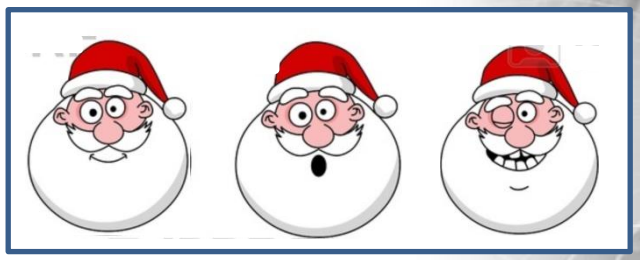
ModSoft

Modellbasierte Software-Entwicklung mit UML 2
im WS 2014/15

Teil III b: Verhaltensmodellierung: Aktivitäten



Prof. Dr. Joachim Fischer
Dr. Markus Scheidgen
Dipl.-Inf. Andreas Blunk

fischer@informatik.hu-berlin.de

Inhalt

Teil I- Einführung

Teil II-Struktur

a)

- Klassen, Assoziationen, Abhängigkeiten
- Interface, Datentypen, Signale, Port,
- Strukturierter Classifier
- Aktive Klassen

b)

- Präzisierungen
- Klassendiagramm, vertiefende Betrachtung
- Operationen

Teil III-Verhalten

a)

- Einfacher Zustandsautomat
- Beispiel: DemonGame
- UML-Modellierungsdefizite

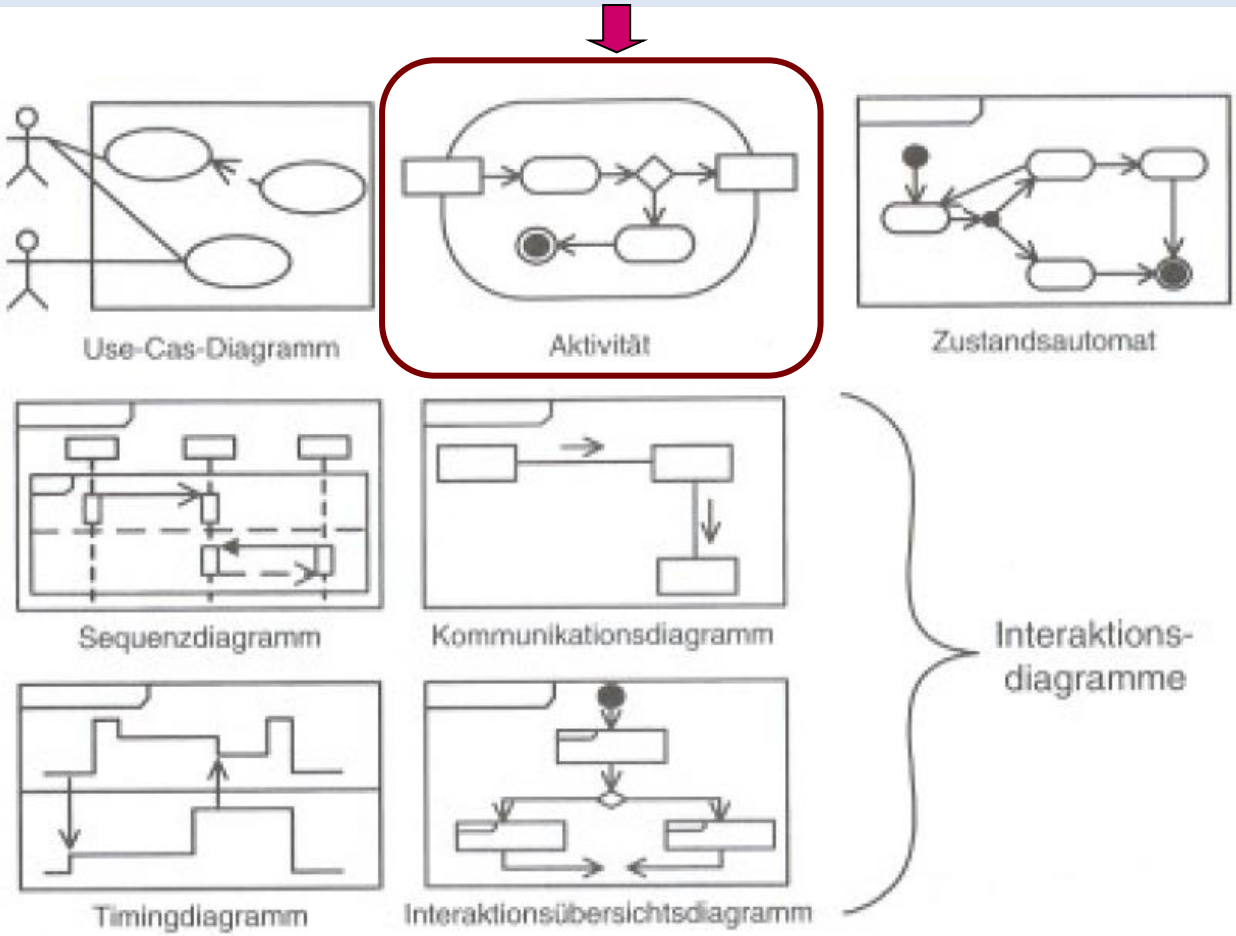
b)

- Aktivitäten

Aktivitätsdiagramme

1. Allgemeine Charakterisierung
2. Kontrollflussknoten: Typen
3. Objektknoten: Typen
4. Objektfluss-Steuerung
5. Strukturierte Aktivitätsknoten
6. Aktionen

