



Vienna University of Technology

Objektorientierte Modellierung

Aktivitätsdiagramm



Business Informatics Group

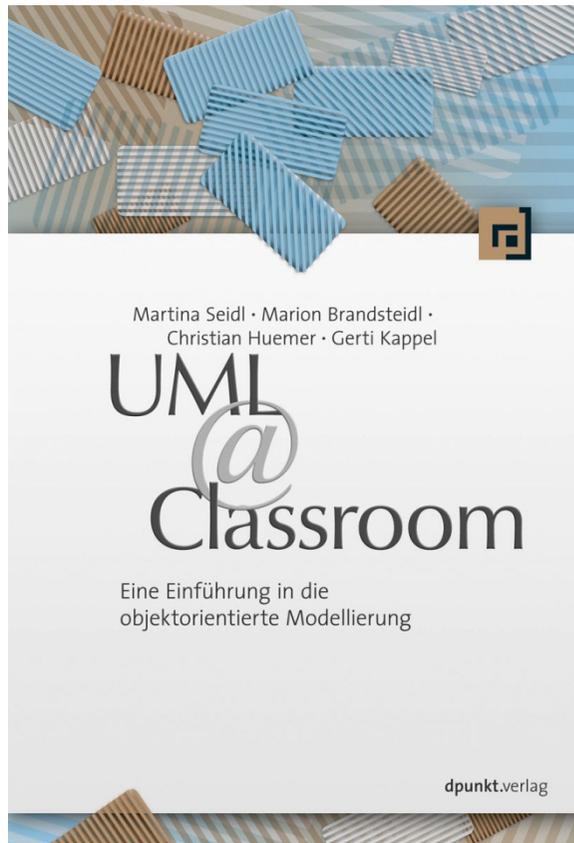
*Institute of Software Technology and Interactive Systems
Vienna University of Technology*

Favoritenstraße 9-11/188-3, 1040 Vienna, Austria

*phone: +43 (1) 58801-18804 (secretary), fax: +43 (1) 58801-18896
office @big.tuwien.ac.at, www.big.tuwien.ac.at*

Literatur

- Die Vorlesung basiert auf folgendem Buch:



UML @ Classroom:

Eine Einführung in die objekt-orientierte Modellierung

*Martina Seidl, Marion Brandsteidl,
Christian Huemer und Gerti Kappel*

dpunkt.verlag

Juli 2012

ISBN 3898647765

- *Anwendungsfalldiagramm*
- *Strukturmodellierung*
- *Zustandsdiagramm*
- *Sequenzdiagramm*
- ***Aktivitätsdiagramm***

Inhalt

- Einführung
- Aktivitäten
- Aktionen
- Kanten
- Initialknoten, Aktivitätstendknoten, Ablaufendknoten
- Token
- Alternative Abläufe
- Parallele Abläufe
- Objektknoten und Objektfluss
- Signale und Ereignisse
- Ausnahmebehandlung und Unterbrechungsbereich

Einführung

- Fokus des Aktivitätsdiagramms: **prozedurale Verarbeitungsaspekte**
- Spezifikation von **Kontroll-** und/oder **Datenfluss** zwischen Arbeitsschritten (Aktionen) zur Realisierung einer Aktivität
- **Aktivitätsdiagramm in UML2:**
 - ablauforientierte Sprachkonzepte
 - basierend u.a. auf Petri-Netzen und BPEL4WS
- Sprachkonzepte und Notationsvarianten decken ein **breites Anwendungsgebiet** ab
 - Modellierung objektorientierter und nichtobjektorientierter Systeme wird gleichermaßen unterstützt
 - Neben vorgeschlagener grafischer Notation sind auch beliebige andere Notationen (z.B. Pseudocode) erlaubt

Aktivität

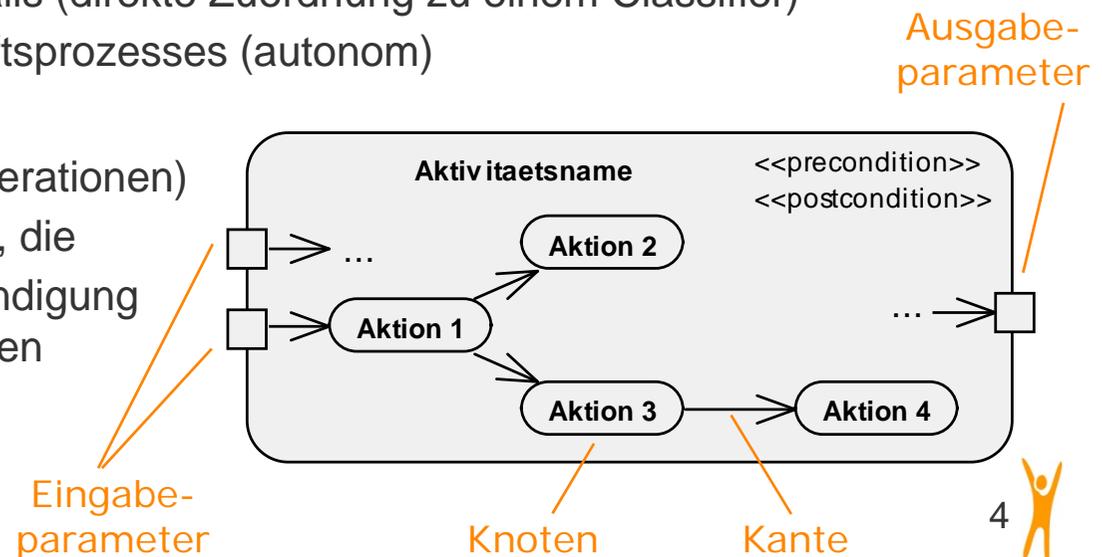
- Eine Aktivität ist ein **gerichteter Graph**
 - Knoten: Aktionen
 - Kanten: Kontroll- und Datenflüsse
- Kontroll- und Datenflüsse legen potentielle »Abläufe« fest
- Spezifikation von **benutzerdefiniertem Verhalten** auf unterschiedlichen Granularitätsebenen

Beispiele:

- Definition einer Operation in Form von einzelnen Anweisungen (indirekte Zuordnung zu einem Classifier)
- Ablauf eines Anwendungsfalls (direkte Zuordnung zu einem Classifier)
- Spezifikation eines Geschäftsprozesses (autonom)

- optional:

- **Parameter** (z.B. wie bei Operationen)
- Vor- und Nachbedingungen, die
 - bei Beginn bzw. bei Beendigung der Aktivität gelten müssen



Aktionen

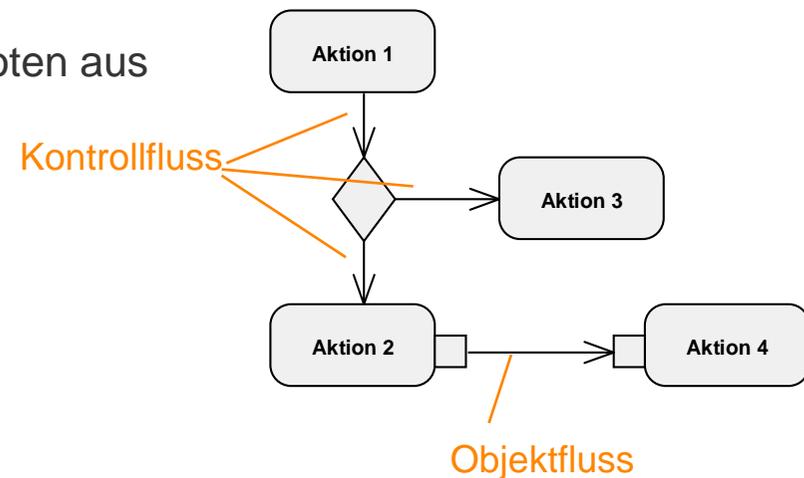
- **Elementare Bausteine** für beliebiges benutzerdefiniertes Verhalten
- **Atomar**, können aber abgebrochen werden
- **Sprachunabhängig**, allerdings Definition in beliebiger Programmiersprache möglich
- Aktionen können Eingabewerte zu Ausgabewerten verarbeiten
- Spezielle **Notation** für bestimmte Aktionsarten
- **Kategorisierung** der 44 in UML vordefinierten Aktionen:
 - Kommunikationsbezogene Aktionen (z.B. Signale und Ereignisse)
 - Objektbezogene Aktionen (z.B. Erzeugen und Löschen von Objekten)
 - Strukturmerkmals- und variablenbezogene Aktionen (z.B. Setzen und Löschen einzelner Werte von Variablen)
 - Linkbezogene Aktionen (z.B. Erzeugen und Löschen von Links zwischen Objekten sowie Navigation)



Aktion A

Kanten

- Kanten verbinden Knoten und legen **mögliche Abläufe** einer Aktivität fest
 - Kontrollflusskanten
 - Drücken eine **reine Kontrollabhängigkeit** zwischen Vorgänger- und Nachfolgerknoten aus
 - Objektflusskanten
 - Transportieren zusätzlich Daten und drücken dadurch auch eine **Datenabhängigkeit** zwischen Vorgänger- und Nachfolgerknoten aus
- Überwachungsbedingung (guard)
 - Bestimmt, ob Kontroll- und Datenfluss weiterläuft oder nicht

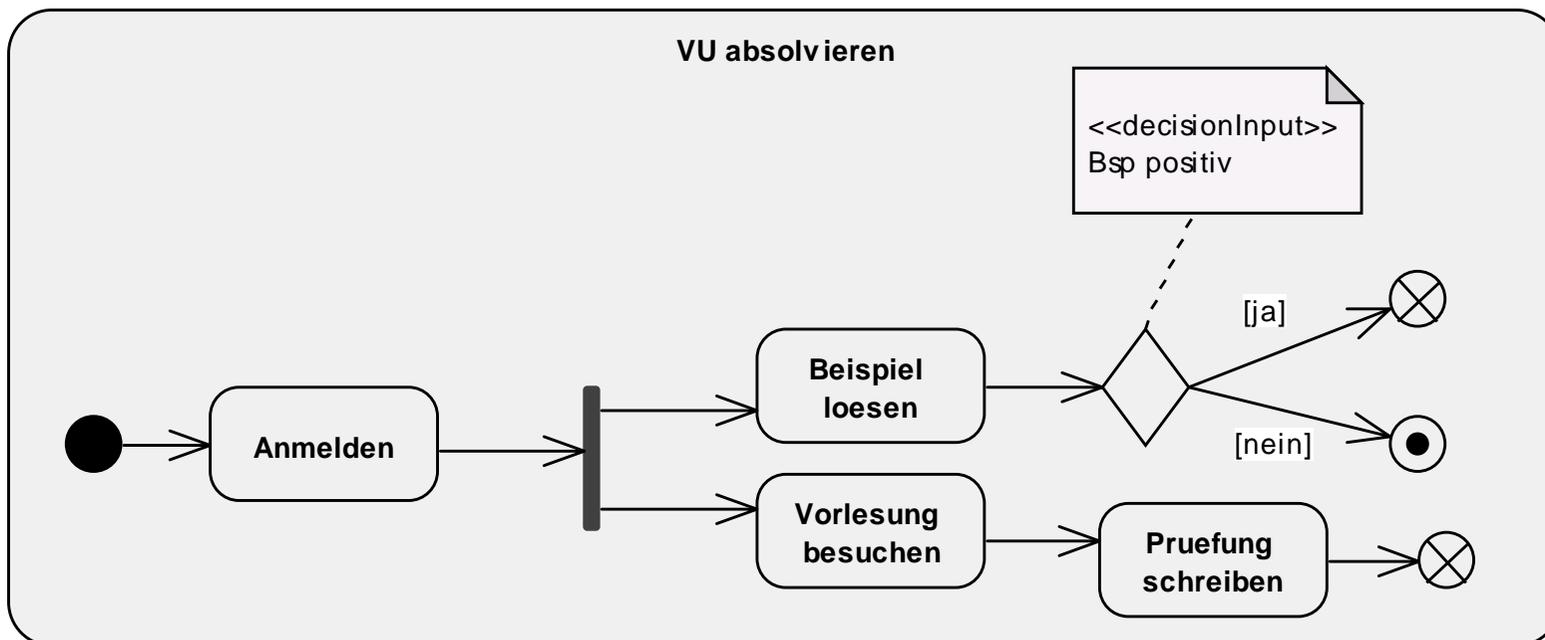


Start und Ende von Aktivitäten und Abläufen

- **Initialknoten** 
 - Beginn eines Aktivitätsablaufs
 - Versorgt alle ausgehenden Kanten mit Kontrolltoken
 - Aufbewahrung von Token erlaubt, da Überwachungsbedingungen die Weitergabe blockieren können
 - Pro Aktivität keine oder mehrere Initialknoten erlaubt – letzteres ermöglicht Nebenläufigkeit
- **Aktivitätseindknoten** 
 - Beendet alle Abläufe einer Aktivität sowie den Lebenszyklus eines Objekts
 - Der erste Token, der zu einem Endknoten gelangt, beendet die Aktivität (egal, wie viele Kanten in den Knoten führen)
 - Keine Ausführung weiterer Aktionen
 - Kontrolltoken werden gelöscht, Datentoken an Ausgabepins der Aktivität dagegen nicht
 - Pro Aktivität mehrere Aktivitätseindknoten erlaubt
- **Ablaufendknoten** 
 - Beendet einen Ablauf einer Aktivität

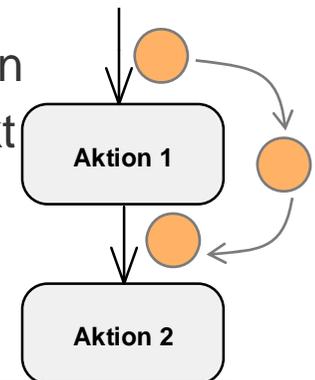
Bsp.: Absolvieren einer VU (für Student)

- Die Aktivität "VU absolvieren" besteht aus den Aktionen „anmelden“, "Beispiele lösen" und "Prüfung schreiben".



