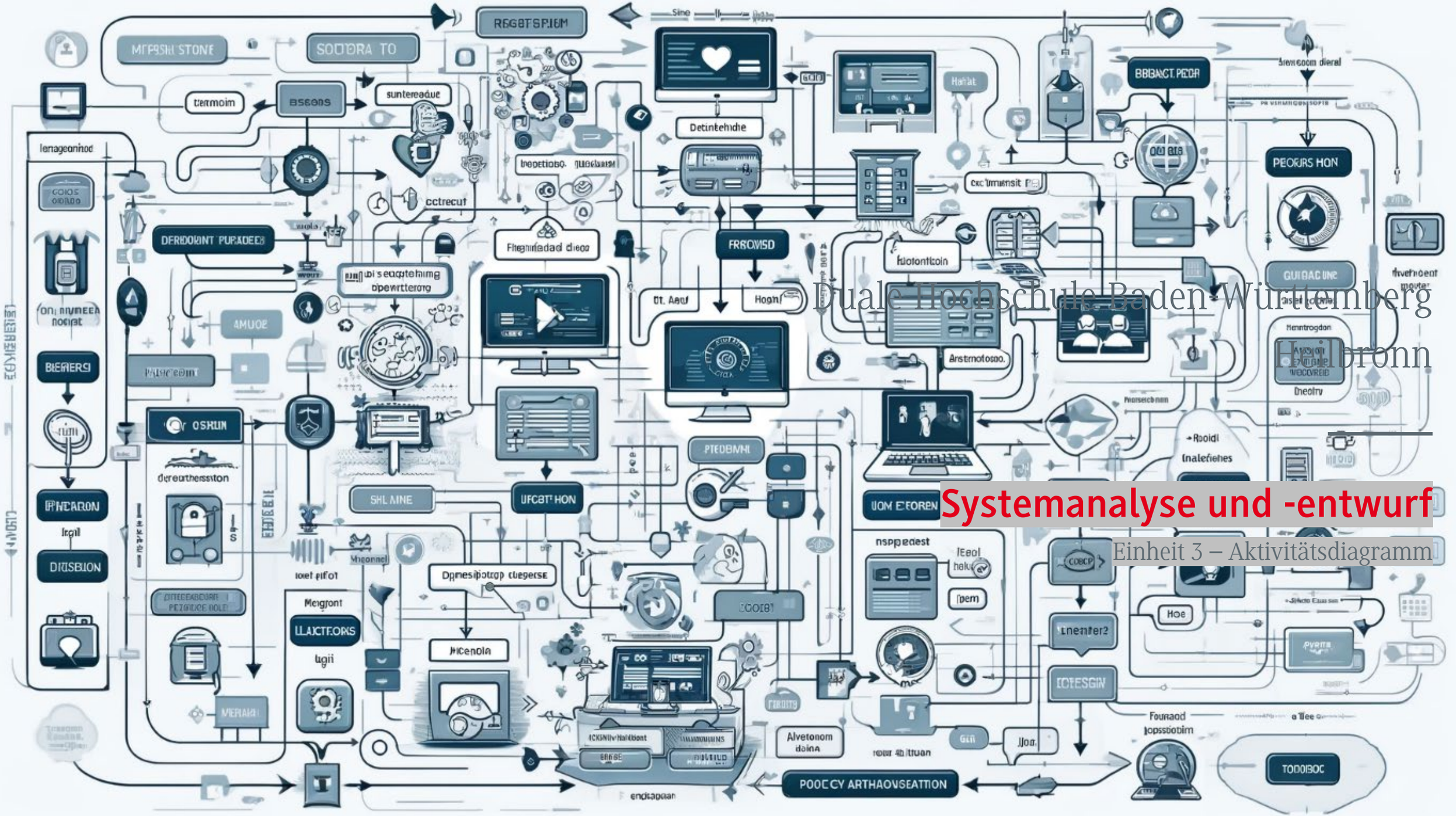


Duale Hochschule Baden-Württemberg

Heilbronn

Systemanalyse und -entwurf

Einheit 3 – Aktivitätsdiagramm

Agenda Systemanalyse und -entwurf

	Thema	Inhalte	(Geplante r) Termin
1	Einführung	Einführung in Veranstaltung. Modulhandbuch, Lernziele und Überblick, empfohlene Literatur. Requirements Game. Überblick Anforderungsmanagement und UML. Aufteilung in Teams und Themen.	31.03 NM
2	Systemanalyse und Entwurf	Das **UML-Anwendungsfalldiagramm** (Use Cases). Zweck und Bestandteile, erläutert mit Beispielen. Umsetzung eines Beispiels in Kleingruppen. Das **UML-Aktivitätsdiagramm**. Zweck und Bestandteile, erläutert mit Beispielen. Umsetzung eines Beispiels in Kleingruppen	02.04. NM
3	Systemanalyse und Entwurf	Das **UML-Aktivitätsdiagramm**. Zweck und Bestandteile, erläutert mit Beispielen. Umsetzung eines Beispiels in Kleingruppen Das **UML-Klassendiagramm**. Zweck und Bestandteile, erläutert mit Beispielen. Umsetzung eines Beispiels in Kleingruppen Vorbereitung zur Anforderungssammlung.	03.04. VM
4	Action! Anforderungssammlung	**Anforderungssammlung** für Projekte (13:30-15:00) mit Kunden oder Brainstorming. Danach (zusammen): Reflexion, Arbeit in Teams (Tooling und Modellierung), Puffer	15.04. NM
5	Guest Lecture	Guest Lecture „Moderne Software-Entwicklung in großen Unternehmen: von der Idee zum Produkt am Beispiel der SAP Internen Development Plattform (IDP) Hyperspace“. Erwartungen an Präsentationen, Puffer	17.04. VM
6	Präsentationen	**Präsentationen** und Integration der Projekte, Reflexion	14.05 NM
7	Integration	**Erstellen von Beispielsaufgaben** inkl. Musterlösungen in Teams.	21.05 NM
8	Integration	**Messe der Diagramme, Reflexion** **Verankern** des erlernten Vorgehens mit UML an weiteren Beispielen. Entweder einzeln oder in Kleingruppen.	03.06 VM
9	Abschluss	Fragerunde, Wiederholungen, Vorbesprechung **Klausur**	03.06. NM

Willkommen im Sprint 2

Sprint 2 - Wichtigsten Grundlagen und Konzepte

Für Teilnehmer/innen verborgen

Für Teilnehmer/innen verborgen

Backlog

- ✓ Als Professorin möchte ich Studierenden vermitteln, was Use Case Diagramme sind, damit sie sie einsetzen können.
- ✓ Als Professorin möchte ich Studierenden vermitteln, wie Use Case Diagramme zu modellieren sind, damit sie sie Anforderungen in Use Cases übersetzen können.
- Als Professorin möchte ich Studierenden vermitteln, was Klassendiagramme sind, damit sie sie einsetzen können.
- Als Professorin möchte ich Studierenden vermitteln, wie Klassendiagramme zu modellieren sind, damit sie sie Anforderungen in Use Cases übersetzen können.
- Als Professorin möchte ich Studierenden vermitteln, was Aktivitätsdiaramme sind, damit sie sie einsetzen können.
- Als Professorin möchte ich Studierenden vermitteln, wie Aktivitätsdiaramme zu modellieren sind, damit sie sie Anforderungen in Use Cases übersetzen können.
- ✓ Als Student:in möchte ich verstehen, wie Anforderungen erhoben werden, damit ich Kunden aufmerksam zuhören und richtige Fragen stellen kann
- Als Student:in möchte ich Notation der Diagramme kennen und richtig einsetzen, damit ich mich auf die Inhalte konzentrieren kann
- Als Student:in möchte ich klausurrelevante Inhalte möglichst früh kennenlernen, damit ich sie besser für die Klausur verinnerlichen kann

Ansicht der Veranstaltung im Moodle

Aktivitätsdiagramm

Agenda

- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Die Einheit basiert mit freundlicher Genehmigung von Frau Prof. Dr. Seidl auf Inhalten der Veranstaltung „Objektorientierte Modellierung“ an der TU Wien

Aktivitätsdiagramm

Agenda

- **Einführung**
- **Aktivitäten, Aktionen und Kanten**
- **Initialknoten, Aktivitätsendknoten, Ablaufendknoten**
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Einführung Aktivitätsdiagramm

Die Zentrale Frage: Wie realisiert ein System ein bestimmtes Verhalten?

- Fokus des Aktivitätsdiagramms: **prozedurale Verarbeitungsaspekte**
- Spezifikation von **Kontroll-** und/oder **Datenfluss** zwischen Arbeitsschritten (Aktionen) zur Realisierung einer Aktivität

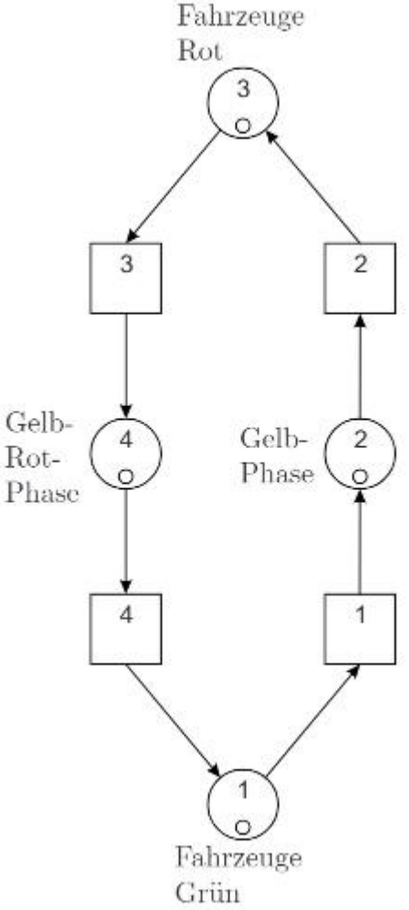
- **Aktivitätsdiagramm in UML2:**
  - ablauforientierte Sprachkonzepte
  - basierend u.a. auf Petri-Netzen

- 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

Kleiner Exkurs Petri-Netze

Was sind Petri-Netze?

- Petrinetze sind ein mathematisches Modell, mit dem man ablauforientierte, parallele und nebenläufige Prozesse formal beschreiben kann.
- Sie bestehen aus:
  - Stellen (Kreise) – symbolisieren Bedingungen oder Zustände
  - Transitionen (Balken/Rechtecke) – symbolisieren Ereignisse/Aktivitäten
  - Kanten – verbinden Stellen und Transitionen
  - Markierungen (Token) – zeigen, welche Zustände gerade aktiv sind



Verbindung zu UML2:

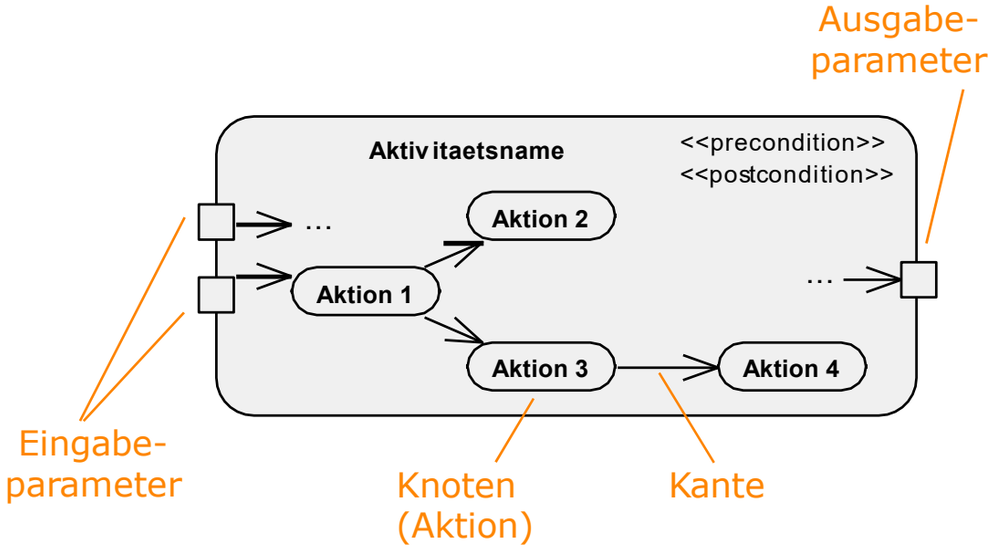
- Viele Konzepte wie Tokenfluss, parallele Abläufe, Entscheidungs- und Synchronisationsknoten stammen aus den Petrinetzen
- UML übernimmt diese Ideen, abstrahiert sie aber in eine für Softwareentwickler verständlichere grafische Sprache.