Verhaltensdiagramme in UML



- stellen Verhaltensspezifikationen aus unterschiedlichen Blickwinkeln dar
- betonen oder vernachlässigen dabei bestimmte Aspekte
- ergeben in ihrer Kombination eine mehr oder weniger vollständige Beschreibung des Gesamtverhaltens

Allgemeine Charakterisierung

7 Aspekte

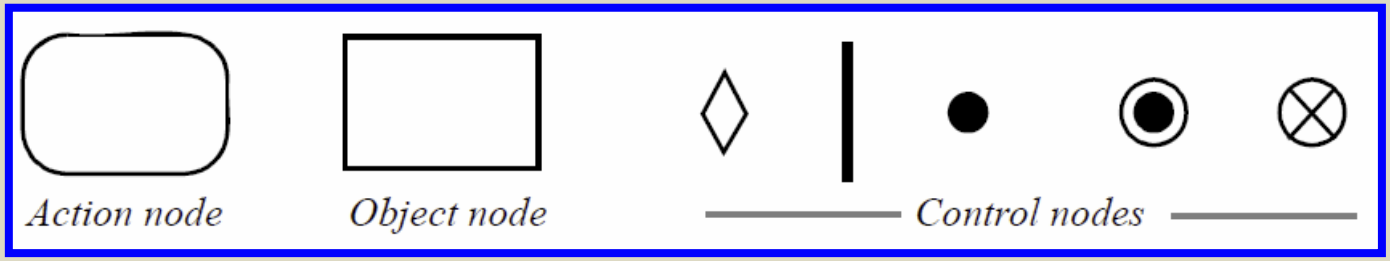
(1) Verhaltensbeschreibung

Aktivitäten stellen eine konkrete Spielart der Verhaltensbeschreibung von UML dar

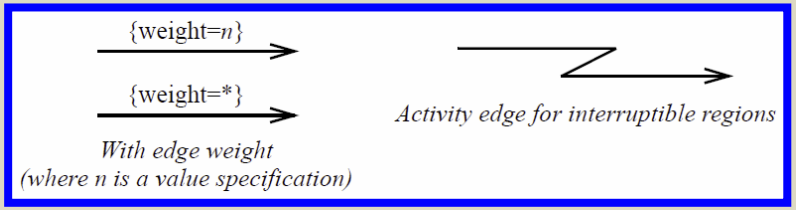
Eine **Aktivität** (als parametrisierbare Verhaltensbeschreibung) ist ein gerichteter **Graph,**
bestehend aus **Knoten und Kanten** (Kontroll- und Datenfluss),
der bei seiner Ausführung schrittweise durchlaufen wird