Token

- »**Virtueller Koordinationsmechanismus**« zur Beschreibung von Aktivitätsabläufen
- Vorgabe für die Implementierung einer Aktivität
- Token beschreibt möglichen Ablauf einer Aktivität zur Laufzeit
- Token fließen entlang der Kanten von Vorgänger- zu Nachfolgerknoten
 - Werden angeboten und aufbewahrt
 - Lösen Verarbeitung aus
- Unterscheidung in Kontroll- und Datentoken
 - **Kontrolltoken**: "Ausführungserlaubnis" für den Nachfolgeknoten
 - **Datentoken**: Transport von Datenwert oder Referenz auf Objekt
- Überwachungsbedingung kann Weitergabe von Token verhindern (Ansammlung mehrerer Token im Vorgängerknoten)



Alternative Abläufe - Entscheidungsknoten

- Definiert alternative Zweige und repräsentiert eine »Weiche« für den Tokenfluss

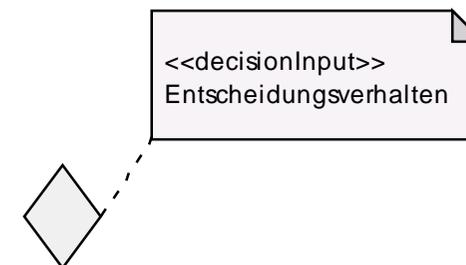
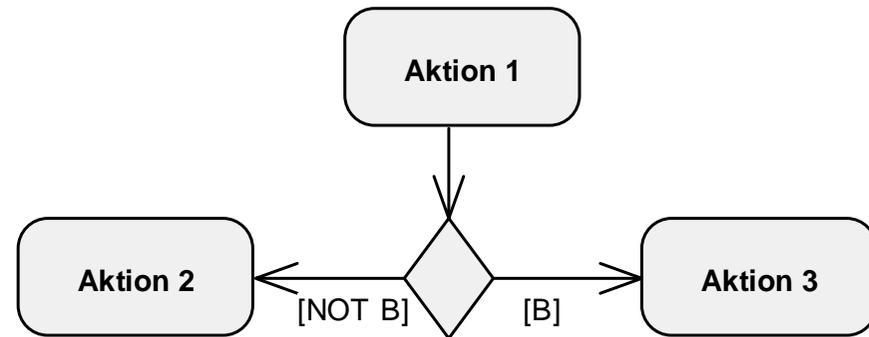
- Verwendung auch zur Modellierung von Schleifen

- Überwachungsbedingungen

- Wählen den Zweig aus
- Müssen wechselseitig ausschließend sein
- [else] ist vordefiniert

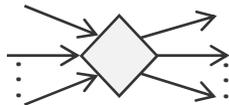
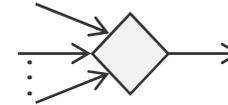
- Entscheidungsverhalten

- Ermöglicht detailliertere Spezifikation der Auswahlentscheidung an zentraler Stelle
- Ankunft von Token startet das Entscheidungsverhalten – Datentoken fungieren als Parameter

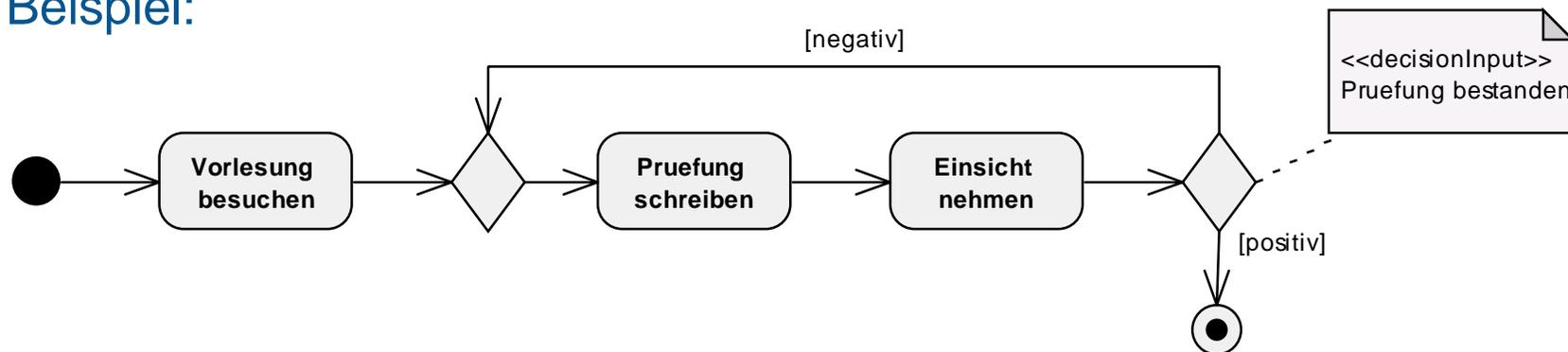


Alternative Abläufe - Vereinigungsknoten

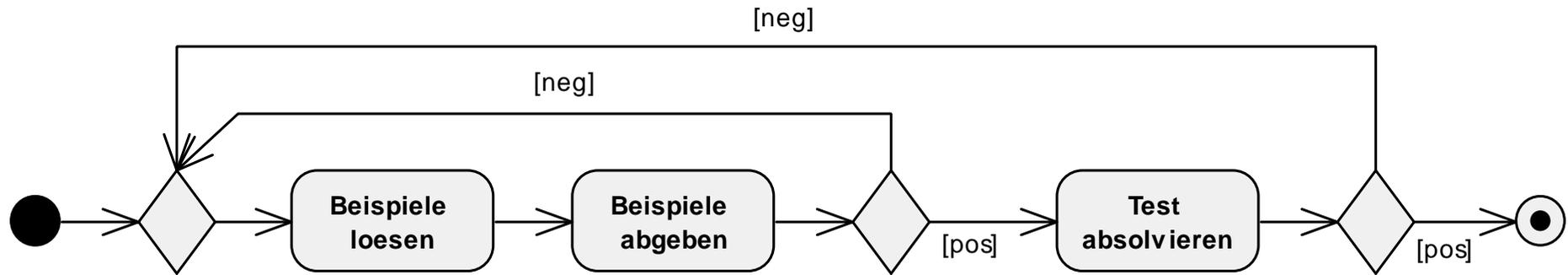
- Ein Vereinigungsknoten führt alternative (keine nebenläufigen!) Abläufe wieder zusammen
- Token werden, sobald möglich, an den Nachfolgerknoten weitergereicht
- Kombiniertes Entscheidungs- und Vereinigungsknoten



- Beispiel:

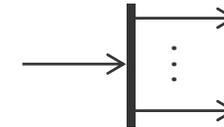


Alternative Abläufe – Bsp.: Absolvieren einer LU



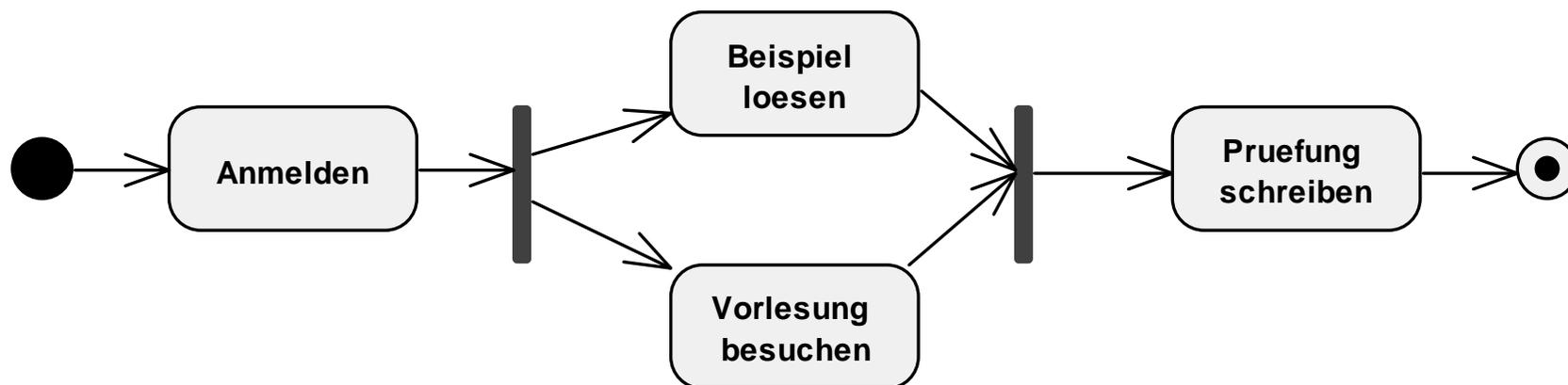
Nebenläufige Abläufe - Parallelisierungsknoten

- Zur Modellierung der Aufspaltung von Abläufen



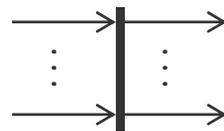
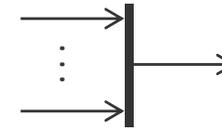
- Eingehende Token werden für alle ausgehenden Kanten dupliziert, sobald zumindest eine Überwachungsbedingung diese akzeptiert
- Nichtakzeptierte Token werden aufbewahrt

- Beispiel:

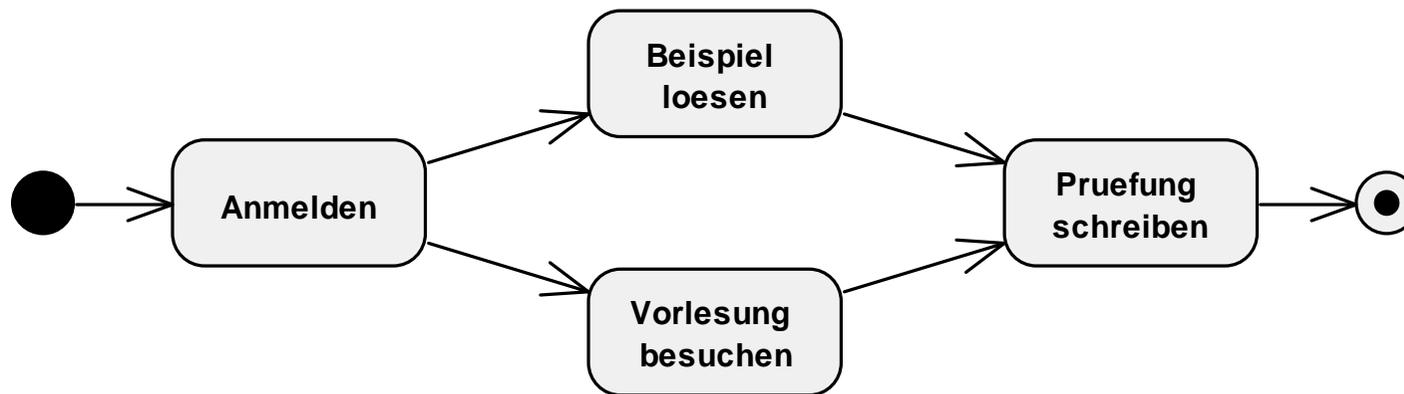


Nebenläufige Abläufe – Synchronisierungsknoten (1/2)