Quelle: https://der-prozessmanager.de/aktuell/wissensdatenbank/petri-netz , https://www.max-academy.de

Nicht Klausurrelevant

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 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)

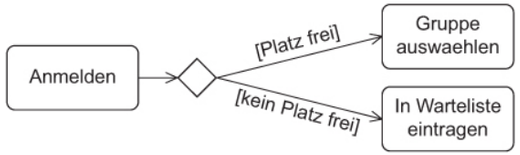
Kanten

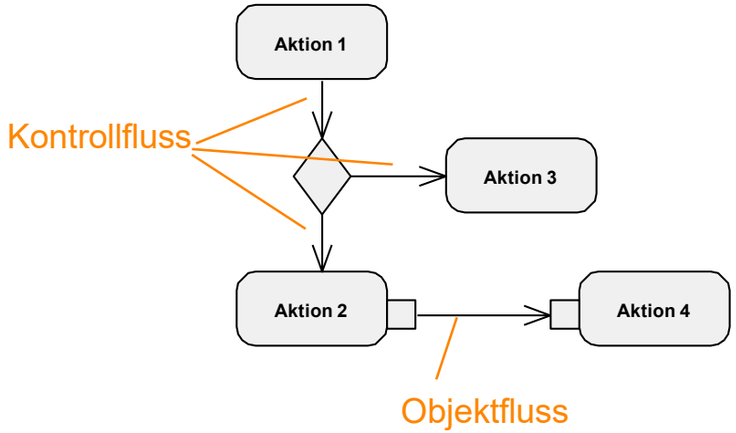
- Kanten verbinden Knoten und legen **mögliche Abläufe** einer Aktivität fest

- **Kontrollflusskanten (ControlFlow)**
  - Drücken eine reine Kontrollabhängigkeit zwischen Vorgänger- und Nachfolgerknoten aus

- **Objektflusskanten (ObjectFlow)**
  - 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
- Pro Aktivität keine oder mehrere Initialknoten erlaubt – letzteres ermöglicht Nebenläufigkeit

Aktivitätsendknoten

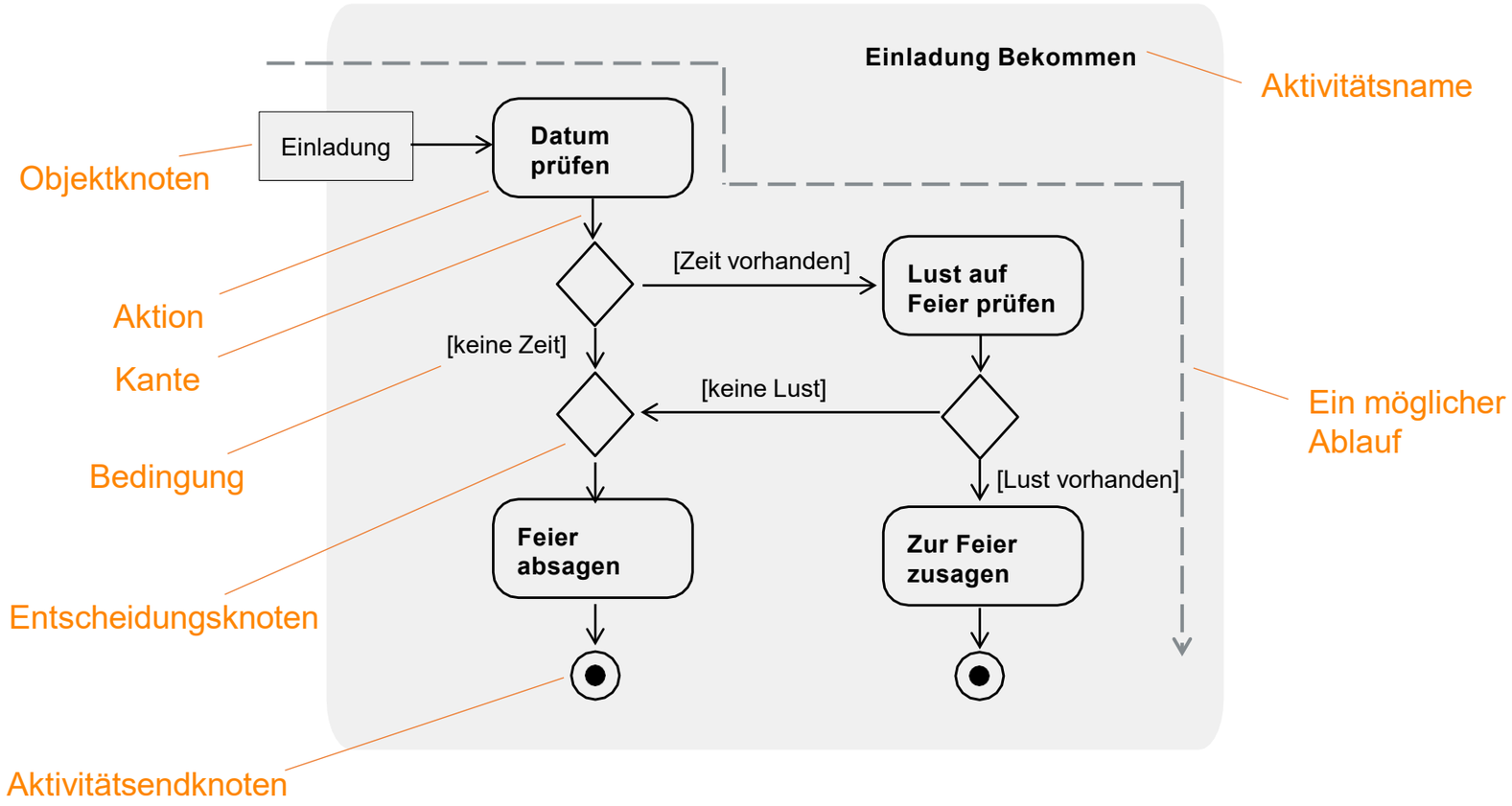
- Beendet alle Abläufe einer Aktivität sowie den Lebenszyklus eines Objekts
- Keine Ausführung weiterer Aktionen
- Pro Aktivität mehrere Aktivitätsendknoten erlaubt

Ablaufendknoten

- Beendet einen Ablauf einer Aktivität

Beispiel Feiern

- Aktivität und ihre Bestandteile



Aktivitätsdiagramm

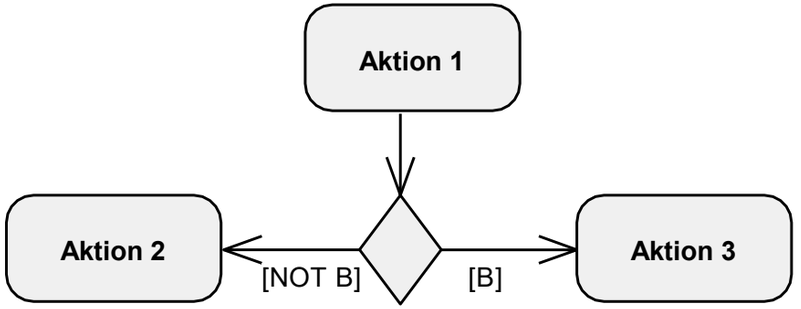
Agenda

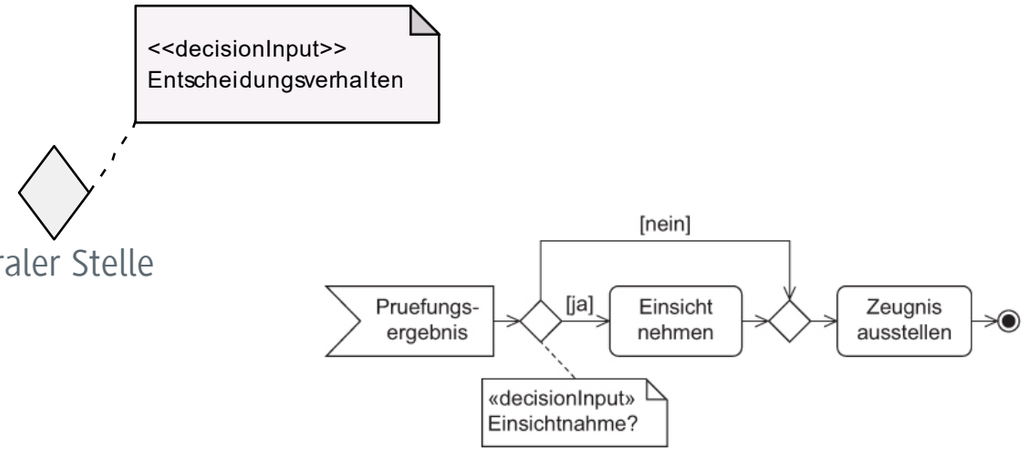
- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- **Alternative Abläufe und parallele Abläufe**
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Alternative Abläufe

Entscheidungsknoten

- Definiert alternative Zweige und repräsentiert eine »**Weiche**«
  - 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

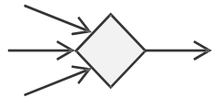




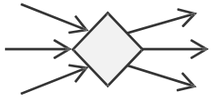
Alternative Abläufe

Vereinigungsknoten

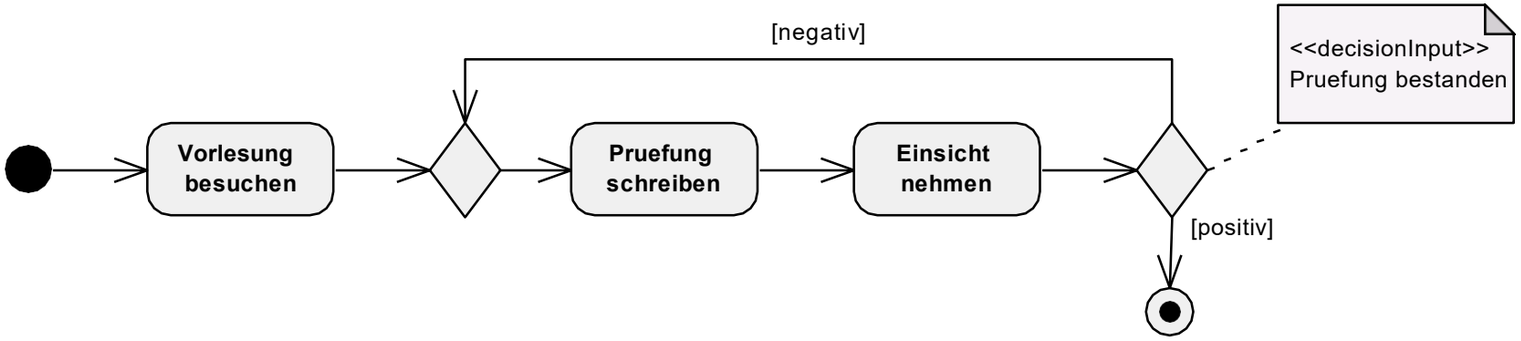
- Ein Vereinigungsknoten führt alternative (keine nebenläufigen!) Abläufe wieder zusammen



- Kombinierter Entscheidungs- und Vereinigungsknoten:



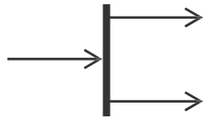
- **Beispiel:**



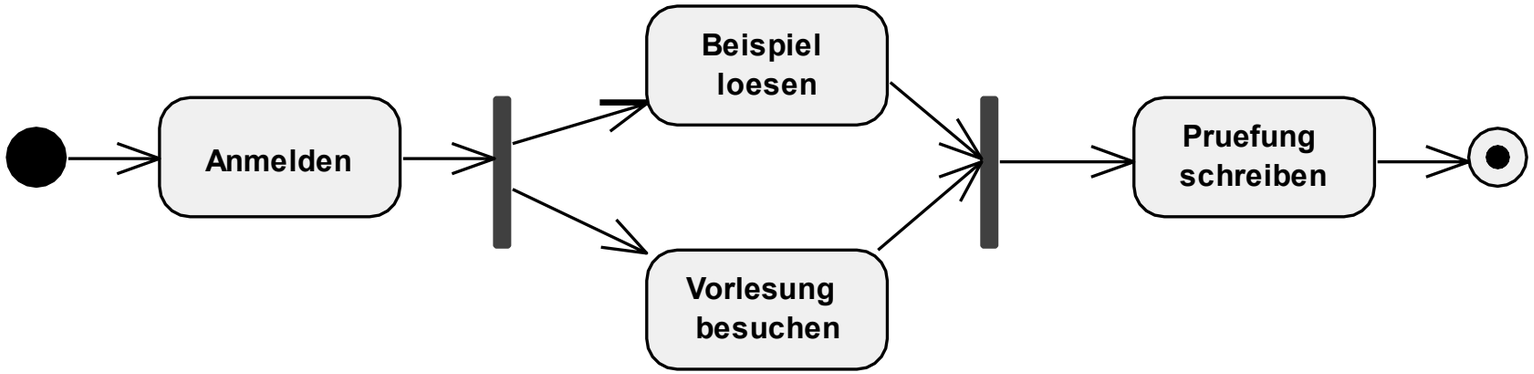
Nebenläufige Abläufe

Parallelisierungsknoten

- Zur Modellierung der Aufspaltung von Abläufen



- **Beispiel:**

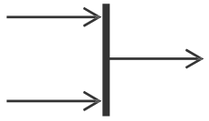


Alle Aktionen innerhalb des Parallelisierungsknoten müssen fertig ausgeführt werden, bevor es weiter geht.

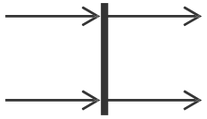
Nebenläufige Abläufe

Synchronisierungsknoten

- Führt nebenläufige Abläufe zusammen

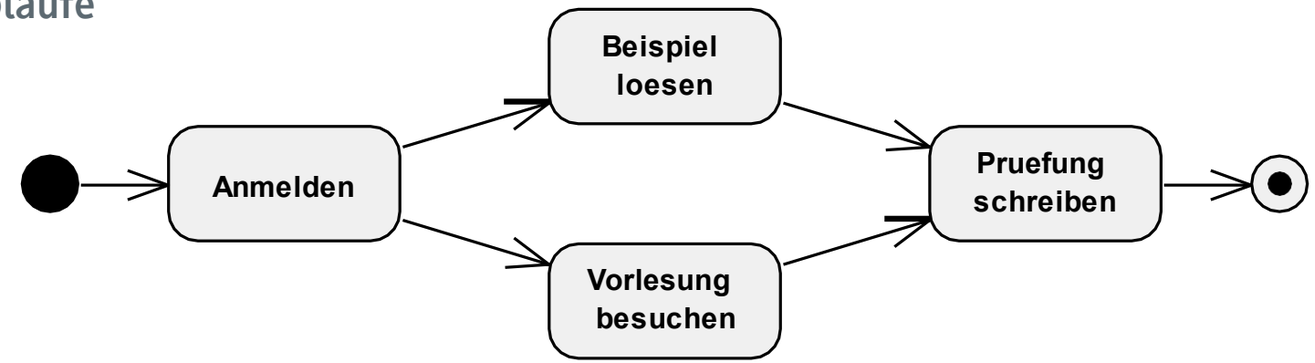


- Kombinierter Parallelisierungs- und Synchronisierungsknoten:

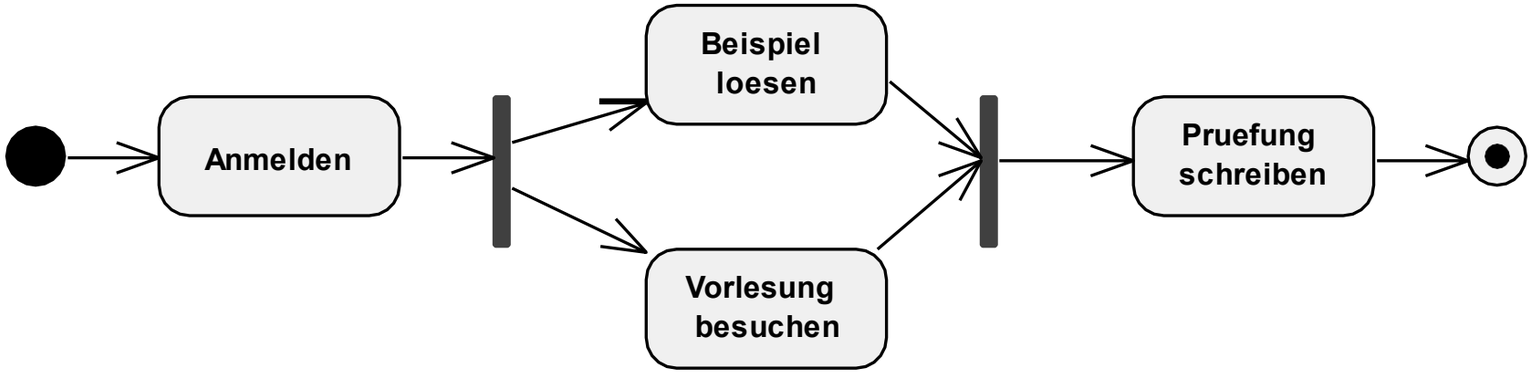


Nebenläufige Abläufe

Alternative Darstellung nebenläufiger Abläufe



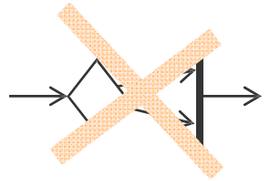
Besser:



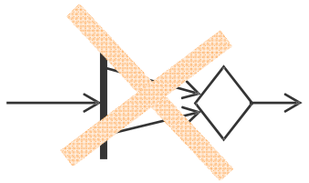
Nebenläufige Abläufe

Falsche Modellierungen

- Der Prozess nach dem Entscheidungsknoten wird nur an einer der beiden ausgehenden Kanten weiter gehen. Der Synchronisierungsknoten benötigt aber zwei eingehende Kanten um diese zu vereinigen und weiterzugeben



- Vom Parallelisierungsknoten gehen normalerweise mehrere Kanten raus, die später an den Vereinigungsknoten weitergegeben werden. I.d.R. kein beabsichtigtes Verhalten die Kanten direkt danach zusammen zu führen



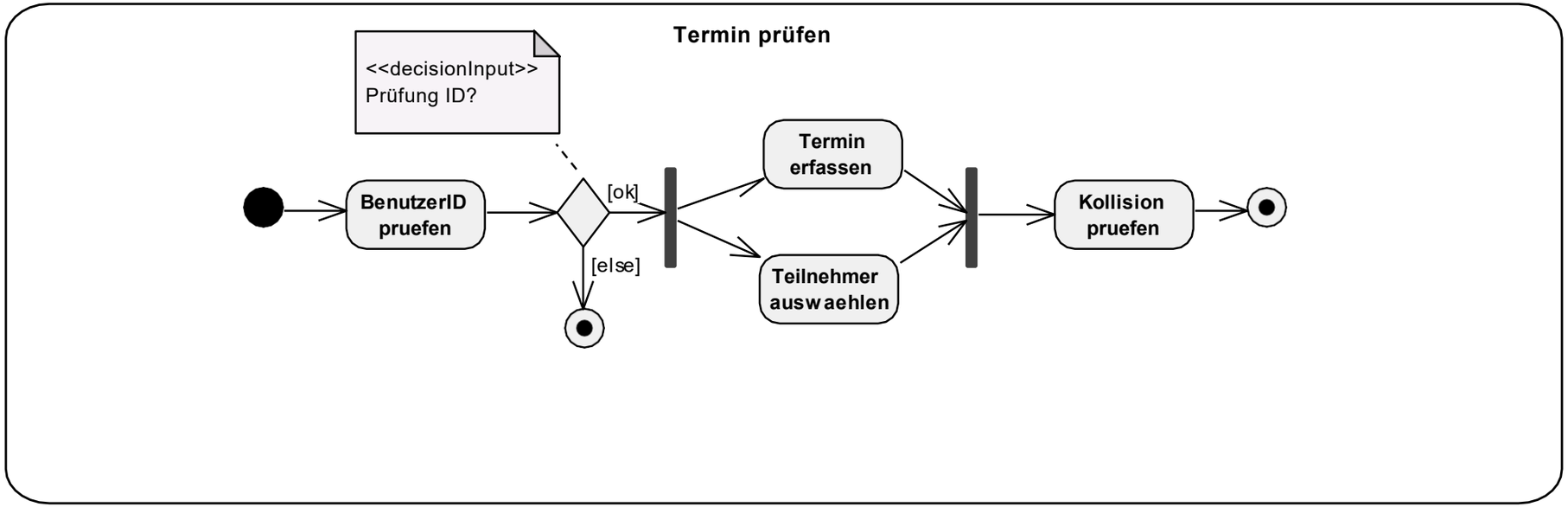
Nebenläufige Abläufe

- **Beispiel: Einladung im CALENDARIUM**

- Denken Sie an Start und Ende
- Beschreibung:
  - Wir möchten modellieren, wie ein Benutzer in der Anwendung CALENDARIUM einen Termin erfassen, parallel Teilnehmer für diesen Termin auswählen und danach prüfen kann, ob eine Kollision zwischen dem Termin und der Verfügbarkeit der Teilnehmer auftritt. Hierzu prüfen wir zu Beginn, ob der Benutzer für die Aktion berechtigt ist. Ist dem der Fall, ist parallel die Terminerfassung und die Teilnehmerauswahl möglich. Erst wenn diese beiden Aktionen abgeschlossen sind, kann die Kollisionsprüfung durchgeführt werden.
- Erledigen Sie die Aufgabe in Kleingruppen. Sie haben 12 Minuten Zeit
- Gemeinsame Ergebnisdiskussion

Nebenläufige Abläufe

- **Lösung: 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.