(2) Strukturell besteht eine Aktivität aus

- Start- und Endknoten,
- einer Reihe von Aktions-, Kontroll- und Objektknoten

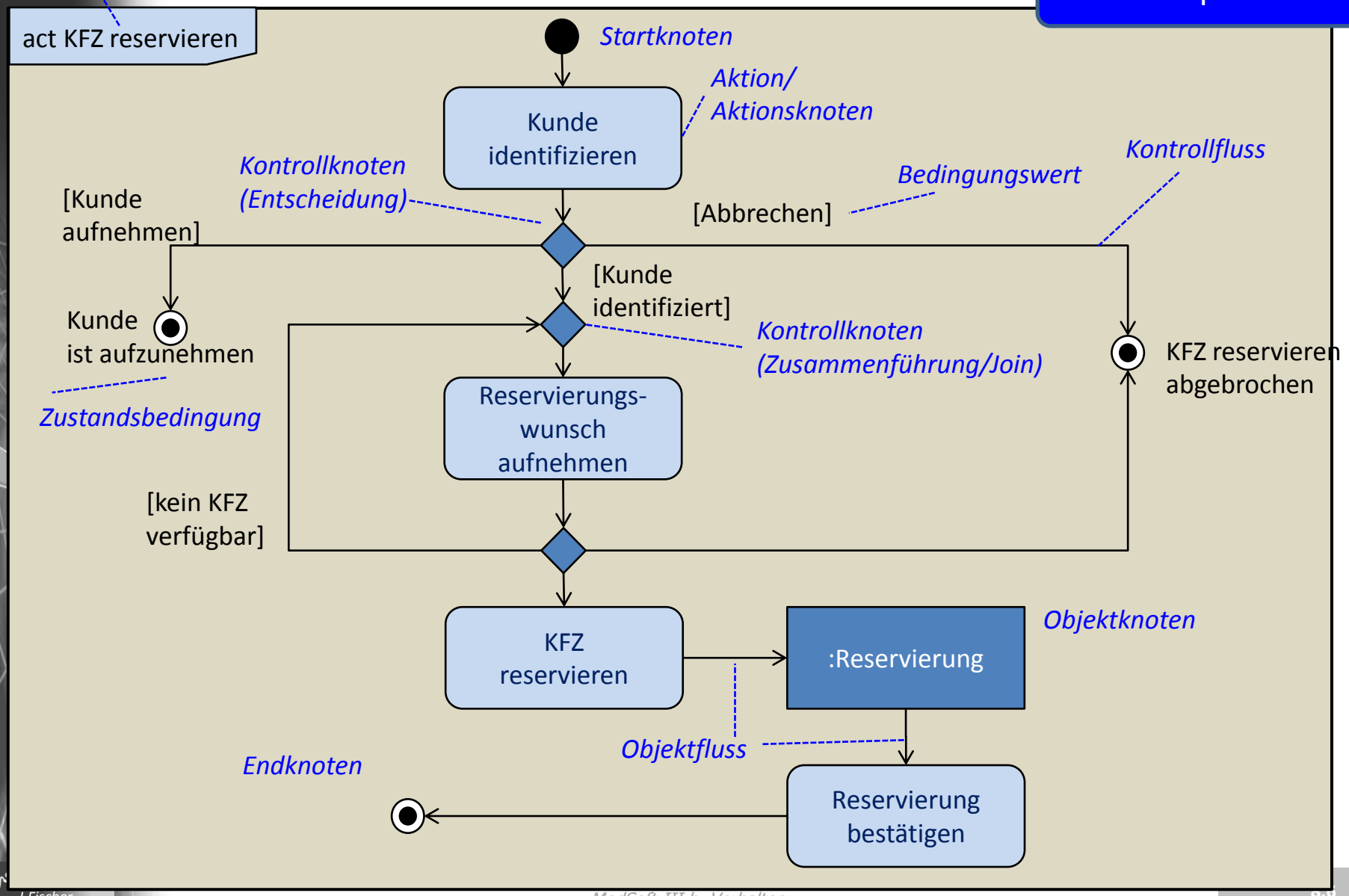


- sowie Objekt- und Kontrollflüssen, mit denen diese Knoten verbunden sind



Konzepte am einfachen Beispiel

7 Aspekte



Allgemeine Charakterisierung (Forts.1)

7 Aspekte

(3) Kontext einer Aktivität ~ Classifier-Zuordnung

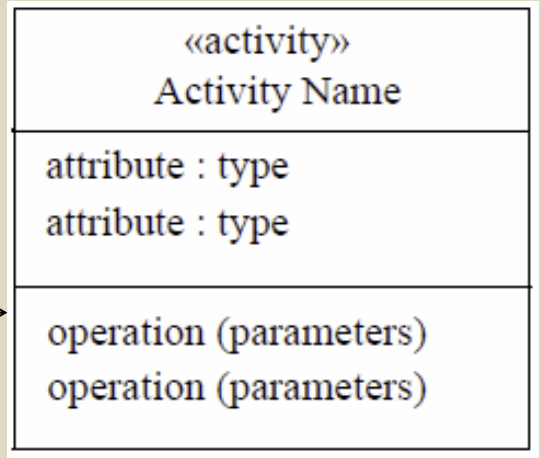
OO-Paradigma von UML (Einheit von Struktur und Verhalten)
~ regelt Zugriff auf Operationen und Attribute des Kontextes,
bereits allgemein in der Oberklasse von Activity (Behavior) definiert