- Führt nebenläufige Abläufe zusammen
- Tokenverarbeitung
 - Vereinigung der Token, sobald an allen Kanten vorhanden
 - An einer Kante anliegende Kontrolltoken werden vereinigt
 - Kontrolltoken verschiedener Kanten werden vereinigt und nur ein einzelnes Token weitergereicht
 - Datentoken werden alle weitergereicht
 - Bei Kontroll- und Datentoken - nur Datentoken werden weitergereicht
 - Nichtakzeptierte Token werden nicht wieder angeboten
- Kombiniertes Parallelisierungs- und Synchronisierungsknoten:

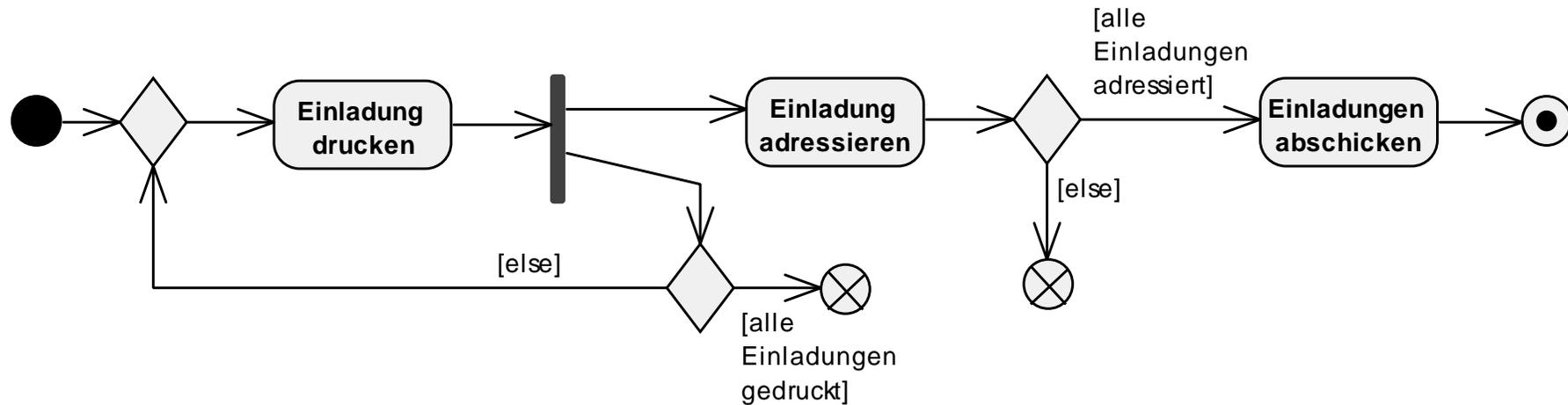


Alternative Darstellung



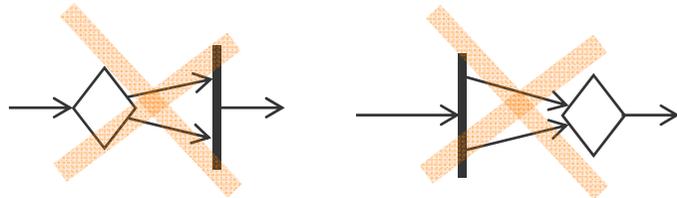
Bsp.: Erstellen und Versenden von Einladungen zu einem Termin im CALENDARIUM

- Einladungen werden einzeln gedruckt und parallel adressiert

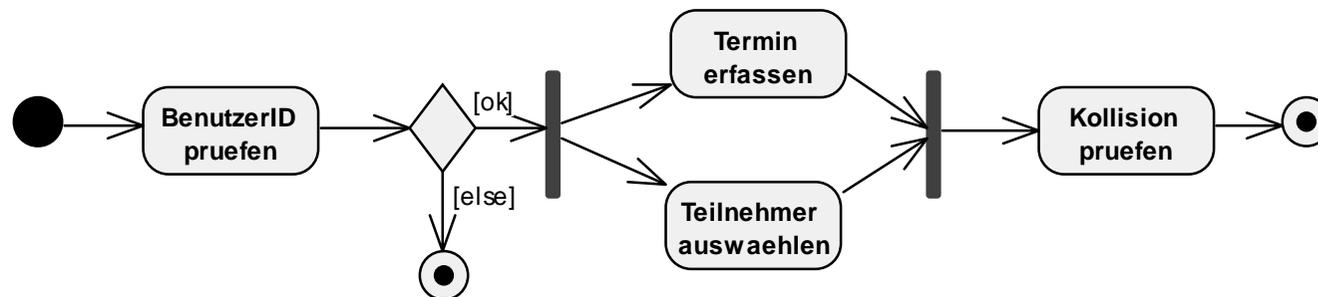


Nebenläufige Abläufe – Synchronisierungsknoten (2/2)

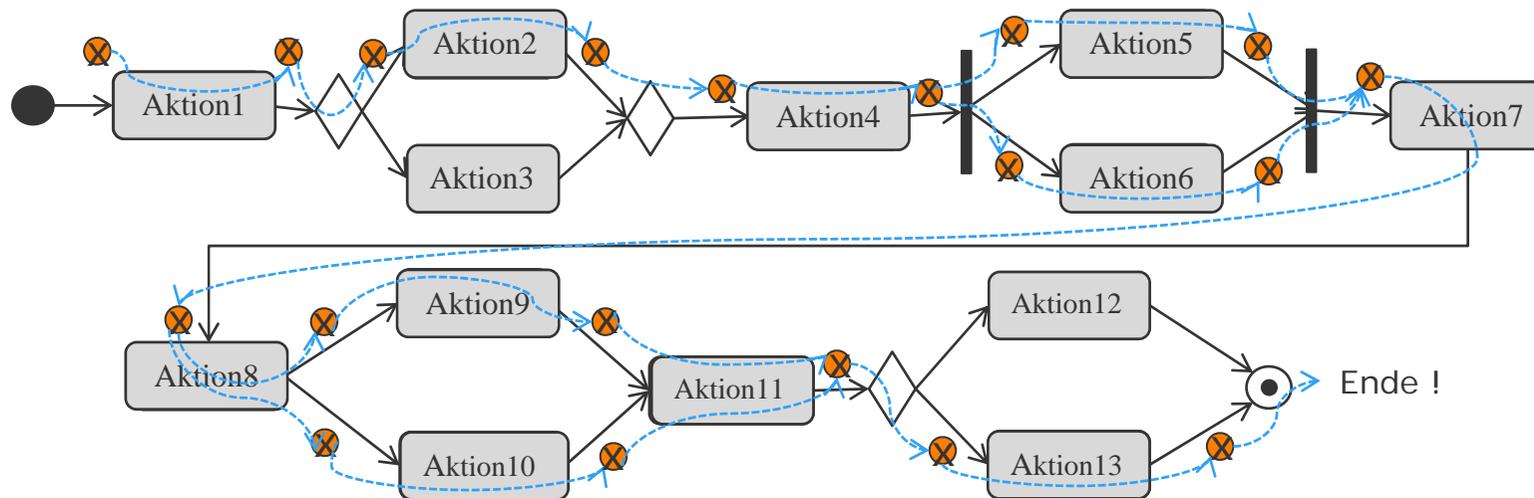
- Falsche Modellierungen



- Beispiel: Einladung im CALENDARIUM
- Wurde die BenutzerID erfolgreich geprüft, ist parallel die Terminerfassung und die Teilnehmerauswahl möglich. Erst wenn diese beiden Aktionen abgeschlossen sind, kann die Kollisionsprüfung durchgeführt werden.



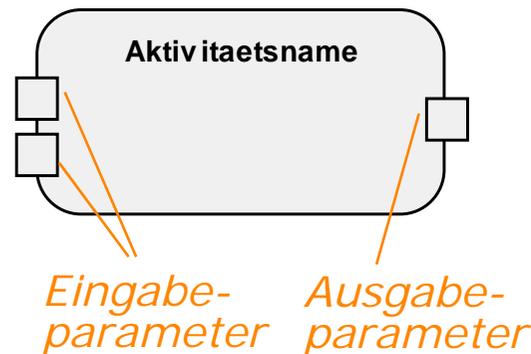
Token – Beispiel (Kontrollfluss)



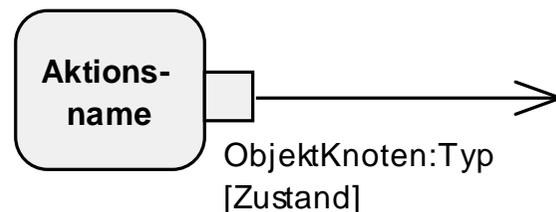
- ... zu Beginn werden alle vom Initialknoten ausgehenden Kanten mit einem Token belegt....
- ... eine Aktion, bei der alle eingehenden Kanten mit einem Token belegt sind, ist aktiviert und kann durchgeführt werden
- ... vor der Durchführung „nimmt“ sich die Aktion von jeder eingehenden Kante einen Token; nach der Durchführung belegt die Aktion jede ausgehende Kante mit einem Token
- ... ein Entscheidungsknoten gibt den Token an **eine** ausgehende Kante weiter
- ... ein Vereinigungsknoten reicht jeden Token, den er bekommt, einzeln weiter
- ... ein Parallelisierungsknoten dupliziert den Token für **jede** ausgehende Kante
- ... ein Synchronisierungsknoten wartet, bis an allen eingehenden Kanten Token anliegen und gibt dann einen Token weiter
- ... der Endknoten ist eine Ausnahme vom Tokenkonzept. Er beendet mit dem ersten Token, den er (egal über welche Kante) bekommt, den gesamten Ablauf

Objektknoten (1/7)

- Inhalt: **Datentoken**
- Objektknoten stehen durch Objektflüsse miteinander in Beziehung
- Inhalt ist **Ergebnis einer Aktion** und Eingabe für eine weitere Aktion
- Typangabe und Zustandseinschränkung sind optional
- Objektknoten als **Ein-/Ausgabeparameter**
 - für Aktivitäten (activity parameter node)

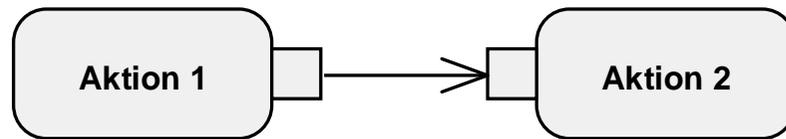


- für Aktionen (pins)

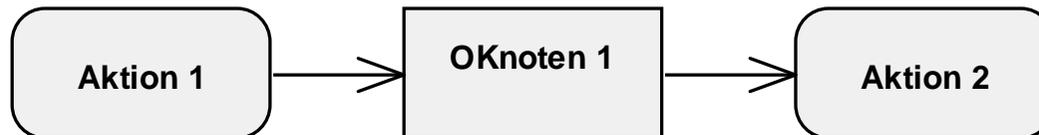


Objektknoten (2/7)

- Kennzeichnung als Ein- und Ausgabepin
 - Notationskonvention: Eingabepins links bzw. oberhalb einer Aktion, Ausgabepins rechts bzw. unterhalb
 - Richtung der Objektflusskante



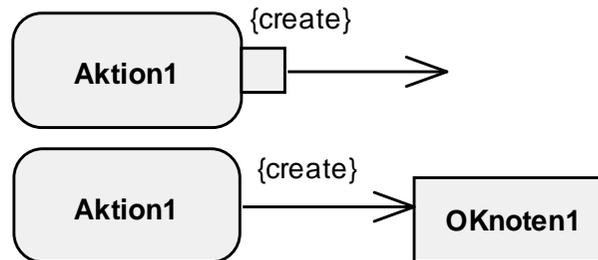
- Weitere Notationsvariante z.B.:



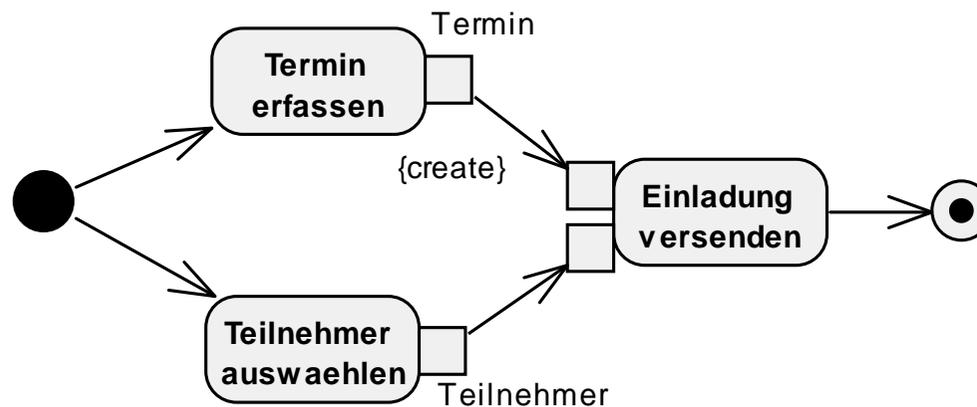
Objektknoten (3/7)

- Effekte einer Aktion auf Daten und Objekte

- create, read, update und delete

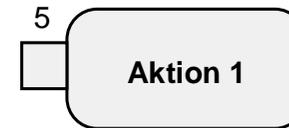


- Beispiel für Ein- und Ausgabepins und Effekte



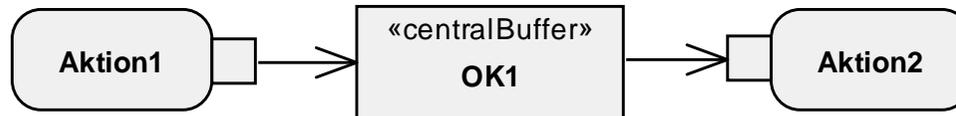
Objektknoten (4/7)

- Konstanter Eingabewert – Wertepin
 - Zur Übergabe konstanter Werte
 - **Startet nicht die Verarbeitung eines Knotens**



- Pufferknoten

- Zentrale Pufferung von Datentoken
- Transienter Pufferknoten (central buffer node)
 - Löscht Datentoken, sobald er sie weitergegeben hat

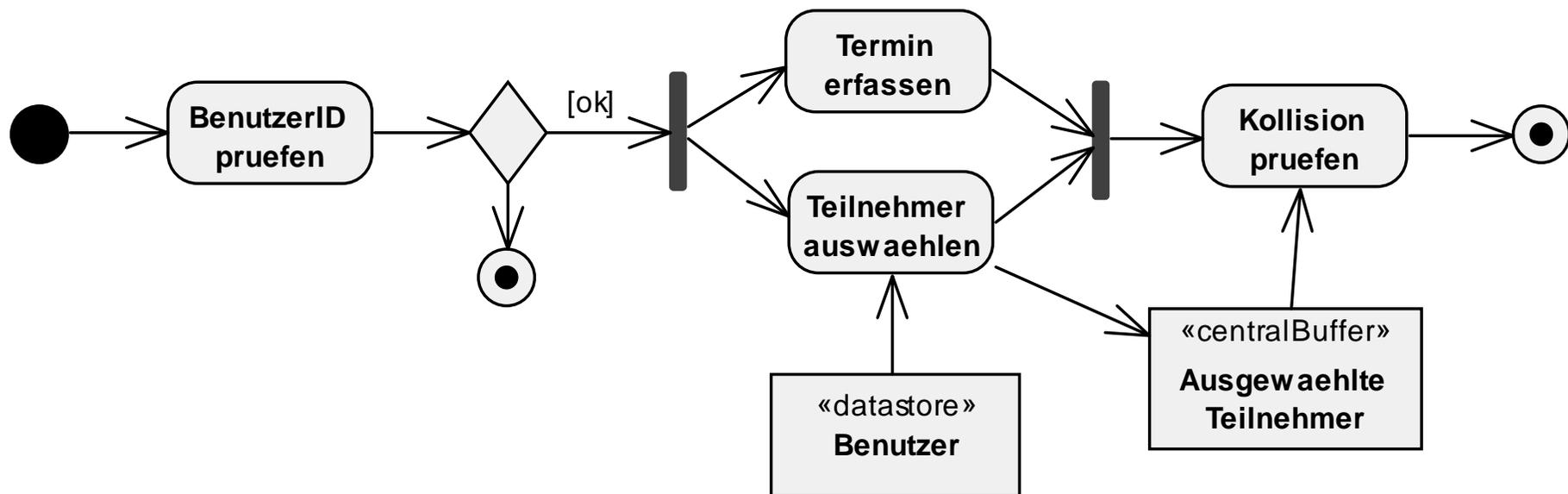


- Persistenter Pufferknoten (data store node)
 - Bewahrt Datentoken auf und gibt Duplikate weiter
 - Keine Mehrfachspeicherung identer Objekte
 - Explizites »Abholen« der Datentoken möglich



Objektknoten (5/7)

- Beispiel: Erweiterung - Einladung im CALENDARIUM
- alle Benutzer werden im Puffer "Benutzer" gespeichert (Datenbank)
- Ausgewählte Benutzer werden in einem transienten Puffer zwischengespeichert und erst für die Versendung von Einladungen wieder entnommen.



Objektknoten (6/7)

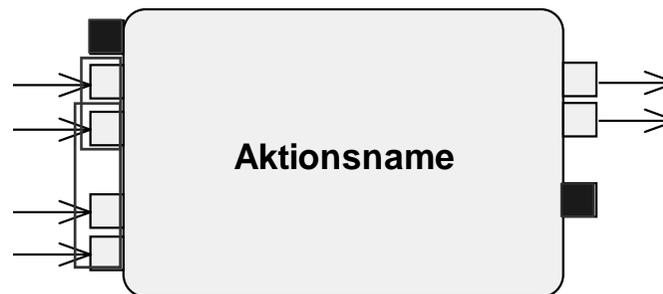
- **Ausgabeparameter für Ausnahmen**

- Werden nur im Falle des Auftretens einer Ausnahme weitergegeben (gewöhnliche Parameter werden in diesem Fall nicht weitergegeben)



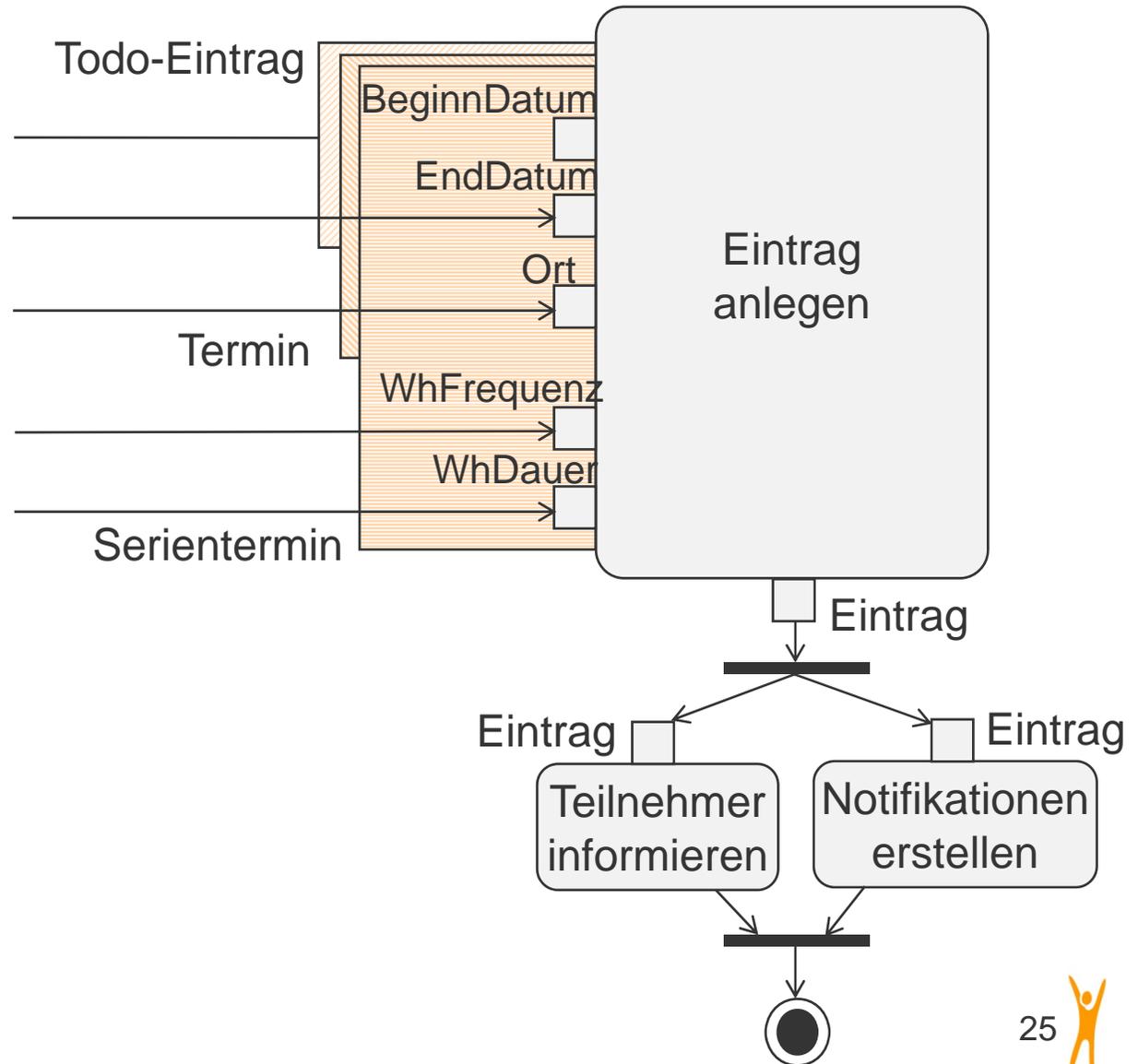
- **Gruppierung von Parametern – Parametersatz**

- Nur ein ausgewählter Parametersatz ist jeweils für die Ausführung der Aktion relevant - Spezifikation von alternativen, einander ausschließenden Gruppen von Ein- bzw. Ausgabewerten



Objektknoten (7/7) – Bsp.: Anlegen eines Kalendereintrags

- 3 Terminarten:
 - Todo-Eintrag
 - + BeginDatum
 - + EndDatum
 - Termin
 - + BeginDatum
 - + EndDatum
 - + Ort
 - Serientermin
 - + BeginDatum
 - + EndDatum
 - + Ort
 - + WhFrequenz
 - + WhDauer



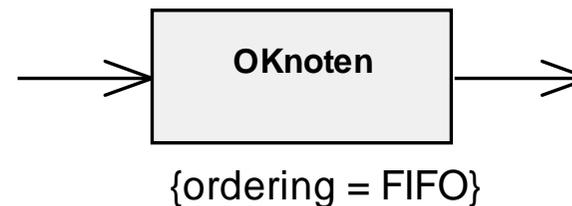
Objektfluss (1/4)

- Hat eine **Transport-** und eine **Kontrollfunktion**
- **Verknüpft** Aktionen nicht direkt, sondern **über Objektknoten**
- Objektknoten bestimmen den Typ der zu transportierenden Objekte

- **Steuerungsmöglichkeiten** der Weitergabe von Datentoken:
 - Reihenfolge
 - Kapazitätsobergrenze und Gewicht
 - Selektionsverhalten
 - Transformationsverhalten