Bekannte Notationen

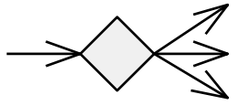
Aktionsknoten

Initialknoten

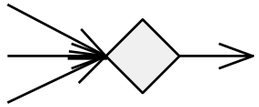
Aktivitätsendknoten

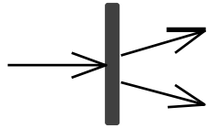
Ablaufendknoten

Transition

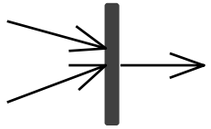


Entscheidungs-/ Vereinigungsknoten





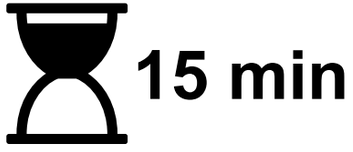
Parallelisierungs-/ Synchronisationsknoten



Modellierungsaufgabe

Modellieren Sie einen typischen morgen von einem Studium an einer Hochschule

- An dem Sie durch den Wecker aufwachen und den Wecker evtl. ein paar Mal snoozen bevor Sie aufstehen
- Wenn es keine Vorlesung morgens gibt, machen Sie Sport
- Dann machen Sie sich fertig (Duschen, Essen, Anziehen in beliebter Reihenfolge)
- Dann gehen Sie aus dem Haus



Aktivitätsdiagramm

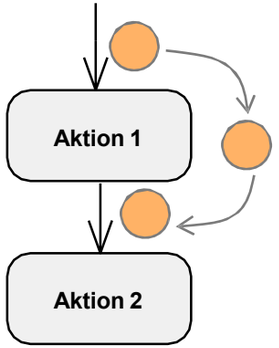
Agenda

- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- **Token**
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Token

Der Stein, der alles ins Rollen bringt

- Token ist ein **logisches Konzept:** „Virtueller Koordinationsmechanismus“
- Tokens werden **nicht modelliert**
- Token beschreibt **möglichen Ablauf** einer Aktivität zur Laufzeit
- In einer Aktivität können **beliebig viele Tokens** unterwegs sein
- Token fließen entlang der Kanten von Vorgänger- zu Nachfolgerknoten
  - können nicht auf einer Kante „warten“ (Ausnahme: Synchronisationsknoten)
  - lösen Verarbeitung aus

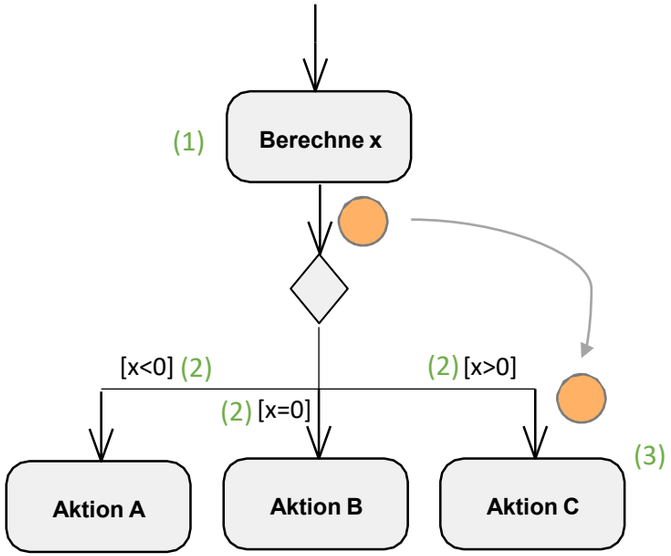


- Unterscheidung in Kontroll- und Datentoken
  - Kontrolltoken: "Ausführungserlaubnis" für den Nachfolgeknoten (sobald an allen eingehenden Kanten ein Kontrolltoken anliegt)
  - Datentoken: Transport von Datenwert oder Referenz auf Objekt
- Überwachungsbedingung kann Weitergabe von Token verhindern (Ansammlung mehrerer Token im Vorgängerknoten)

Token: Verzweigung und Vereinigung

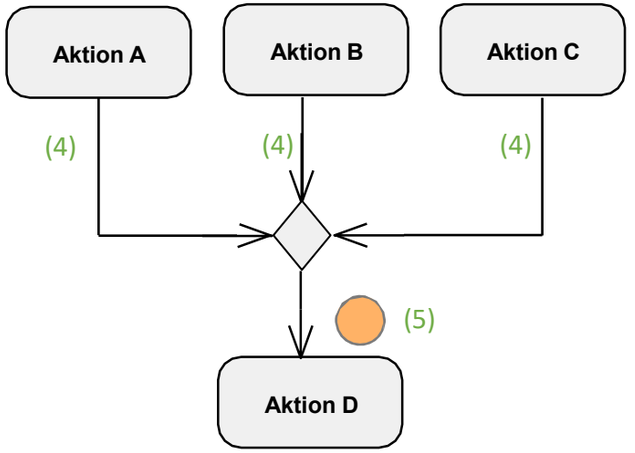
Alternativabläufe

- Eine Art „Wegkreuzung“ für die Token
- Abhängig vom Ergebnis der Aktion „Berechne x“ (1) liegt das Token nur bei einer Folgeaktion an, für die die Bedienung zutrifft
- Z.B. x=27, d.h. das Token würde nur bei Aktion C (3) anliegen



Zusammenführung von Alternativzweigen zu einem Zweig

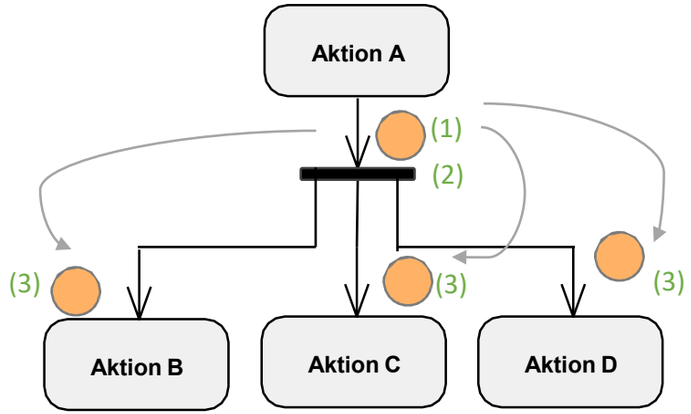
- Verschiedene Alternativzweige (4) lassen sich zu einem Zweig zusammenführen
- Hier liegt das Token – egal von welcher Aktion es kommt – immer bei Folgeaktion an (5)



Token: Parallelabarbeitung und Synchronisation

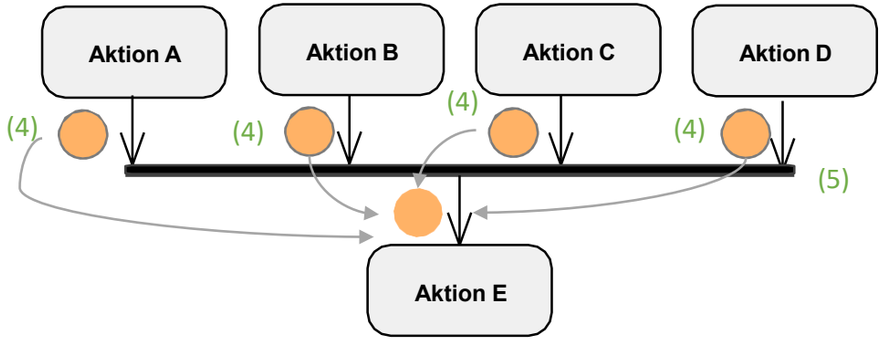
Nebenläufige Aufarbeitung von Aktionen

- Aufteilung eines Ablaufs
- Alle Aktionen werden parallel aufgearbeitet
- Ausganstoken (1) wird am „Splittingknoten“ (2) vervielfacht, so dass er an **jeder** Folgeaktion (3) anliegt
- „Teilabläufe“ können unabhängig von einander abgearbeitet werden



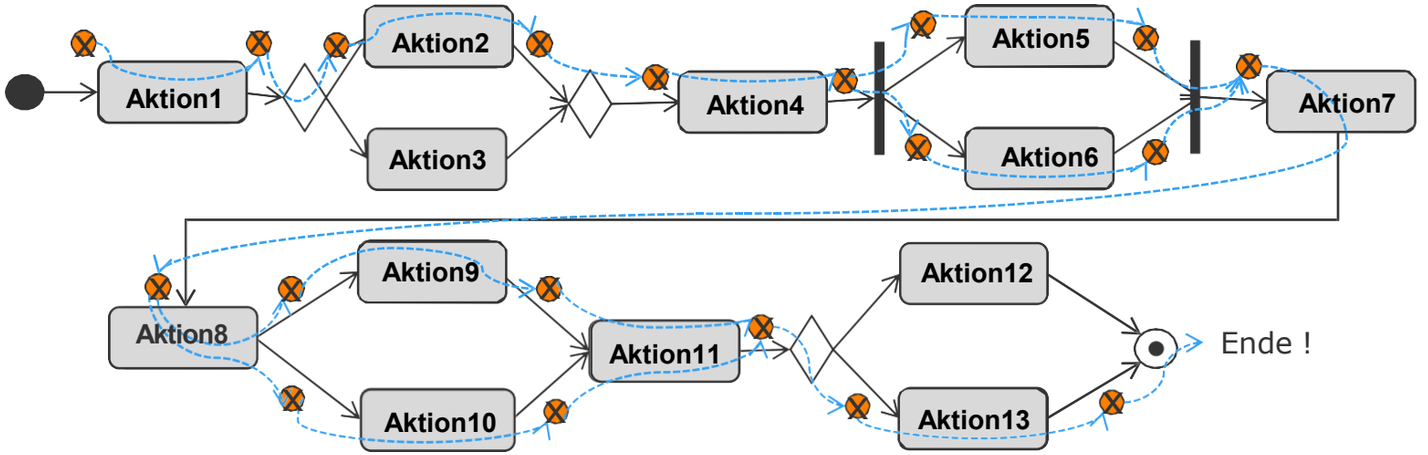
Zusammenführen paralleler Abläufe

- UND-Synchronisation: Tokens werden auf dem Synchronisationsknoten **aufgesammelt bis alle ankommen**
- Danach werden sie zu einem Token verschmolzen (5), das bei der Folgeaktion anliegt



Token

Beispiel Kontrollfluss



… zu Beginn werden alle vom Initialknoten ausgehenden Kanten mit einem Token belegt…. Laufzeit auf der Kante = 0, Verweilsdauer im Knoten >0

… 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

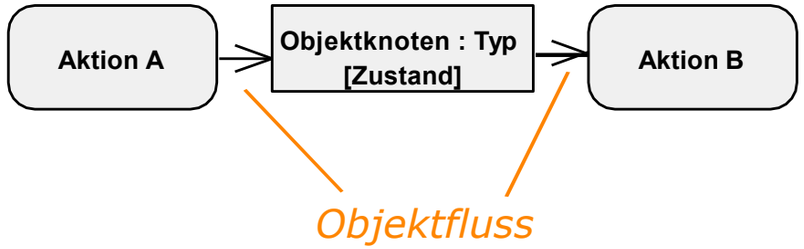
Aktivitätsdiagramm

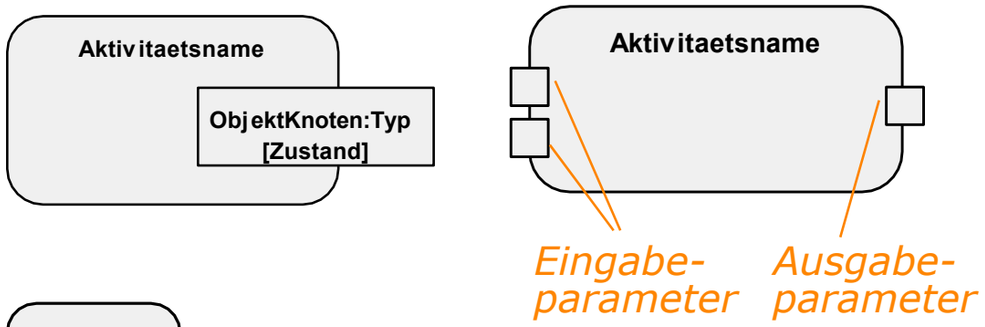
Agenda

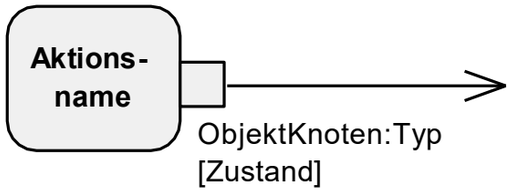
- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- **Objektknoten und Objektfluss**
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Objektknoten (1/7)

- Inhalt: **Datentoken**
- Kein Objekt im Sinne einer Klasseninstanz, sondern logisches Gerüst für den Transport der Daten und Werte
- Objektknoten stehen durch Objektflüsse in Beziehung zu einander
- 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)

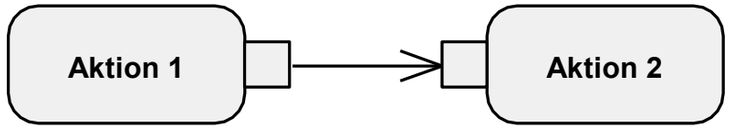






Objektknoten (2/7)