Zuordnungsvarianten des Kontextes

– **indirekt**:
Aktivität stellt method dar (Verhaltensbeschreibung) einer Operation eines Classifiers

– **direkt**: Context-Classifier ist eine aktive Klasse
- Ausführung mit Instanziierung des Kontextes verbunden,
- Löschen des Classierfier-Objekt während der Ausführung führt zum Abbruch der Aktivität
- Ende der Aktivität führt zur Terminierung des Classifieres

– zur Unterstützung nicht-oo-Anwendungen sind auch **autonome Aktivitäten** zulässig,
- jedoch haben auch diese einen Kontext



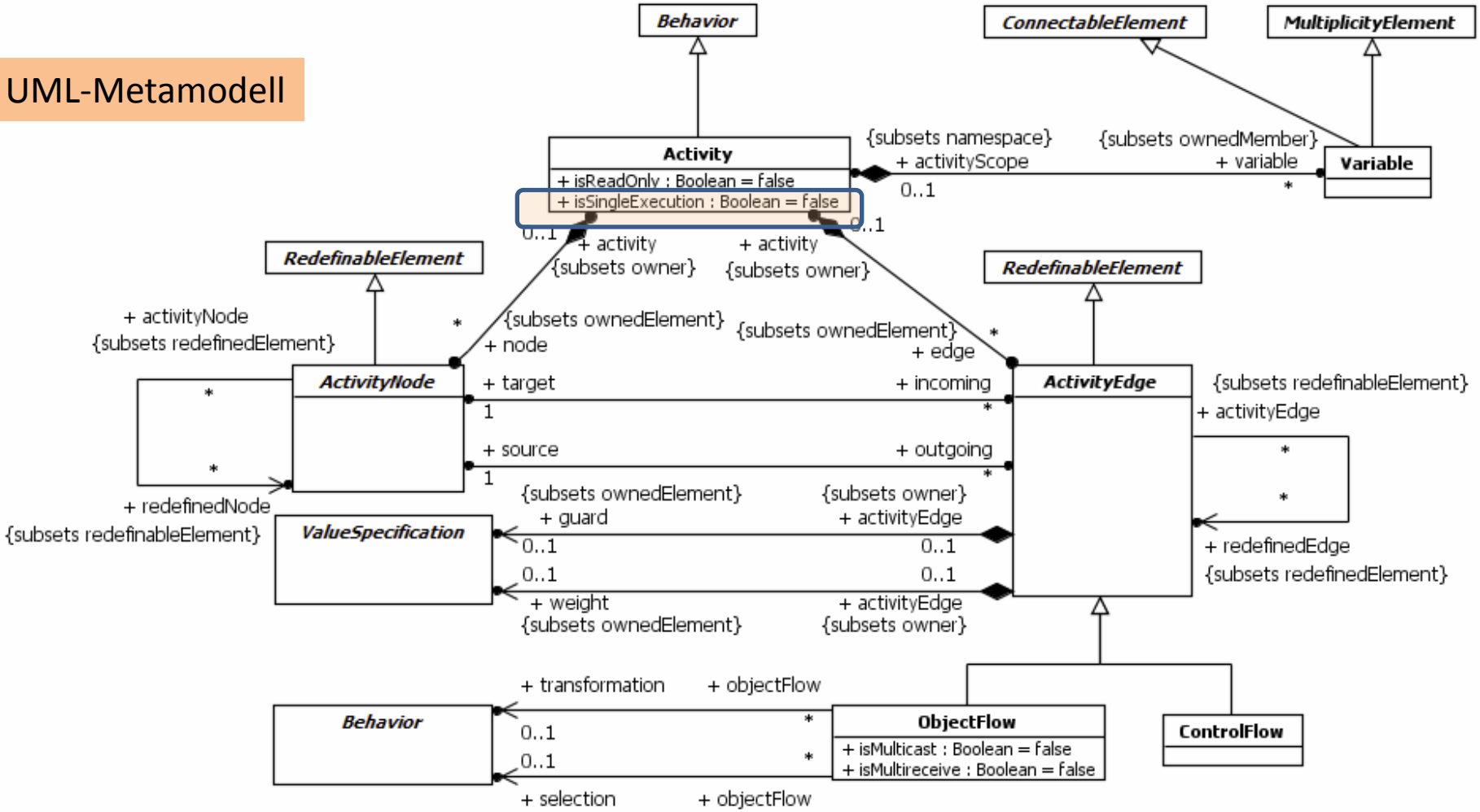
Allgemeine Charakterisierung (Forts.2)