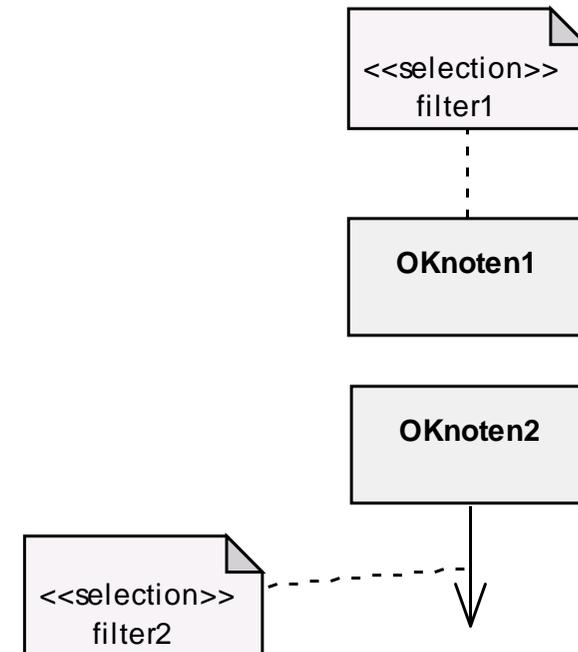
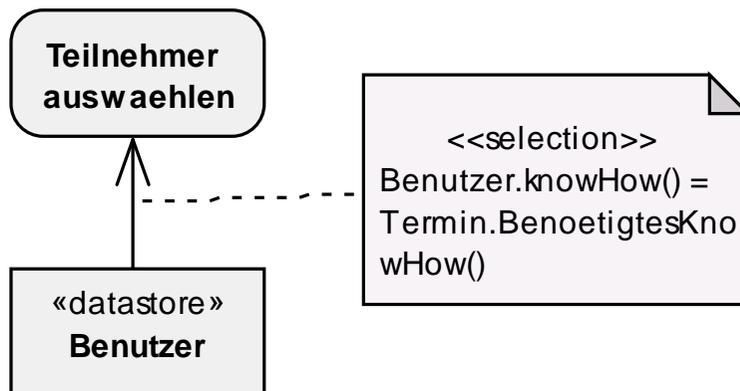
Objektfluss (2/4) – Reihenfolge der Tokenweitergabe

- Explizites Festlegen der Reihenfolge, in der ein Datentoken an eine ausgehende Objektflusskante weitergegeben werden kann
 - **FIFO** (first in, first out) - {ordering = FIFO}
 - Token werden in jener Reihenfolge weitergegeben, in der sie den Objektknoten erreichen (default)
 - **LIFO** (last in, first out) - {ordering = LIFO}
 - Token, die zuletzt eingegangen sind, werden als erste weitergegeben
 - **Geordnet** - {ordering = ordered}
 - benutzerdefinierte Reihenfolge (Angabe von Selektionsverhalten)
 - **Ungeordnet** - {ordering = unordered}
 - Reihenfolge in der die Token eingehen, hat keinen Einfluss auf die Reihenfolge, in der sie weitergereicht werden



Objektfluss (3/4) - Selektionsverhalten

- Wählt bestimmte Token zur Weitergabe aus
- Objektknoten und Objektflusskanten können Selektionsverhalten aufweisen
- Selektionsverhalten – z.B. in Form einer Aktivität – muss einen Eingabe- und einen Ausgabeparameter aufweisen
- Beispiel:



Objektfluss (4/4)

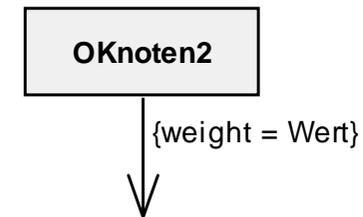
- **Kapazitätsobergrenze eines Objektknotens**

- max. Anzahl von Token, die sich zu einem Zeitpunkt in diesem Knoten befinden dürfen



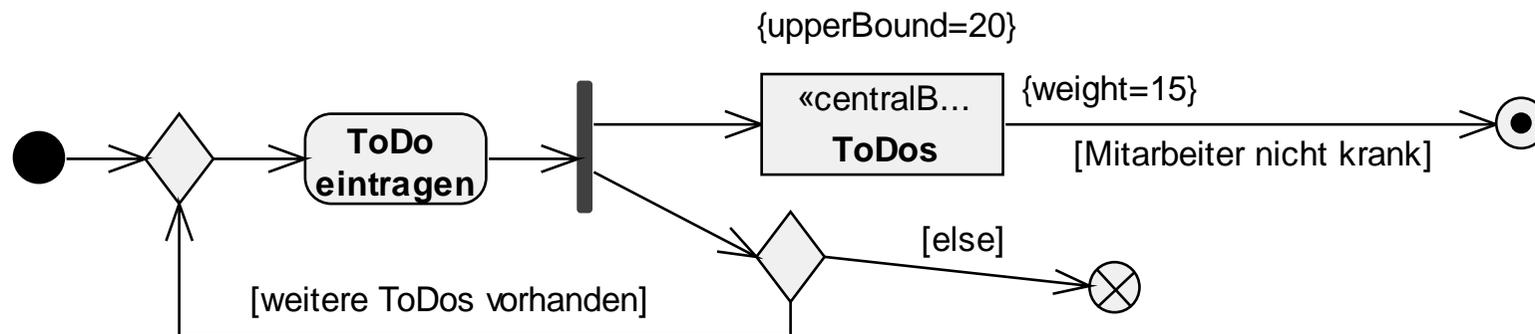
- **Gewicht einer Objektflusskante:**

- Anzahl der Token die anliegen müssen, bevor sie an Nachfolgeknoten weitergegeben werden



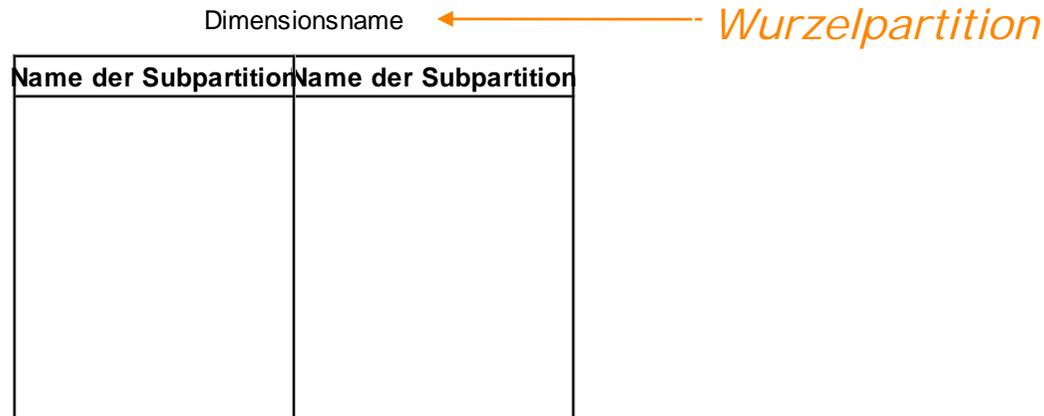
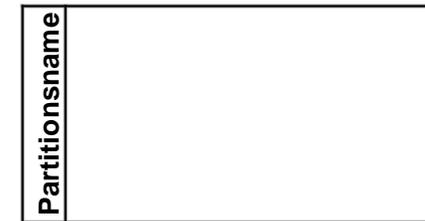
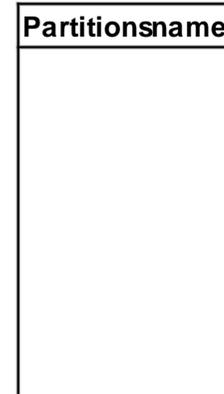
- **Beispiel: Pufferknoten kann max. 20 ToDos aufnehmen.**

- Wenn mind. 15 ToDos vorhanden sind und der Mitarbeiter nicht krank ist, werden 15 davon an den Aktivitätseindknoten weitergereicht

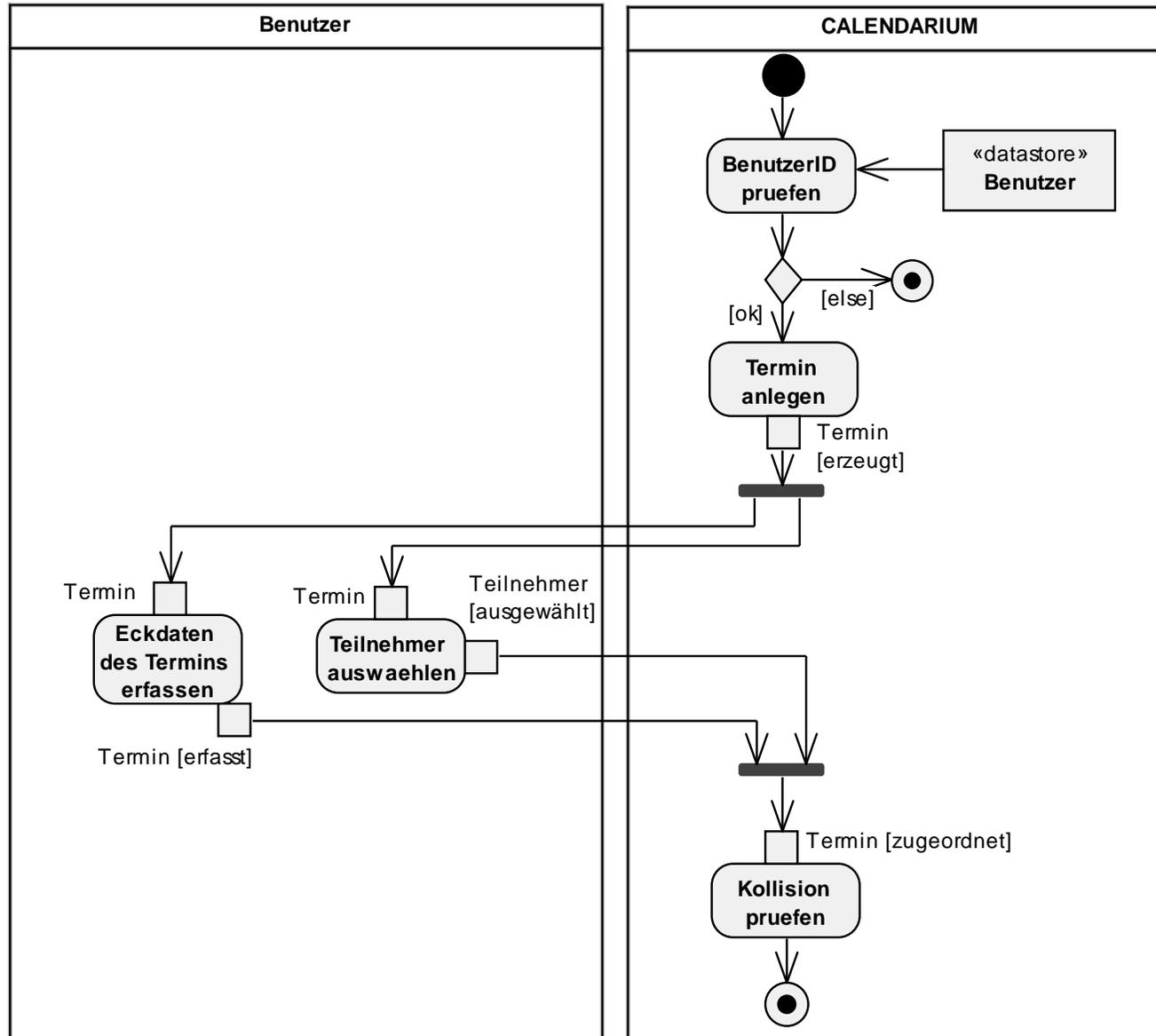


Partitionen

- Erlauben die **Gruppierung von Knoten und Kanten** einer Aktivität nach bestimmten Kriterien
- **Logische Sicht auf eine Aktivität** zur Erhöhung der Übersichtlichkeit und Semantik des Modells
- »Schwimmbahnen«-Notation (partitions, swimlanes)
- Hierarchische Partitionen
 - Zur Schachtelung auf verschiedenen Hierarchieebenen



Partitionen – Bsp.: Koordination von Terminen mit 2 Partitionen "Benutzer" und "CALENDARIUM"



Signale und Ereignisse

- Sonderformen von Aktionen
- Senden von Signalen:

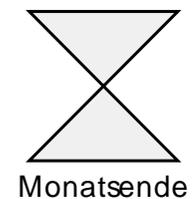


- Empfangen von Ereignissen:

Asynchrones Ereignis

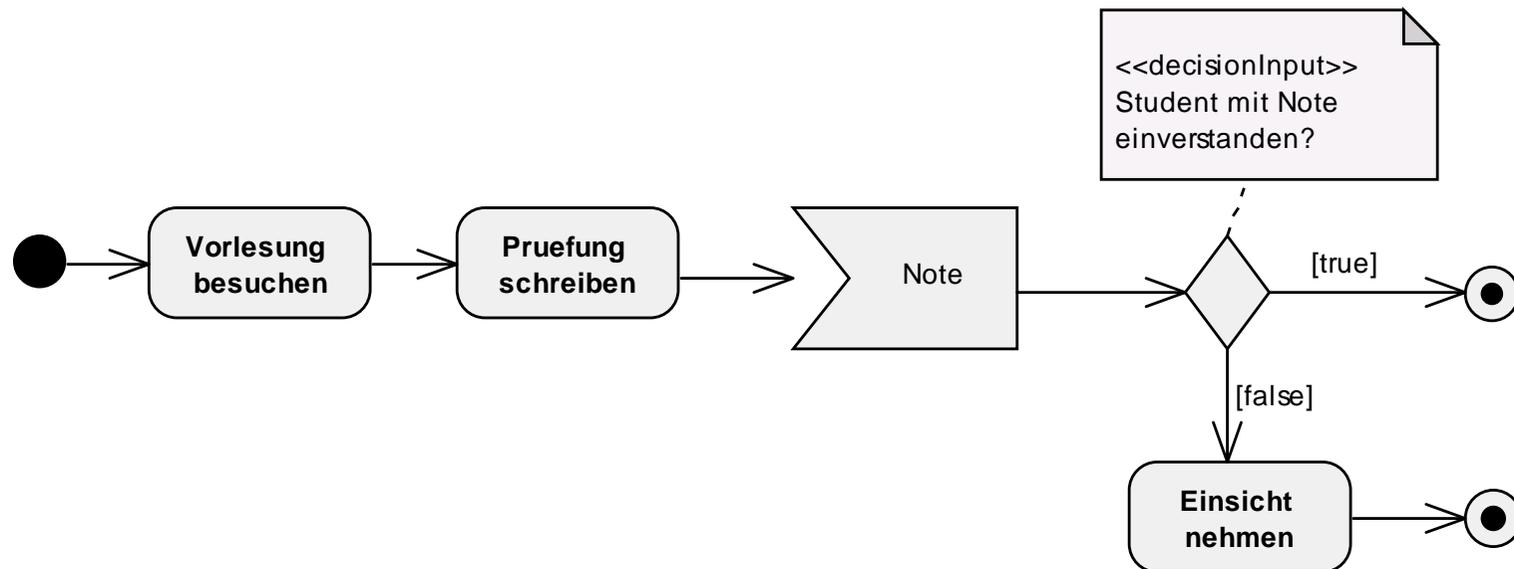


Asynchrones Zeitereignis

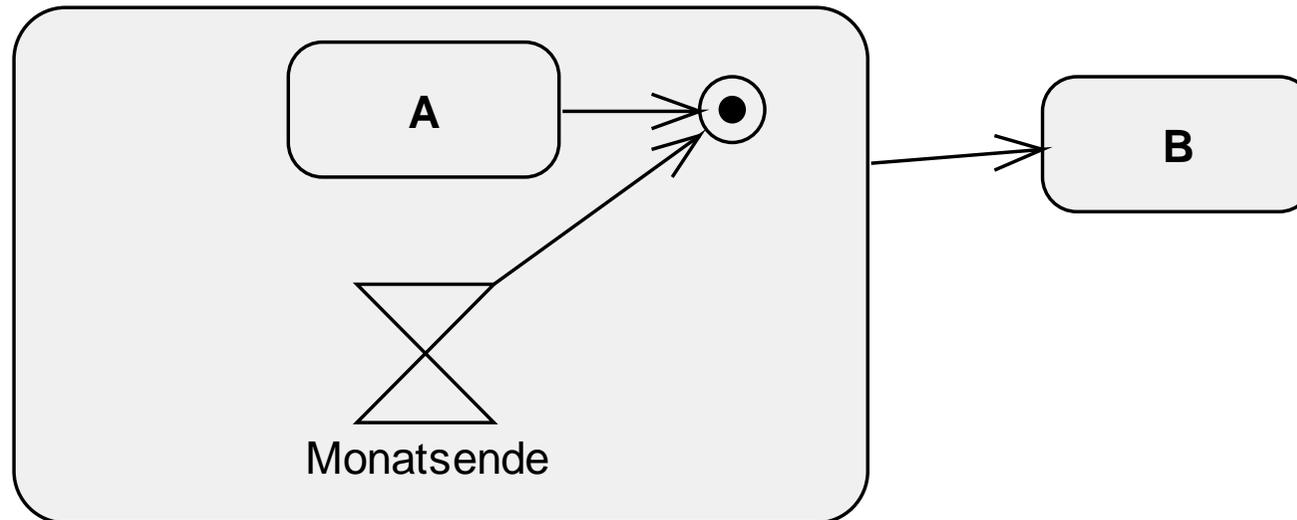


Bsp.: Asynchrones Ereignis

- Um eine Vorlesung zu absolvieren, besucht ein Student zuerst die Vorlesung, dann schreibt er eine Prüfung und wartet auf die Note. Der Student wird informiert, sobald die Note verfügbar ist. Nachdem der Student die Note erfahren hat, besteht die Möglichkeit, Einsicht zu nehmen.

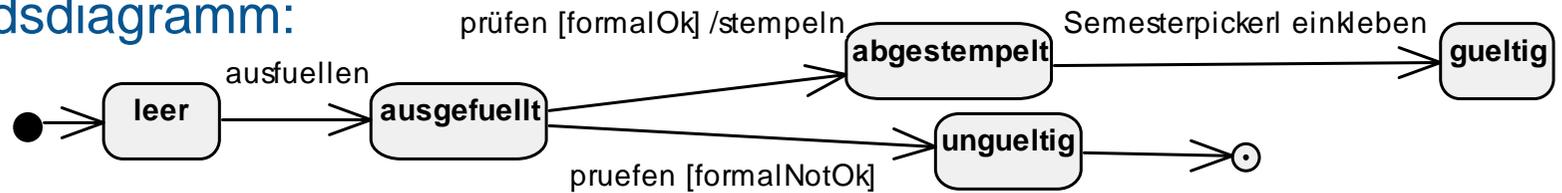


Bsp.: Asynchrones Zeitereignis

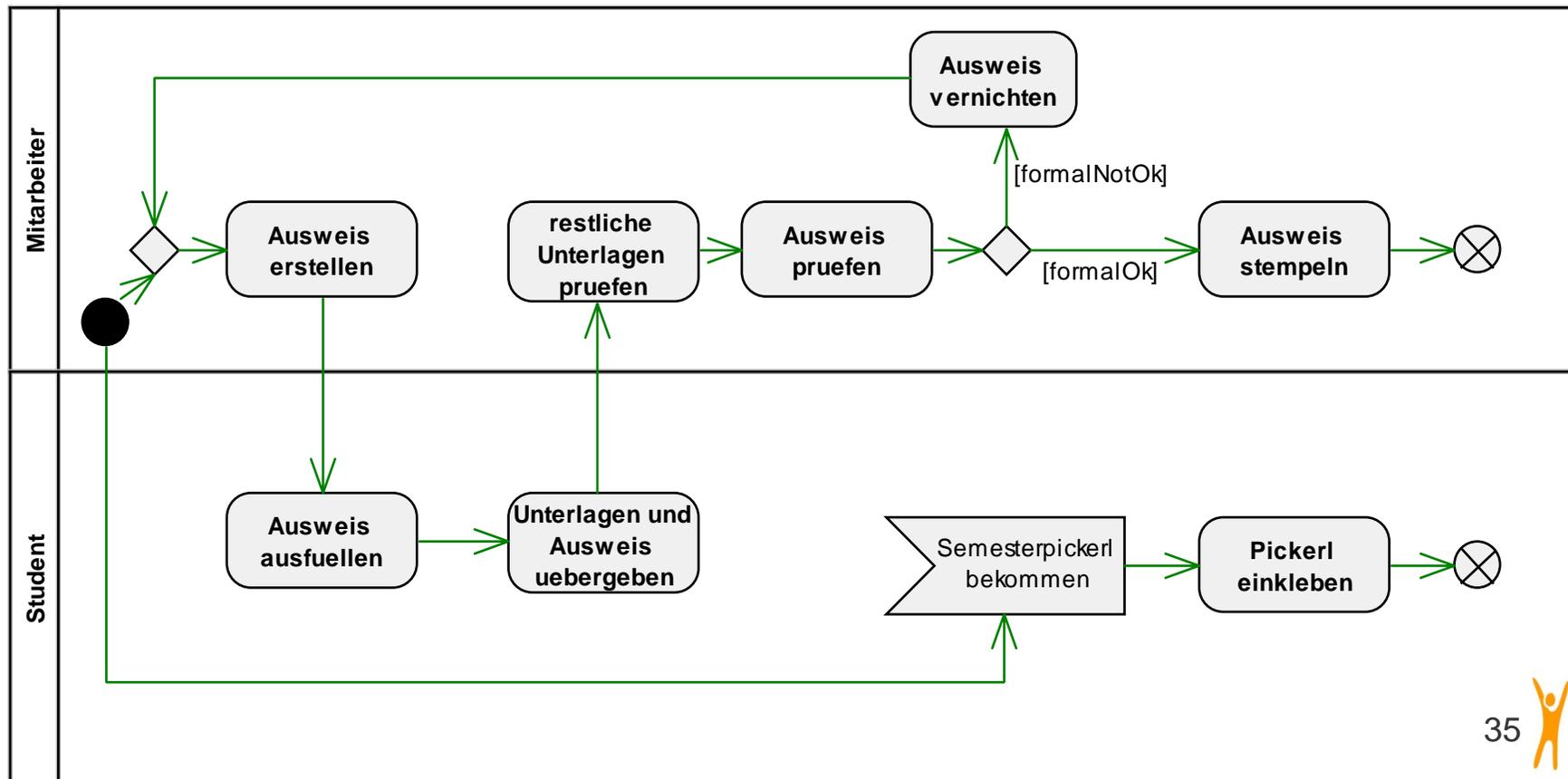


Beispiel: Studentenausweis (1/3)

- Zustandsdiagramm:

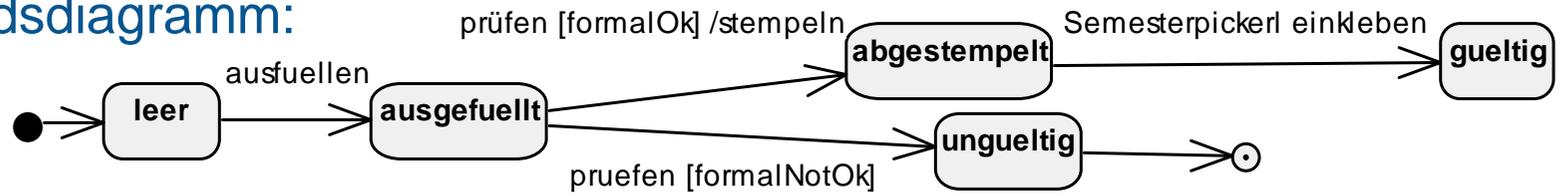


- Aktivitätsdiagramm - Kontrollfluss:

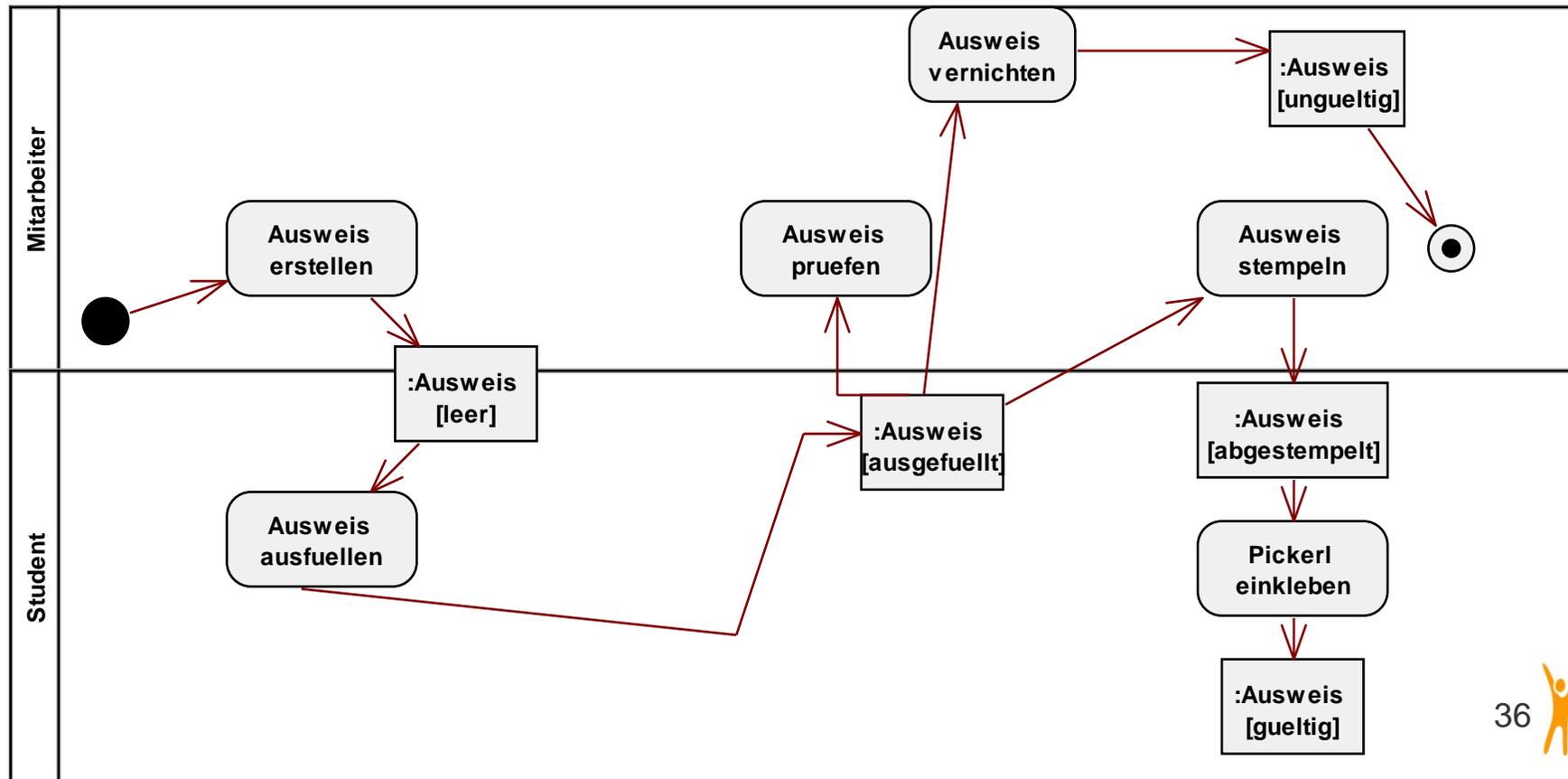


Beispiel: Studentenausweis (2/3)

- Zustandsdiagramm:

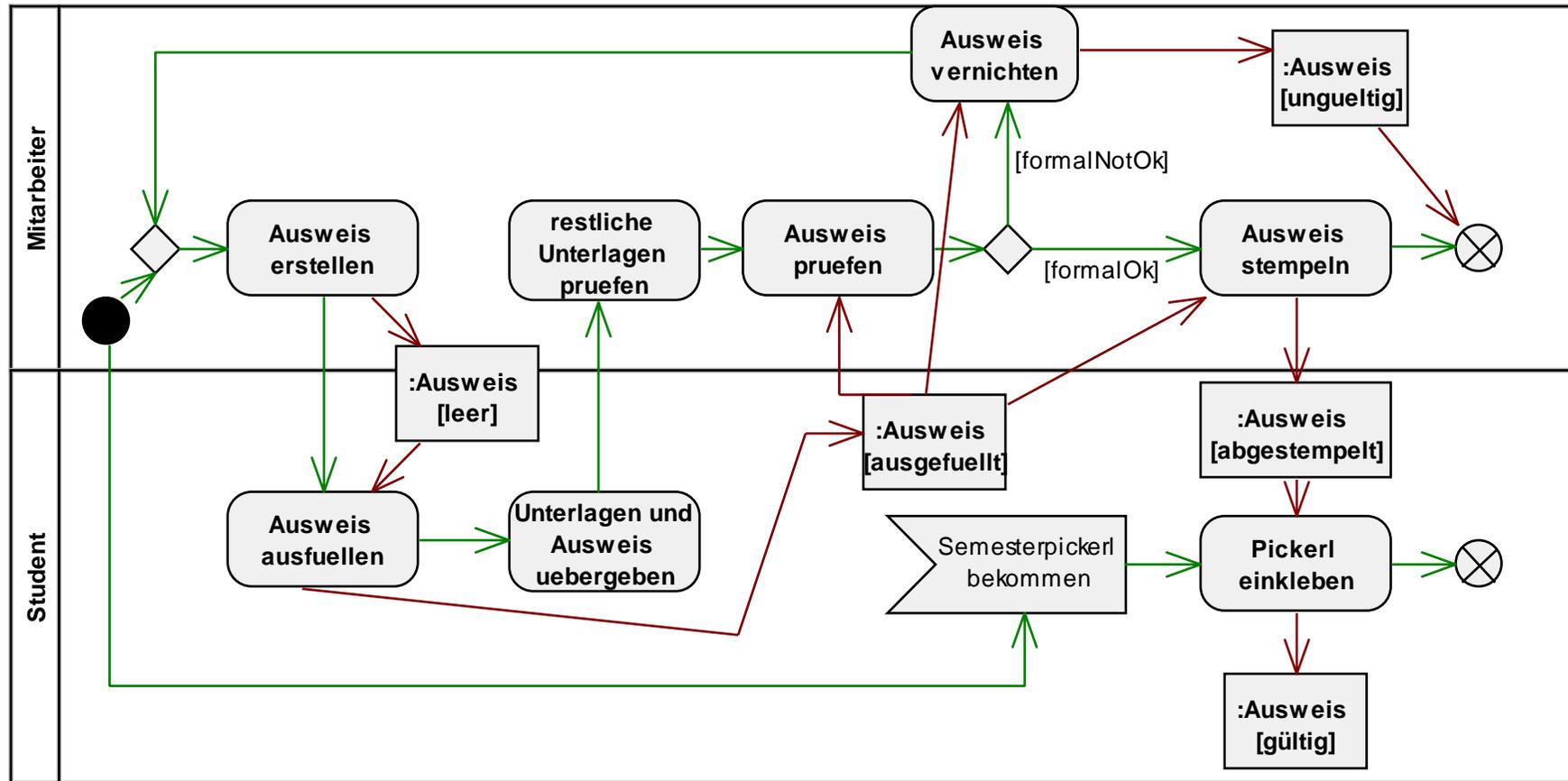


- Aktivitätsdiagramm - Objektfluss:



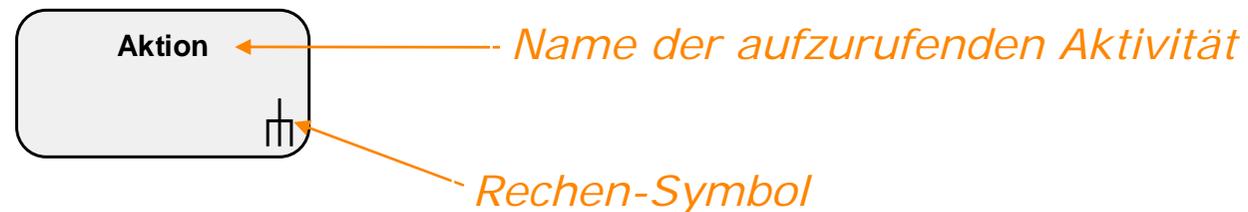
Beispiel: Studentenausweis (3/3)

- Kontrollfluss (grün) und Objektfluss (rot) in einem Diagramm



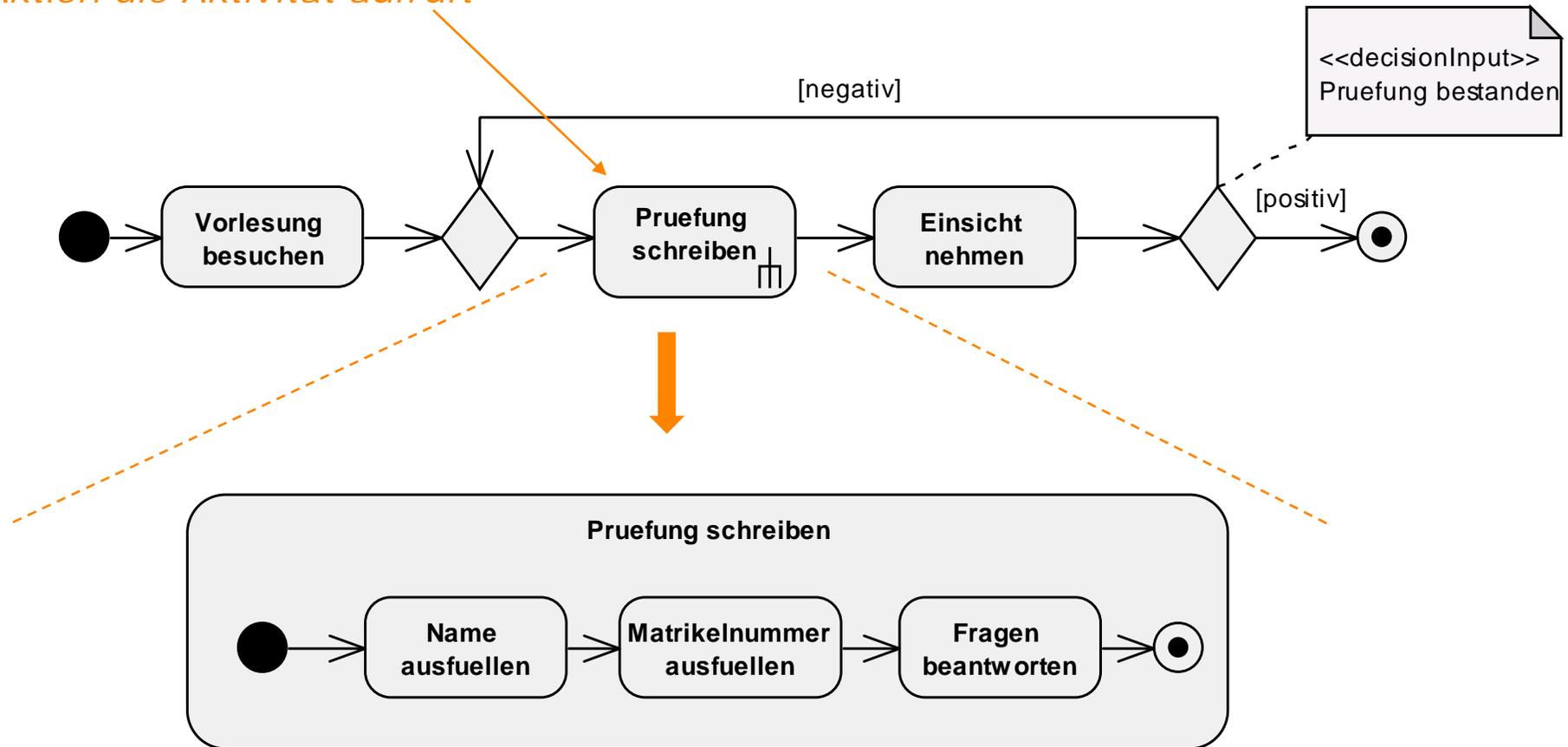
Schachtelung von Aktivitäten

- Aktivitäten können wiederum Aktivitäten aufrufen
- So können Details in eine tiefere Ebene ausgelagert werden
- Vorteile:
 - Bessere Lesbarkeit
 - Wiederverwendung
- Notation:
 - In einer Aktion wird eine Aktivität aufgerufen