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

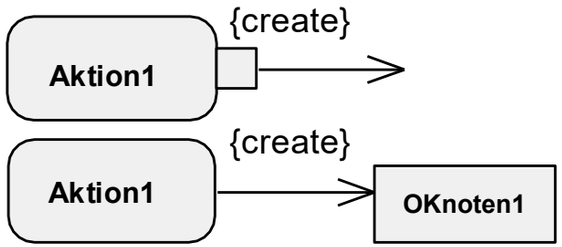


  - Weitere Notationsvariante z.B.:

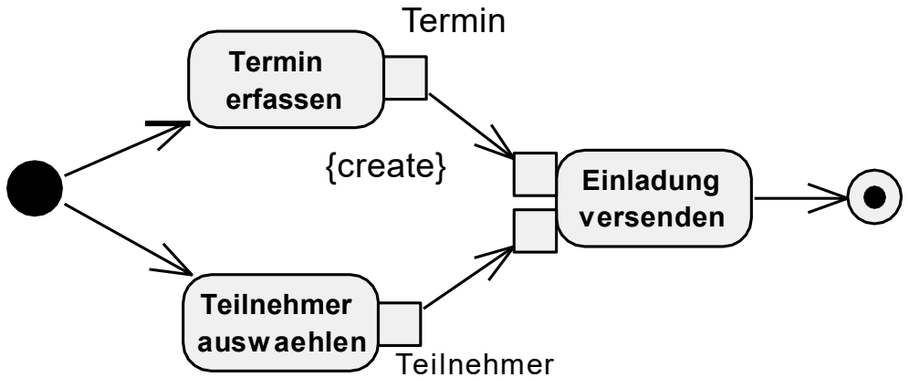
  - Objektknoten die gleichzeitig Aus- und Eingabeparameter sind, können ohne Pin-Notation dargestellt werden (Name und Typ müssen gleich sein)
    - Beispiel: Cocktail wird von Aktion 1 zu 1 durchgereicht → lässt sich mit Objektknoten darstellen.
    - Wäre anders wenn von Aktion 1 Getränk der Output wäre und bei 2 Cocktail der Input → dann ist es die Pin-Notation
    - Kompatibilität der Objektknoten erforderlich:

Objektknoten (3/7)

- Effekte einer Aktion auf Daten und Objekte
  - create, read, update und delete



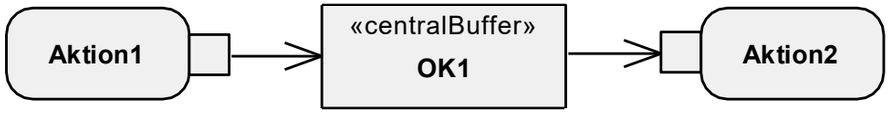
  - Beispiel für Ein- und Ausgabepins und Effekte



Objektknoten (4/7)

- Konstanter Eingabewert – Wertepin

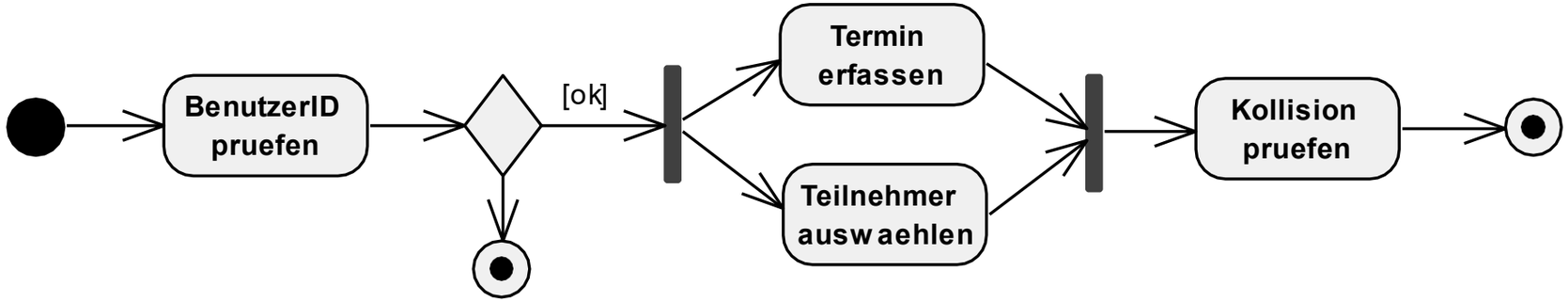
  - Zur Übergabe konstanter Werte
  - **Startet nicht die Verarbeitung eines Knotens**
- Pufferknoten
  - Zentrale Pufferung von Datentoken
  - Transienter Pufferknoten (central buffer node)
    - **Löscht Datentoken**, sobald er sie weitergegeben hat



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

Objektknoten: Beispiel

- **Beispiel: Erweiterung - Einladung im CALENDARIUM**
  - Erweitern Sie das vorherige Beispiel mit der Termin-Teilnehmer-Kollisionsprüfung wie unten stehend



  - alle Benutzer werden im persistenten Puffer "Benutzer" gespeichert (Datenbank). Auf diesen kann bei Aktionen zurückgegriffen werden, die Daten zu den Benutzern benötigen
  - Ausgewählte Benutzer werden in einem transienten Puffer zwischengespeichert und erst für die Versendung von Einladungen wieder entnommen.
- Erledigen Sie die Aufgabe in Kleingruppen
- Gemeinsame Ergebnisdiskussion


8 min

Objektfluss (1/4)

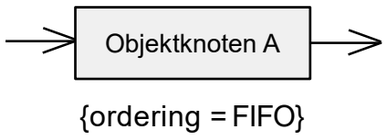
- Objektknoten ist ein Notationsmittel um den Fluss von Objekten oder Werten zu modellieren
- Der Knoten an sich stellt kein Objekt bzw. Datenwert dar
- Eingehende Token (1) repräsentieren Daten oder Werte, die im Objektknoten gesammelt werden
- Ausgehende Token (2) transportieren das Objekt bzw. Daten des Objekts in die Folgeaktion

- 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 Token:
  - Reihenfolge
  - Kapazitätsobergrenze und Gewicht
  - Selektionsverhalten
  - Transformationsverhalten

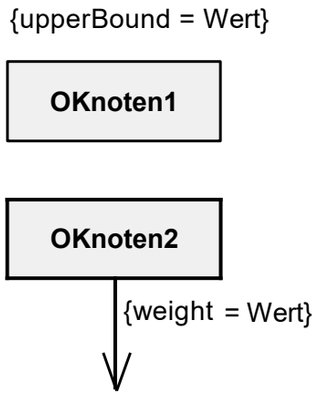
Objektfluss (2/4): Reihenfolge der Tokenweitergabe

- 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

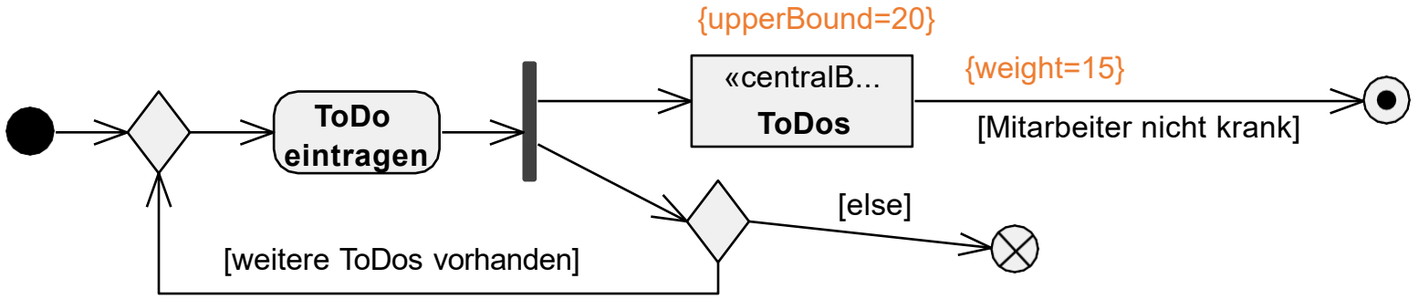


Objektfluss (3/4): Kapazitätsobergrenze und Gewicht

- **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



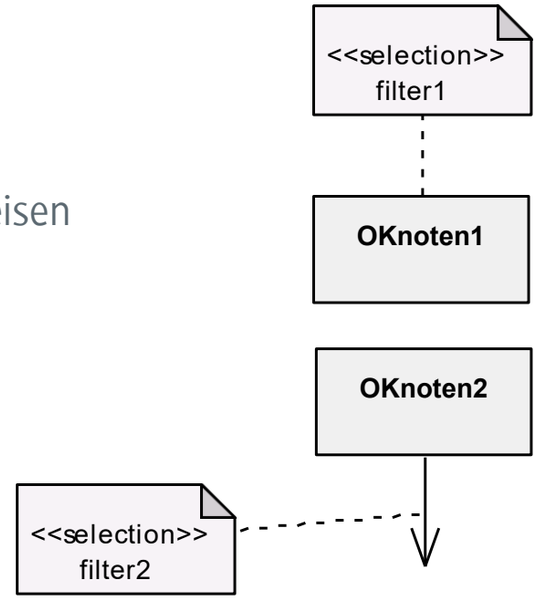
- **Beispiel:** Pufferknoten „ToDos“ kann max. 20 ToDos aufnehmen.
  - Ihr Chef bittet Sie, eine ToDo Liste für die Abteilung zu pflegen
  - Sie tragen ein ToDo in eine Liste ein und parallel entscheiden ob weitere ToDos vorhanden sind
  - Wenn mind. 15 ToDos vorhanden sind und der Mitarbeiter nicht krank ist, werden 15 davon an den Aktivitätsendknoten weitergereicht

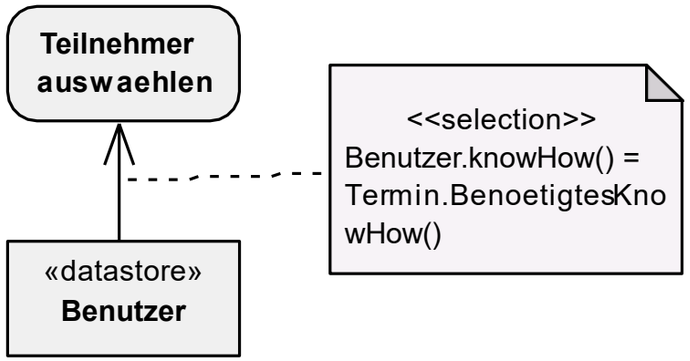


Objektfluss (4/4): Selektionsverhalten

- 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:





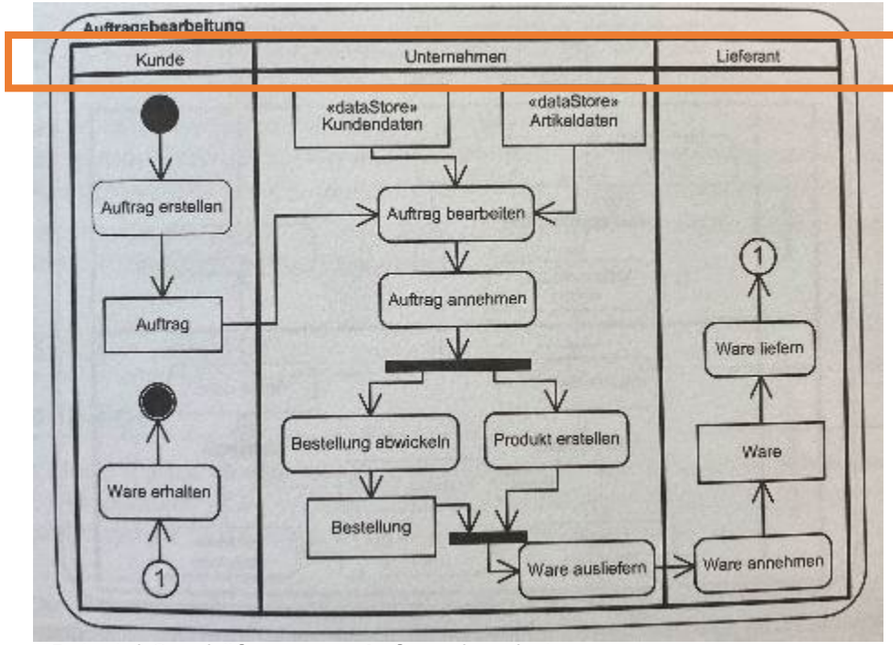
Aktivitätsdiagramm

Agenda

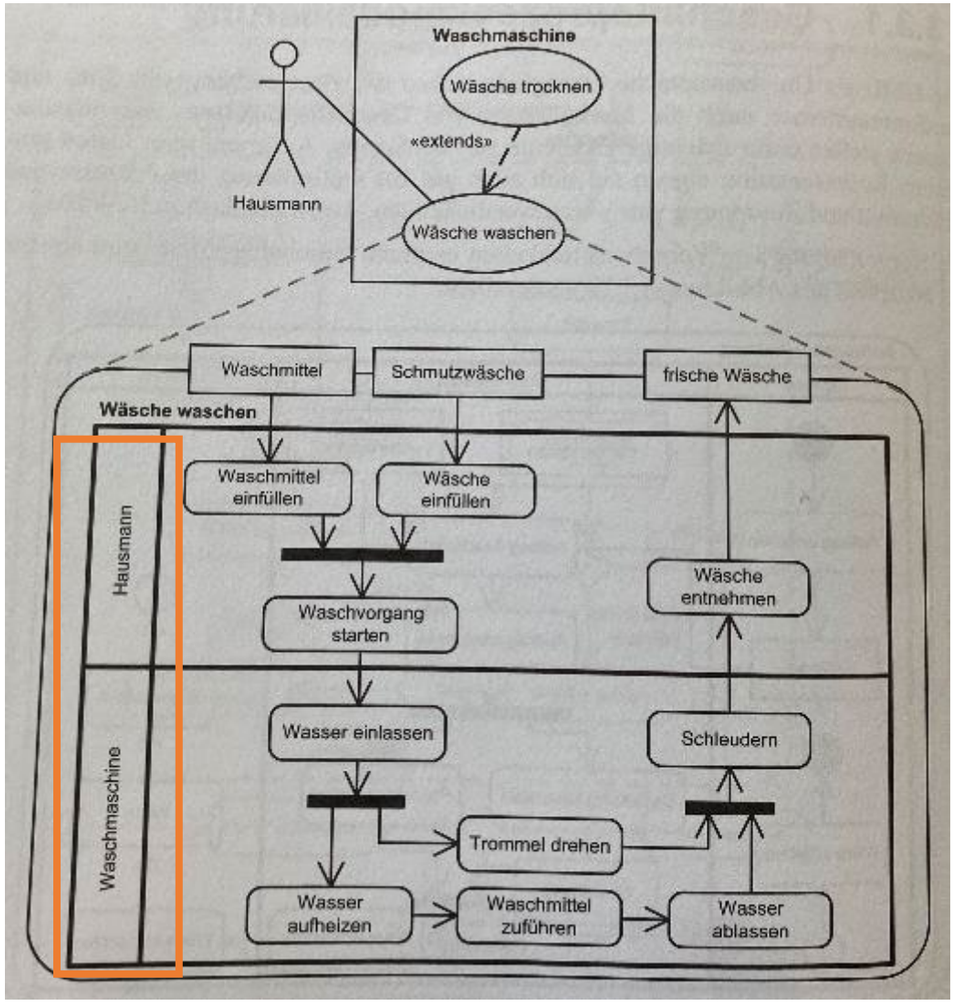
- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- **Partitionen, Signale und Ereignisse**
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- Zusammenfassung Elemente / Rückblick

Anwendung der Aktivitätsdiagramme im Projekt

- Geschäftsprozessmodellierung
- Beschreibung von Use-Cases
- Implementierung einer Operation



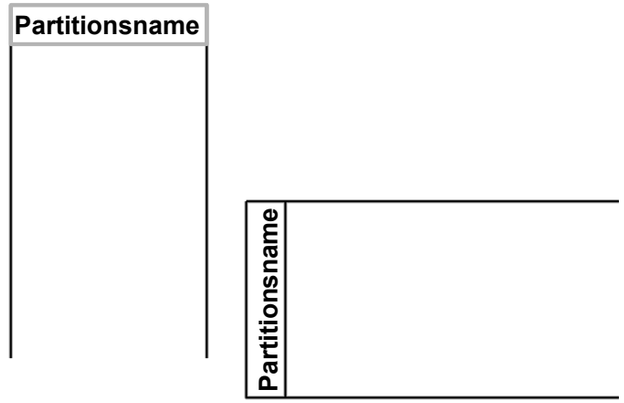
Beispiel Geschäftsprozess Auftragsbearbeitung



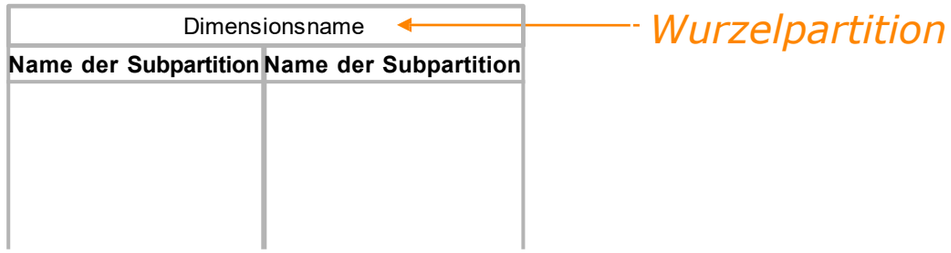
Beispiel Use Case: Wäsche Waschen

Partitionen

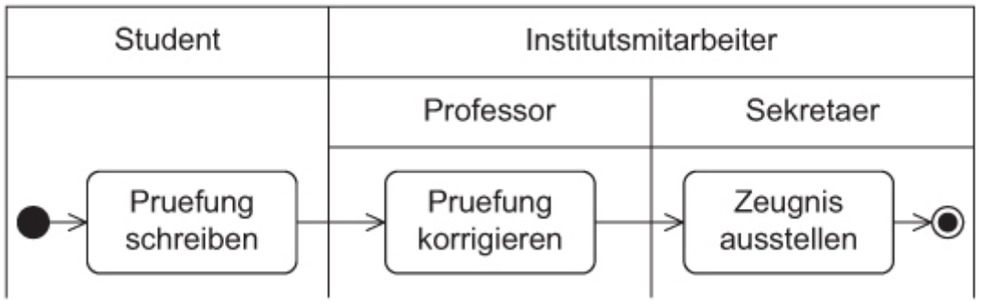
- Erlauben die **Gruppierung von Knoten und Kanten** einer Aktivität nach bestimmten Kriterien
- **Logische Sicht auf eine Aktivität** zur Erhöhung der Übersichtlichkeit und Semantik des Modells
- »Schwimmbahnen«-Notation (partitions, swimlanes)



- Hierarchische Partitionen
  - Zur Schachtelung auf verschiedenen Hierarchieebenen

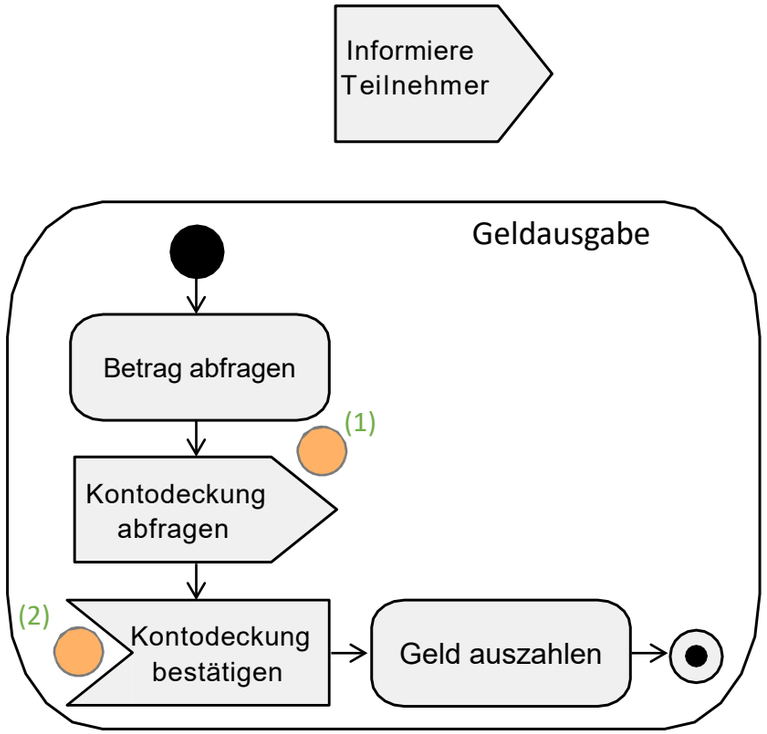


- Beispiel hierarchische Partitionen:



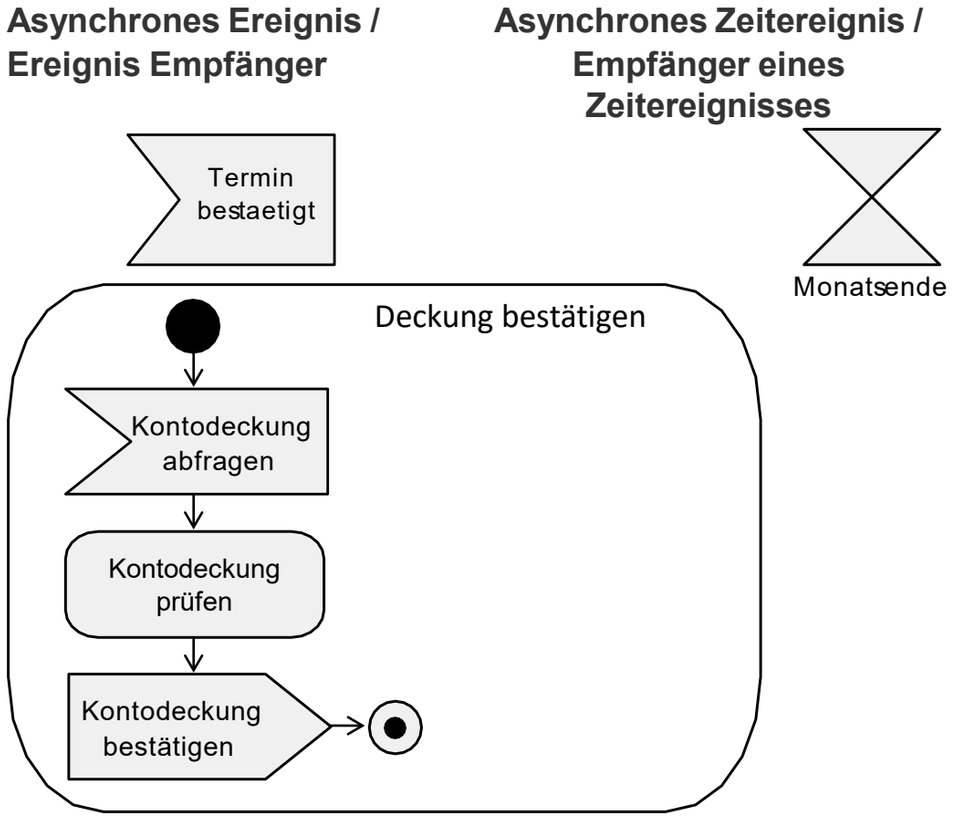
Sonderformen von Aktionen: Signale und Ereignisse

- Senden von Signalen (SendSignalAction)



Nach dem Senden des Signals ist die Aktion (1) zu Ende
Token wird im asynchronen Ereignis (2) aufbewahrt bis ein Signal eingeht