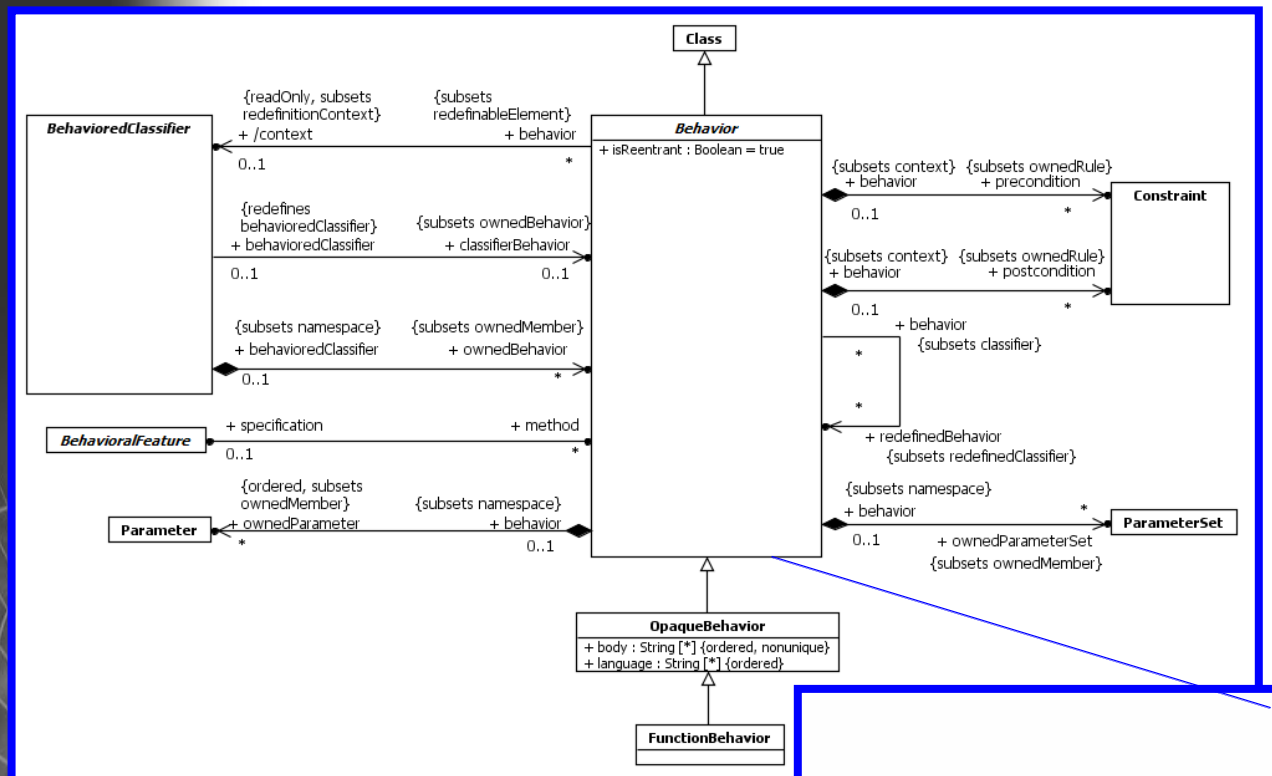
7 Aspekte

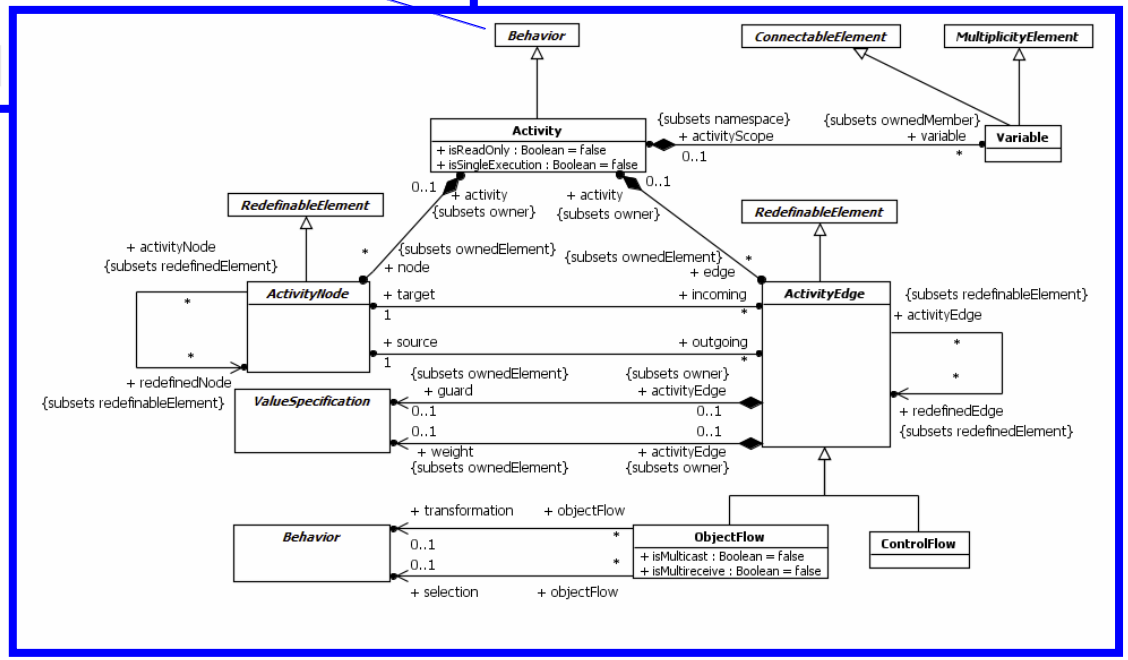
(4) Namensraum

Aktivität stellt einen Namensraum dar, der eingeschränkte Sichtbarkeit der Bestandteile sichert

UML-Metamodell







Activity verfügt über:

- Context
- Pre- und Post-Conditions
- Parameter, Parametermengen
- …

Allgemeine Charakterisierung (Forts.3)

7 Aspekte

(5) Aspekte der Verhaltensrealisierung

a) Zeitdauer

die Ausführung einer Aktivität benötigt/verbraucht Zeit

spannende Frage: kann man das quantifizieren?

b) Ausführungsvarianten

Zeitlich überlagerte Ausführungen von ein und derselben Aktivität sind unterschiedlich vorstellbar:

1. Aktivitätsausführungen operieren über verschiedene Instanzen des Kontext-Classifieres
   ➔ die Instanzen des Kontextclassifiers agieren zeitlich parallel und **konfliktfrei**
   isSingleExecution= false

2. Aktivitätsausführungen operieren über ein und die gleiche Kontext-Classifier-Instanz
   ➔ die Abläufe im Verhalten eines Kontextclassifiers-Objektes agieren parallel

   verschiedene Modelle vorstellbar:

   – ~ SLX Parent- und Child-Puck: hier ist gesichert, dass zu einem Zeitpunkt nur eine der Verhaltensinstanzen die Steuerung besitzt →**konfliktfrei**

   – echt parallel:

     ~ massiv-parallele Systeme, konfliktfrei

     ~ i.allg. aber konfliktbehaftet

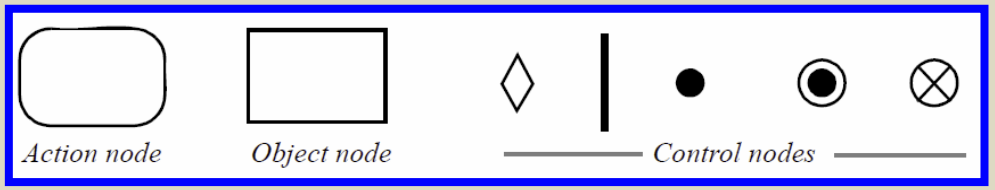
     isSingleExecution= true

Allgemeine Charakterisierung (Forts.4)

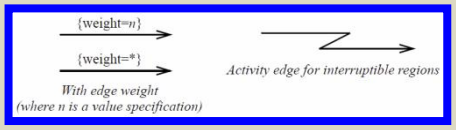
7 Aspekte

(5) Knoten und Kanten (detaillierter)

- **Aktivitätsknoten** =
  a) Aktionsknoten
  b) Objektknoten
  c) Kontrollknoten



- ad a) **Aktionsknoten**: ~ Ausführung in **endlicher Zeit**
  - arithmetische, logische Berechnung
  - Operationsrufe, Objekt-Manipulationen
  - Kontrollfluss-Manipulation: Synchronization, Entscheidungen, Parallelisierung
- ad b) **Kontrollknoten** (ohne Verhalten) **zeitlos**,
  dennoch können sich bei einigen Koten (Ausnahmen) Verweildauern von KontrollToken ergeben
- ad c) **Objektknoten**
  Verweildauer von Objekten ~ **endliche Zeit** (Betrachtungszeitraum)
- **Flusskanten**: Weiterleitung von Token **zeitlos** (drücken aber ein **Vorher/Nacher** aus)