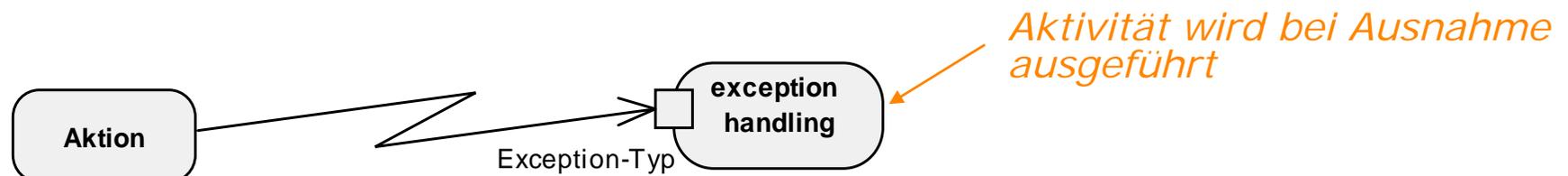
Schachtelung von Aktivitäten - Beispiel

Aktion die Aktivität aufruft



Ausnahmebehandlung – Exception Handler (1/2)

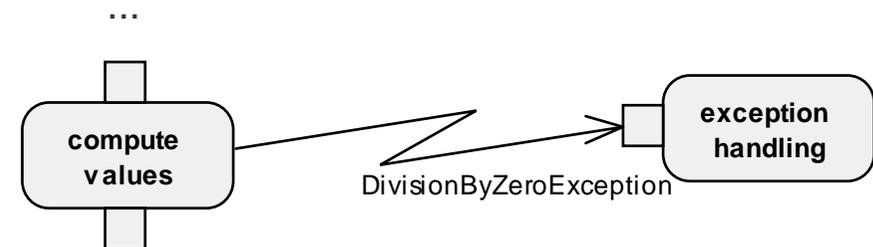
- **Vordefinierte Ausnahmen**, beispielsweise durch das Laufzeitsystem (z.B. Division durch 0)
- **Benutzerdefinierte Ausnahmen**
 - RaiseExceptionAction
- Behandlung einer Ausnahme durch **dezidierten Ausnahmebehandlungsknoten** – nach Abarbeitung der Ausnahme kann mit dem "normalen" Ablauf fortgefahren werden
- Der Ausnahmebehandlungsknoten **substituiert** den „geschützten“ Knoten und hat daher keine eigenständigen ausgehenden Kontroll- oder Objektflüsse
- Notation:



Ausnahmebehandlung – Exception Handler (2/2)

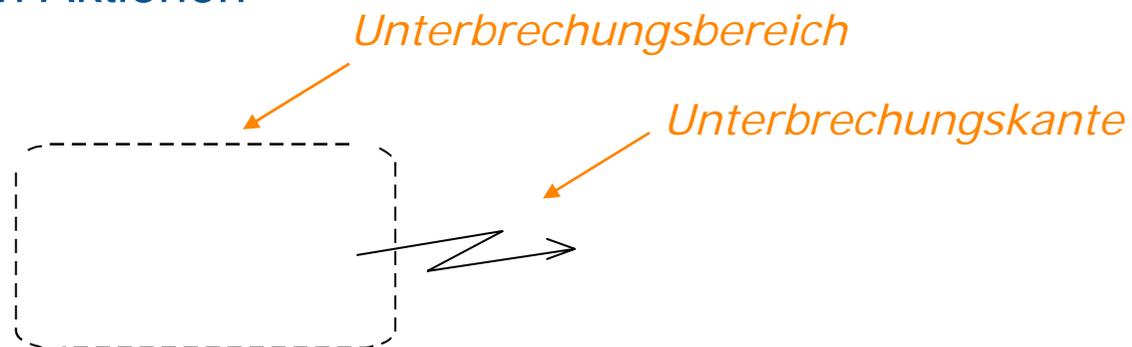
- Existiert für einen Ausnahmetyp keine Ausnahmebehandlung, wird die betroffene Aktion beendet und die Ausnahme nach außen propagiert (d.h. es wird in der umgebenden Aktivität nach passender Ausnahmebehandlung gesucht)
- Beispiel

```
try {  
    // compute values  
} catch (DivisionByZeroException) {  
    // exception handling  
}
```



Ausnahmebehandlung – Unterbrechungsbereich (1/2)

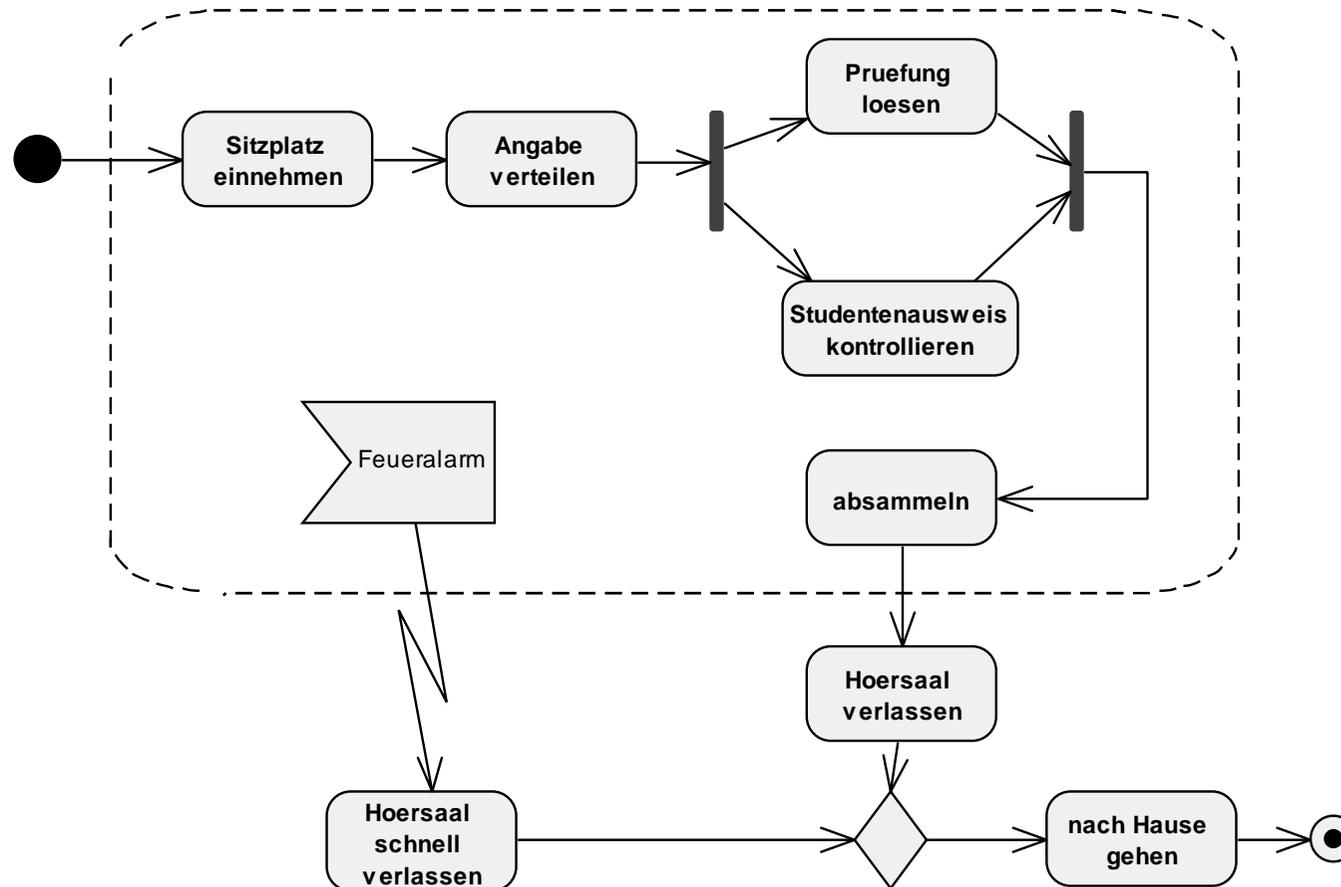
- Umschließt 1-n Aktionen
- Notation:



- Wird der Unterbrechungsbereich über die Unterbrechungskante verlassen, so werden alle in der Region vorhandenen Token gelöscht

Ausnahmebehandlung – Unterbrechungsbereich (2/2)

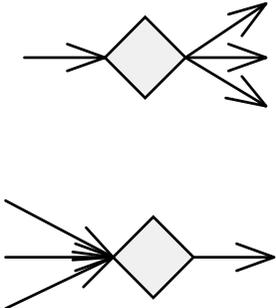
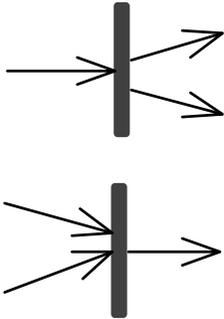
- Bsp.: Prüfung schreiben



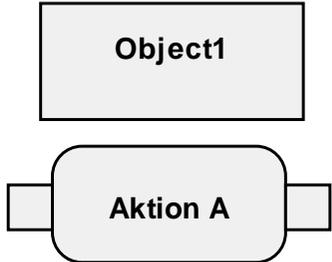
Basiselemente (1/5)

Name	Syntax	Beschreibung
Aktionsknoten		Repräsentation von Aktionen (Aktionen sind atomar!)
Initialknoten		Kennzeichnung des Beginns eines Ablaufs einer Aktivität
Aktivitätse- ndknoten		Kennzeichnung des Endes ALLER Abläufe einer Aktivität

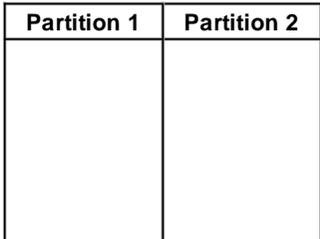
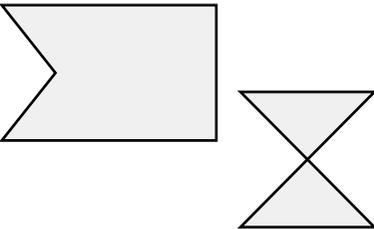
Basiselemente (2/5)

Name	Syntax	Beschreibung
Entscheidungs-/ Vereinigungsknoten		Aufspaltung/Zusammenführung von alternativen Abläufen
Parallelisierungs-/ Synchronisationsknoten		Aufspaltung eines Ablaufs in nebenläufige Abläufe / Zusammenführung von nebenläufigen Abläufen in einen Ablauf

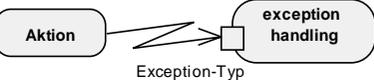
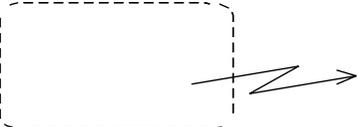
Basiselemente (3/5)

Name	Syntax	Beschreibung
Ablaufend-knoten		Ende von EINEM Ablauf
Transition		Verbindung der Knoten einer Aktivität
Aktivitätsaufruf		Schachtelung von Aktivitäten
Objektknoten, Pins		Beinhalten Daten und Objekte

Basiselemente (4/5)

Name	Syntax	Beschreibung
Partition		Gruppierung von Knoten und Kanten innerhalb einer Aktivität
Signal		Übermittlung eines Signals an einen Empfänger
asynchrones Ereignis/ Zeitereignis		Warten auf ein Ereignis bzw. einen Zeitpunkt

Basiselemente (5/5)

Name	Syntax	Beschreibung
Exception Handler		Aktivität wird bei Ausnahme ausgeführt
Unterbrechungsbereich		Wird der Unterbrechungsbereich über die Unterbrechungskante verlassen, so werden alle in der Region vorhandenen Token gelöscht

Zusammenfassung

- Sie haben diese Lektion verstanden, wenn Sie wissen ...
- was mit dem Aktivitätsdiagramm modelliert wird.
- was Aktivitäten und Aktionen sind und wie Daten- und Kontrollfluss festgelegt werden.
- dass es unterschiedliche Knoten und Kanten im Aktivitätsdiagramm gibt.
- wie Nebenläufigkeit dargestellt wird.
- was Ein- und Ausgabepins sind.
- dass Aktivitätsdiagramme partitioniert werden können.