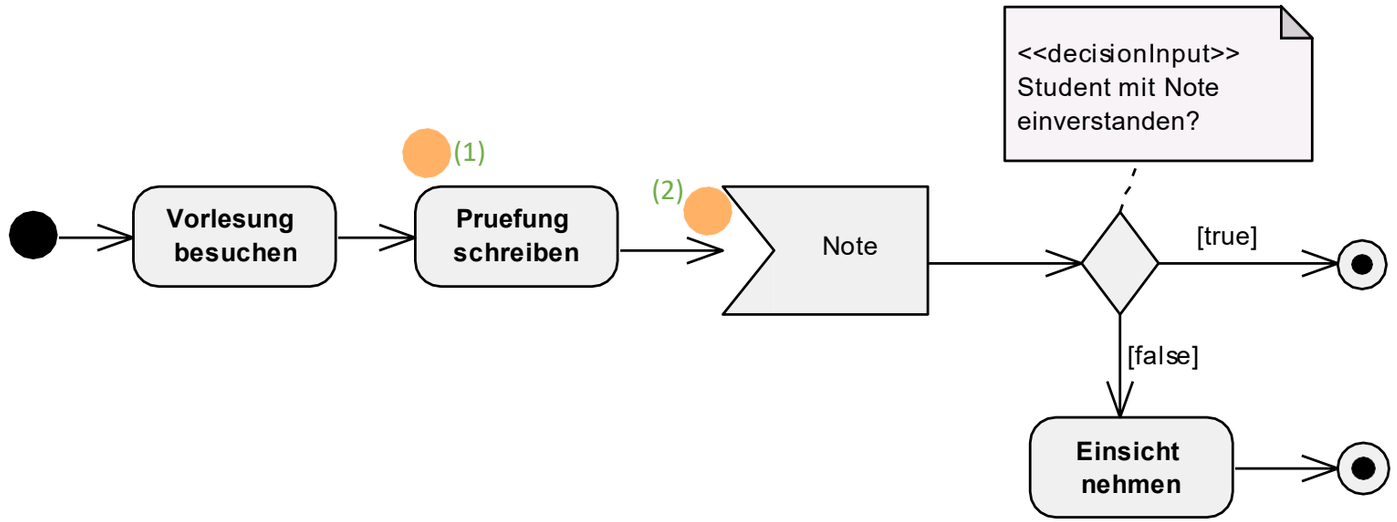
- Empfangen von Ereignissen (AcceptEventAction)



Signale und Ereignisse

Beispiel: Asynchrones Ereignis

- Beschreibung: 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.



Nach dem „Prüfung schreiben“ (1) wartet der Student im Asynchronen Ereignis „Note“ (2)
Keine explizite Aktion „auf Note warten“ dafür notwendig, sonst unklar, wie der Token weiter gereicht wird

Aktivitätsdiagramm

Agenda

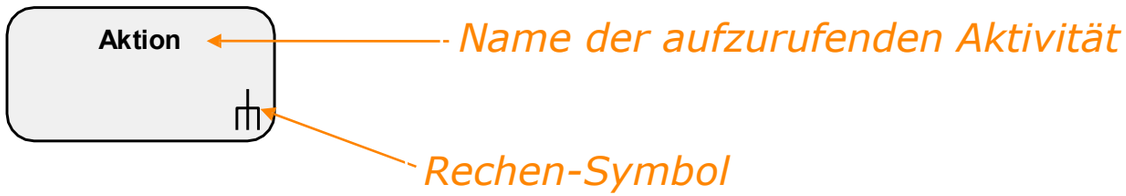
- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- **Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich**
- Beispiel
- Zusammenfassung Elemente / Rückblick

Schachtelung von Aktivitäten

- Aktivitäten können wiederum Aktivitäten aufrufen
- So können Details in eine tiefere Ebene ausgelagert werden

- Vorteile:
  - Bessere Lesbarkeit
  - Wiederverwendung

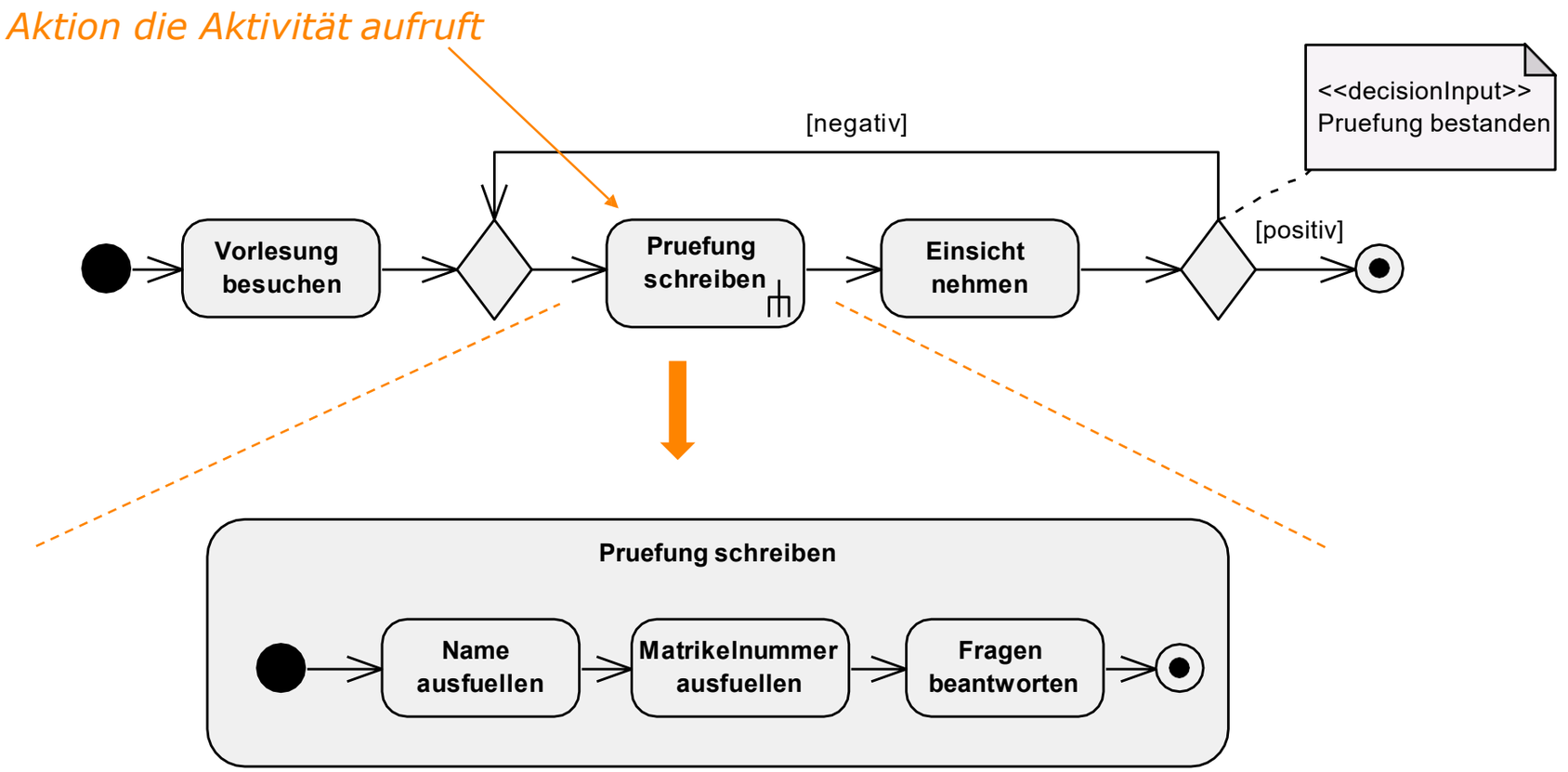
- Notation:
  - In einer Aktion wird eine Aktivität aufgerufen



[Notation in Visual Paradigm Online so nicht enthalten, dafür direkt „Activity“ verwenden]

Schachtelung von Aktivitäten

- Beispiel für die Schachtelung einer Aktivität: Vorlesung bestehen



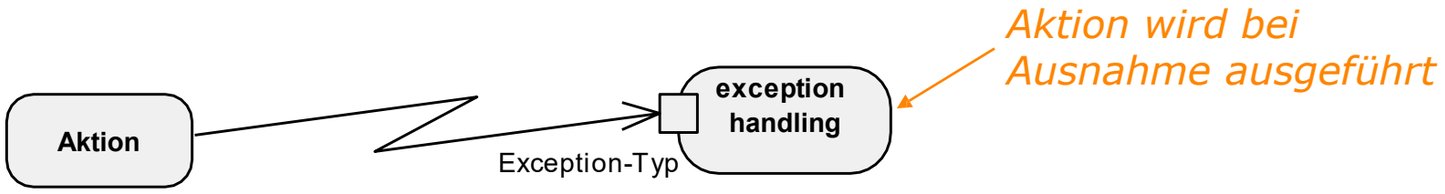
Ausnahmebehandlung – Exception Handler (1/2)

- **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:**

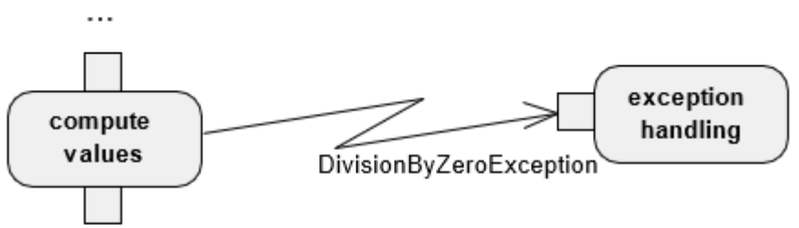


Ausnahmebehandlung – Exception Handler (2/2)

- 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)

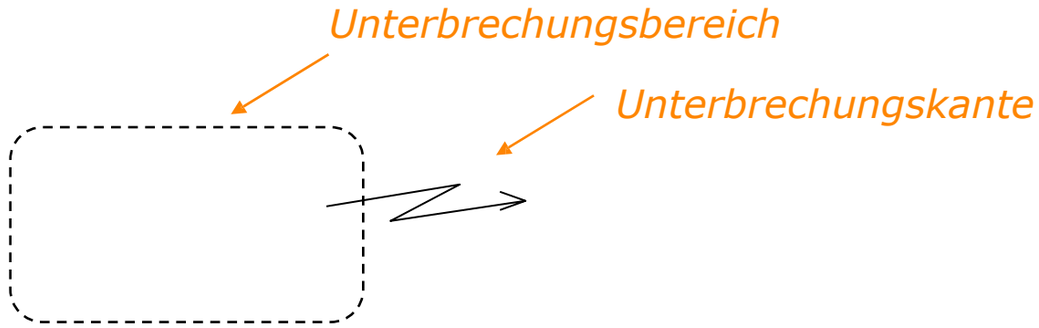
- **Beispiel:**

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



Ausnahmebehandlung – Unterbrechungsbereich (1/2)

- **Umschließt** 1-n Aktionen
- Notation:



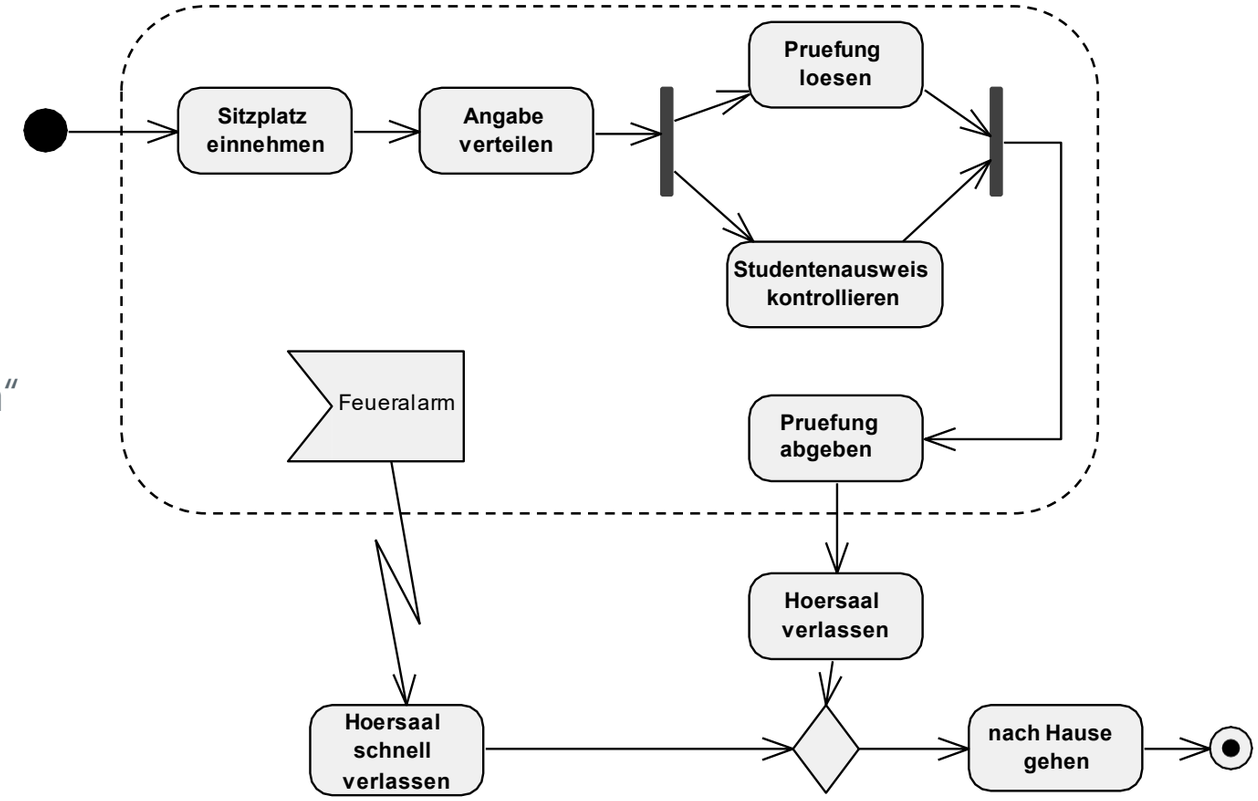
- Wird der Unterbrechungsbereich über die Unterbrechungskante verlassen, so werden alle in der Region vorhandenen Token gelöscht
- Man kann auch nur einen **Teil des Diagramms** im Unterbrechungsbereich haben, es muss nicht immer das ganze Diagramm sein

Ausnahmebehandlung – Unterbrechungsbereich (2/2)

- **Beispiel Unterbrechungsbereich:** Prüfung schreiben

- Modell beschreibt was passiert, wenn beim Schreiben einer Prüfung ein Feueralarm ausbricht

- Da der Feueralarm im Unterbrechungsbereich startet, wird der Token weitergereicht an „Hörsaal schnell verlassen“
- Alle Token im Unterbrechungsbereich werden gelöscht
- Als nächste Aktion wird „nach Hause gehen“ ausgeführt
- Danach ist die Aktivität zu Ende.



Aktivitätsdiagramm

Agenda

- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- **Beispiel**
- Zusammenfassung Elemente / Rückblick

Beispiel: Feier veranstalten – Teil 1

Ziel der Übung: vereinfachte Darstellung der Vorbereitung und Durchführung einer Feier von WI24A3 zum erfolgreichen Bestehen der Klausur zu Systemanalyse und -entwurf als Aktivitätsdiagramm.

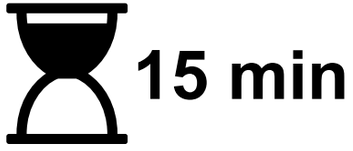
- **Verbale Beschreibung des Beispiels – Teil 1**
  - Zur Durchführung der Feier werden drei Aktivitäten benötigt, die nacheinander durchgeführt werden (Einkaufen, Kochen, Feiern).
  - Modellieren Sie zunächst diese drei Aktivitäten in einem Aktivitätsdiagramm, berücksichtigen Sie dabei auch die Objekte, die zwischen den Aktivitäten weitergereicht werden und denken Sie an die Start- und Endpunkte.
  - Die einzelnen durchzuführenden Aktionen des Einkaufens sind in der Aktivität Einkaufen angeführt. Bei diesen Aktionen handelt es sich darum einen Einkaufszettel zu schreiben, die Waren zu besorgen und die Waren zu bezahlen.
  - Aus der Aktivität Einkaufen fließt ein Objektfluss mit der/den erworbenen Ware(n) in die Aktivität Kochen. Die Aktivität Kochen erhält Ware als Input und reicht Speise weiter.
  - Kochen ist in einem eigenen Diagramm modelliert. Dieses modellieren wir in einem zweiten Schritt.
  - Die Aktivität Feiern erhält Speise von der Aktivität Kochen. Wie die Feier selbst abläuft werden wir nicht modellieren, das überlassen wir Ihrer Fantasie. Aber auch die Feier hat irgendwann ein Ende.

[Beispiel abgeleitet von Steinpichler und Kargl – Projektabwicklung mit UML und Enterprise Architect]

Beispiel: Feier veranstalten – Teil 1

Ziel der Übung: vereinfachte Darstellung der Vorbereitung und Durchführung einer Feier von WI24A3 zum erfolgreichen Bestehen der Klausur zu Systemanalyse und -entwurf als Aktivitätsdiagramm.

- **Vorgehensweise – Teil 1**
  - Erledigen Sie die Aufgabe in Kleingruppen
  - Sie haben 15 Minuten Zeit
  - Gemeinsame Ergebnisdiskussion



Beispiel: Feier veranstalten – Teil 2

- **Verbale Beschreibung des Beispiels – Teil 2**
  - Wir modellieren nun die Aktivität Kochen in einem eigenen Diagramm.
  - Wir betrachten zunächst die Aktionen auf Seite WI24A3 und geben ihnen den entsprechenden Rahmen.
  - Der Prozess des Kochens beginnt mit dem Eingang der Ware. Während gekocht wird (was viele Schritte umfasst), wird zunächst auch ein Lied gesungen.
  - Der Prozess „Ein Liedchen singen“ wird nach dem Singen beendet. Das heißt, wenn die Aktion (Singen des Liedes) beendet ist, „stirbt“ dieser Zweig des Prozesses.
  - Der zweite nebenläufige Prozess (das eigentliche Kochen) beginnt mit dem Vorbereiten der Waren und dem anschließenden Zubereiten der Waren.
  - In die Aktivität Waren zubereiten fließt Ware hinein und Speise wieder hinaus. Hier werden aus Waren Speisen.
  - Ist die Speise ungenießbar, wird sie entsorgt und eine Pizza wird bestellt. Das Bestellen der Pizza ist keine „normale“ Aktion.
  - Bestellung entgegennehmen (auch keine „normale“ Aktion) fängt das Signal auf.
  - Das triggert den Start eines unabhängigen weiteren Prozesses, das Backen der Pizza in der Pizzeria. Auch dieser erhält einen angemessenen Rahmen.

[Beispiel abgeleitet von Steinpichler und Kargl – Projektabwicklung mit UML und Enterprise Architect]

Beispiel: Feier veranstalten – Teil 2

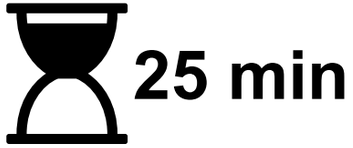
- **Verbale Beschreibung des Beispiels – Teil 2**
  - Nachdem die Pizza von WI24A3 bestellt wurde, wird die Aktion Fernsehen ausgeführt. Dabei wird ein Unterbrechungsbereich betreten. Der Unterbrechungsbereich wird entweder dadurch getriggert, dass der Pizzadienst läutet, oder dass eine Stunde vergangen ist. Die Pizza wird dann entweder bezahlt oder ein Butterbrot gestrichen (wie es hier weiter geht, steht weiter unten).
  - Im Prozess der Pizzeria findet die Aktivität Pizza backen statt, die einzelnen Schritte davon bleiben allerdings ein Geheimnis der Pizzeria. Anschließend liefert die Pizzeria die Pizza aus, ihr Prozess ist danach beendet.
  - Zurück zu WI24A3: Befinden wir uns im Unterbrechungsbereich und das Signal des Pizza ausliefern trifft ein, verlassen wir den Unterbrechungsbereich und fernsehen wird unterbrochen. Im nächsten Schritt wird die Pizza bezahlt. Alternativ wird der Unterbrechungsbereich verlassen, wenn die Pizza nicht innerhalb von einer Stunde eintrifft, dann läuft der Timer ab, der Unterbrechungsbereich wird verlassen und ein Butterbrot wird als Alternative gestrichen. Anschließend wird der Prozessfluss wieder zusammengeführt.
  - Nachdem die Speise auf den Teller gelegt wurde, ist die Aktivität Kochen beendet. Die Speise wird als Ergebnis der Aktivität übergeben.

[Beispiel abgeleitet von Steinpichler und Kargl – Projektabwicklung mit UML und Enterprise Architect]

Beispiel: Feier veranstalten – Teil 2

- **Ziel der Übung:** vereinfachte Darstellung der Vorbereitung und Durchführung einer Feier von WI24A3 zum erfolgreichen Bestehen der Klausur zu Systemanalyse und -entwurf als Aktivitätsdiagramm.

- **Vorgehensweise – Teil 2**
  - Erledigen Sie die Aufgabe in Kleingruppen
  - Sie haben 25 Minuten Zeit
  - Gemeinsame Ergebnisdiskussion



Aktivitätsdiagramm

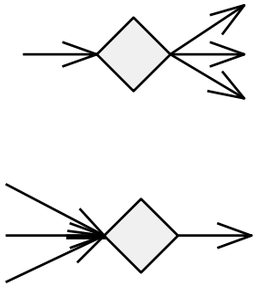
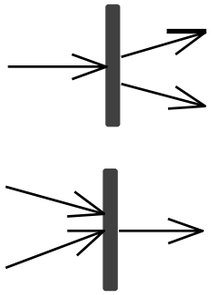
Agenda

- Einführung
- Aktivitäten, Aktionen und Kanten
- Initialknoten, Aktivitätsendknoten, Ablaufendknoten
- Alternative Abläufe und parallele Abläufe
- Token
- Objektknoten und Objektfluss
- Partitionen, Signale und Ereignisse
- Schachtelung von Aktivitäten, Ausnahmebehandlung und Unterbrechungsbereich
- Beispiel
- **Zusammenfassung Elemente / Rückblick**

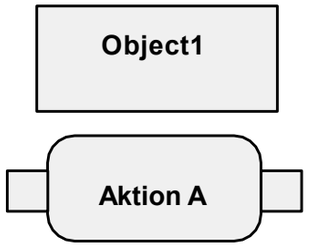
Aktivitätsdiagramm – Elemente (1/5)

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

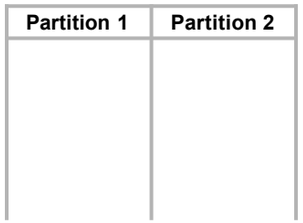
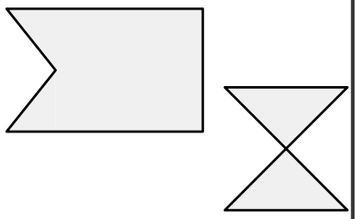
Aktivitätsdiagramm – Elemente (2/5)

Name	Syntax	Beschreibung
Entscheidungs-/ Vereinigungs-knoten		Aufspaltung/Zusammenführung von alternativen Abläufen
Parallelisierungs-/ Synchronisations-knoten		Aufspaltung eines Ablaufs in nebenläufige Abläufe / Zusammen-führung von nebenläufigen Abläufen in einen Ablauf

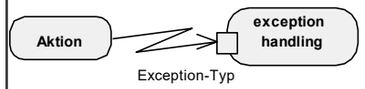
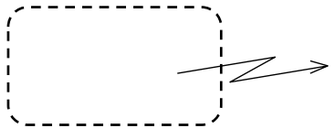
Aktivitätsdiagramm – Elemente (3/5)

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

Aktivitätsdiagramm – Elemente (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

Aktivitätsdiagramm – Elemente (5/5)

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