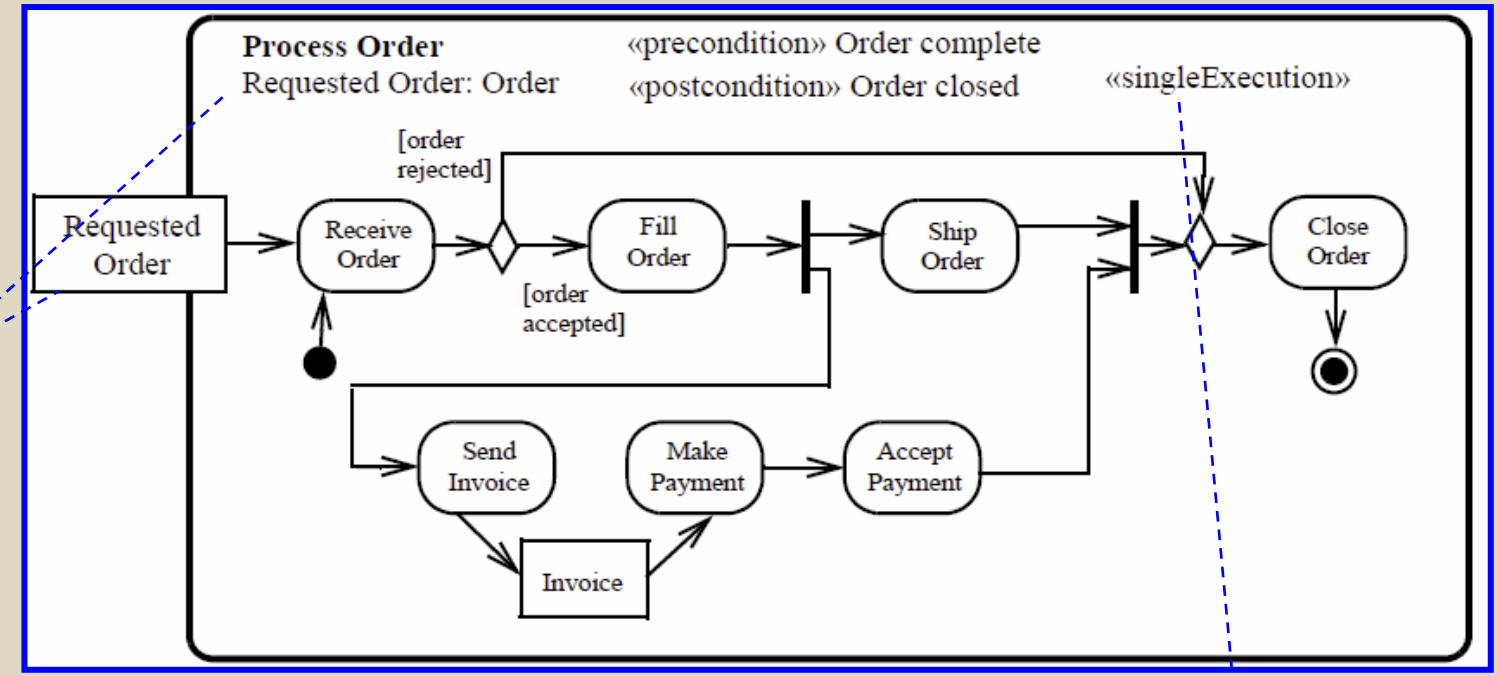
Beschriftungen sind möglich

Allgemeine Charakterisierung (Forts.5)

7 Aspekte

(6) Parameter und Ausführungsbedingungen

- Ein- und ausgehende Objekte einer Aktivität werden als **Parameter** der Aktivität bezeichnet (Pins einer Aktivität)

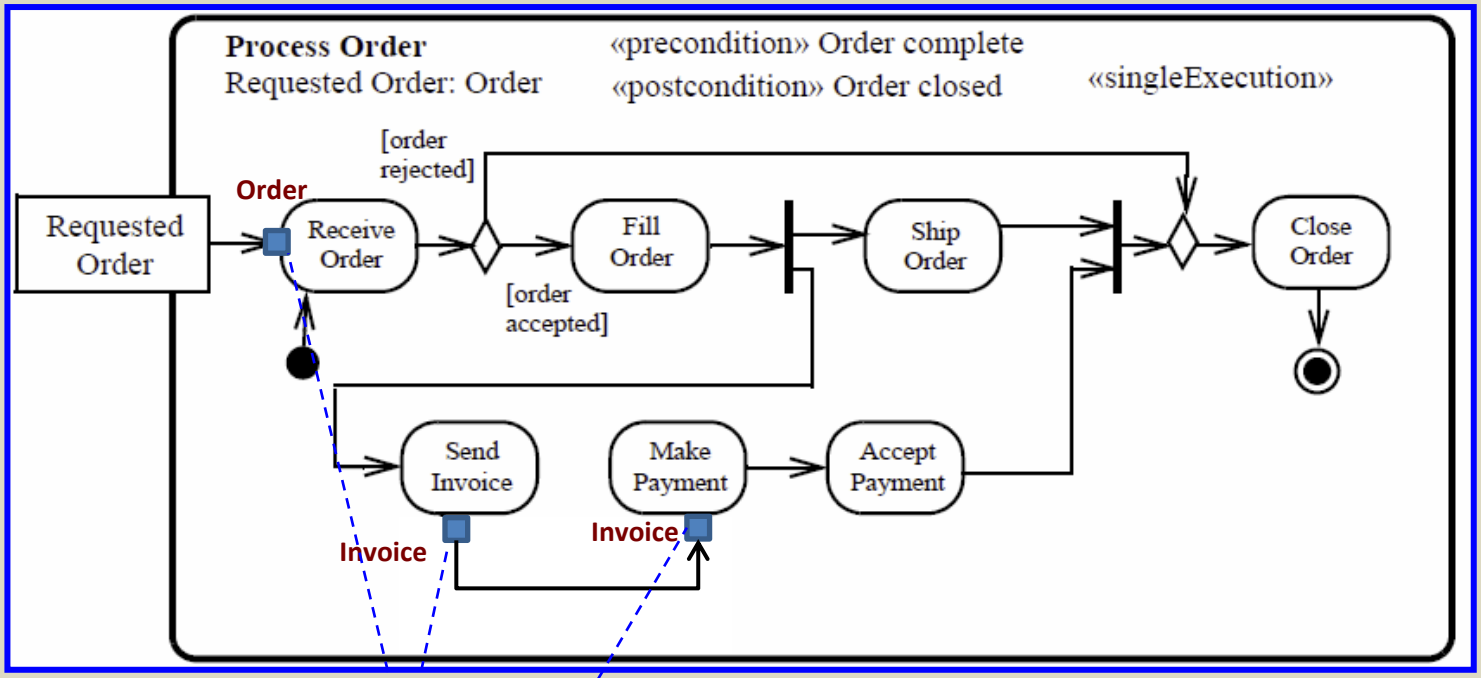


- Optionale Angabe von **Ausführungsbedingungen** (Vor-, Nachbedingung, Ausführungsvariante)
- Hier: Jeder **Aufruf** von Process Order setzt die Bereitstellung eines Order-Objektes als **aktuellen Parameter** voraus
- Beispiel-Diagramm aus dem UML-Standard kommt mit impliziten Ersetzungen daher

