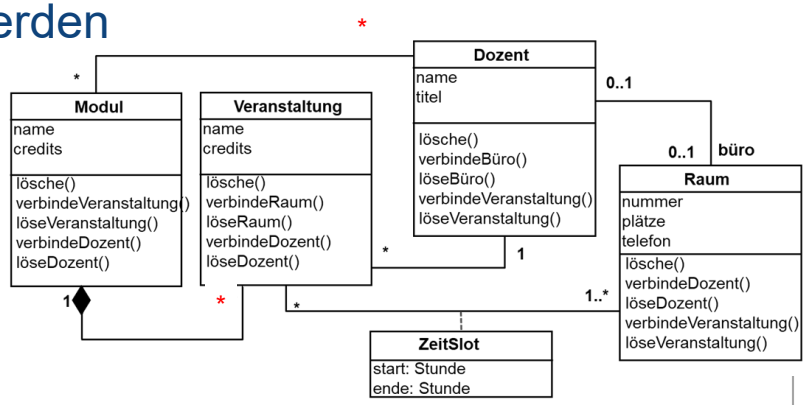


Max. 10 Minuten!

Aufgabe 1: Zustandsmodellierung - Zustandsdiagramm

- Erstellen Sie ein Zustandsdiagramm für die Instanzen der Klasse Modul! Beachten Sie dabei:
  - Dozenten können einem Modul jederzeit zugeordnet werden
  - Veranstaltungen dürfen nur dann zu einem Modulen zugeordnet werden, wenn dem Modul mindestens ein Dozent zugeordnet ist
  - Wenn eine Veranstaltung dem Modul zugeordnet ist, dürfen nicht alle Dozenten gelöscht werden





Lösungsidee Aufgabe 1: Zustandsdiagramm

Zusammenfassung

- Reaktives Verhalten modelliert mit Zuständen und Transitionen, wie die Instanzen einer Klasse auf Ereigniseintritte reagieren
  - Zustände abstrahieren (Mengen) konkrete(r) Attributwerte und Objekt-verbindungen
  - Ereignisse beschreiben „Geschehen“ in der Raum-Zeit, können Parameter haben und Zustandsübergänge auslösen (triggern)
  - Zustandsübergänge werden durch Ereignisse getriggert, ggfs. durch „Guards“ bewacht, und können weiteres Verhalten auslösen (Aktionen)
- Zustandsdiagramm = Alle Abläufe für alle Instanzen einer Klasse
  - Reaktionen der Instanzen auf Ereignisse (Nachrichten/Operationsaufrufe/...)
  - Objekt-Lebenszyklus