Allgemeine Charakterisierung (Forts.6)

7 Aspekte

- Implizite Ersetzung: weitere syntaktische Variante

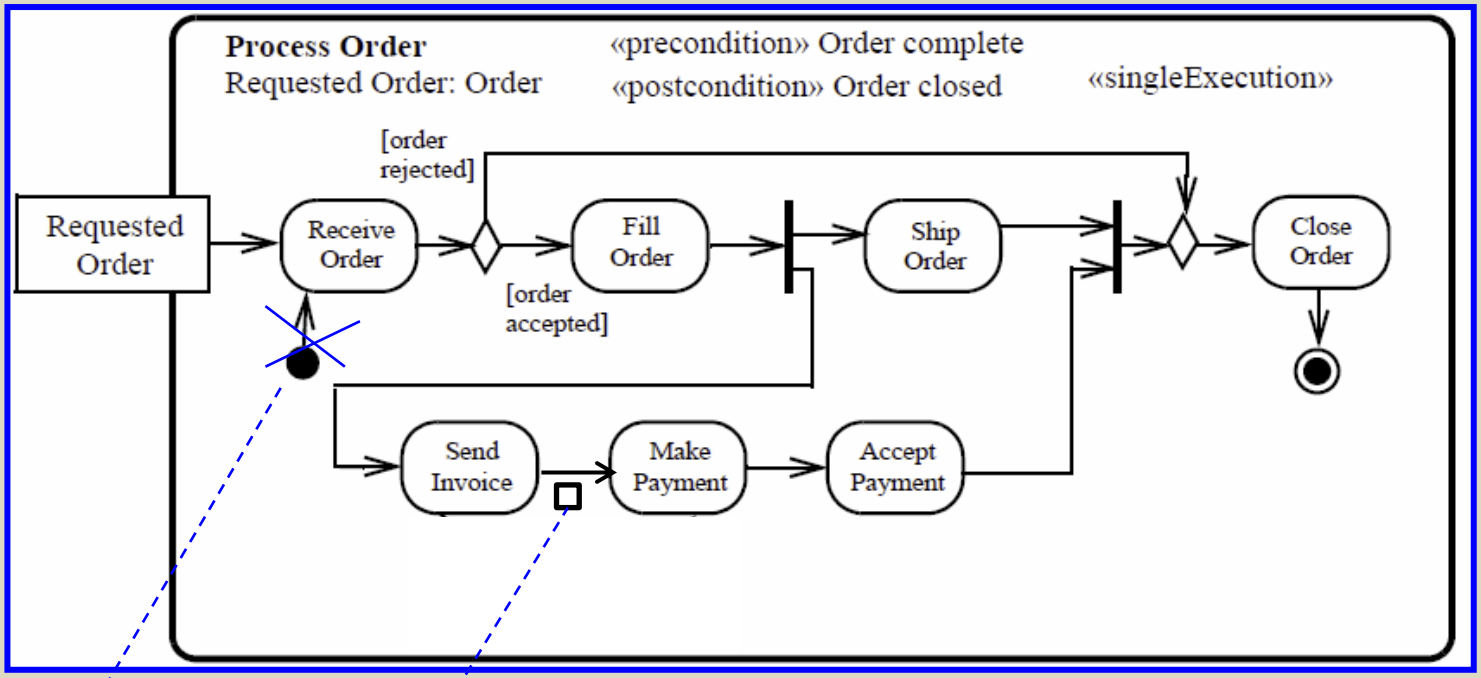


Ersetzungen, die noch weiter abstrahiert werden können

Allgemeine Charakterisierung (Forts.7)

7 Aspekte

- abstrakte syntaktische Variante



Objektflusskante:= Pfeil mit MiniQuadrat ohne Textzusatz

Verzicht auf Startknoten wäre hier auch möglich (Erklärung später)

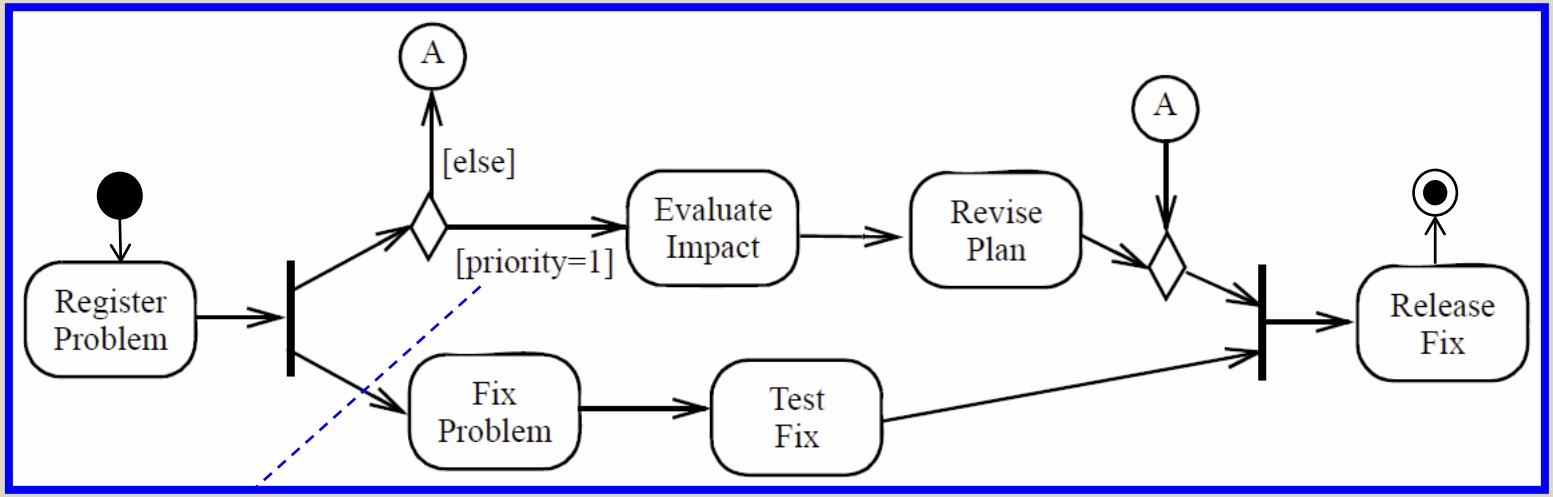
Allgemeine Charakterisierung (Forts.8)

7 Aspekte

(7) Darstellungskraft

- **Komplexe Abläufe** einer Aktivität können mit
  - zahlreichen Ausnahmen, Varianten, Zyklen übersichtlicher und verständlicher dargestellt werden,
  - als mit reinen Text-Darstellungen

- Einsatz von **Konnektoren** erlaubt dabei die Vermeidung sich kreuzender Flußlinien

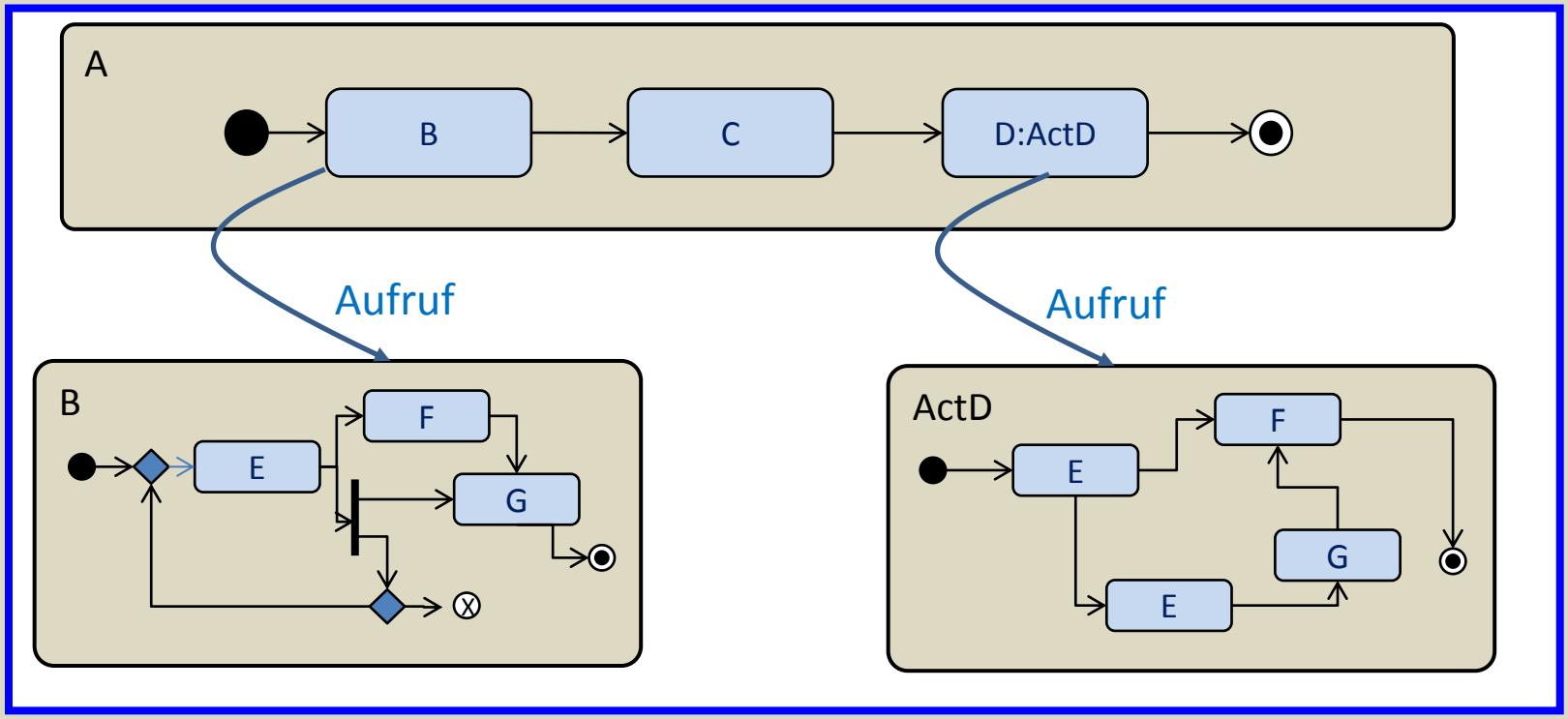


Priority muss im Namensraum der Aktivität (Context) definiert sein

Allgemeine Charakterisierung (Forts.9)

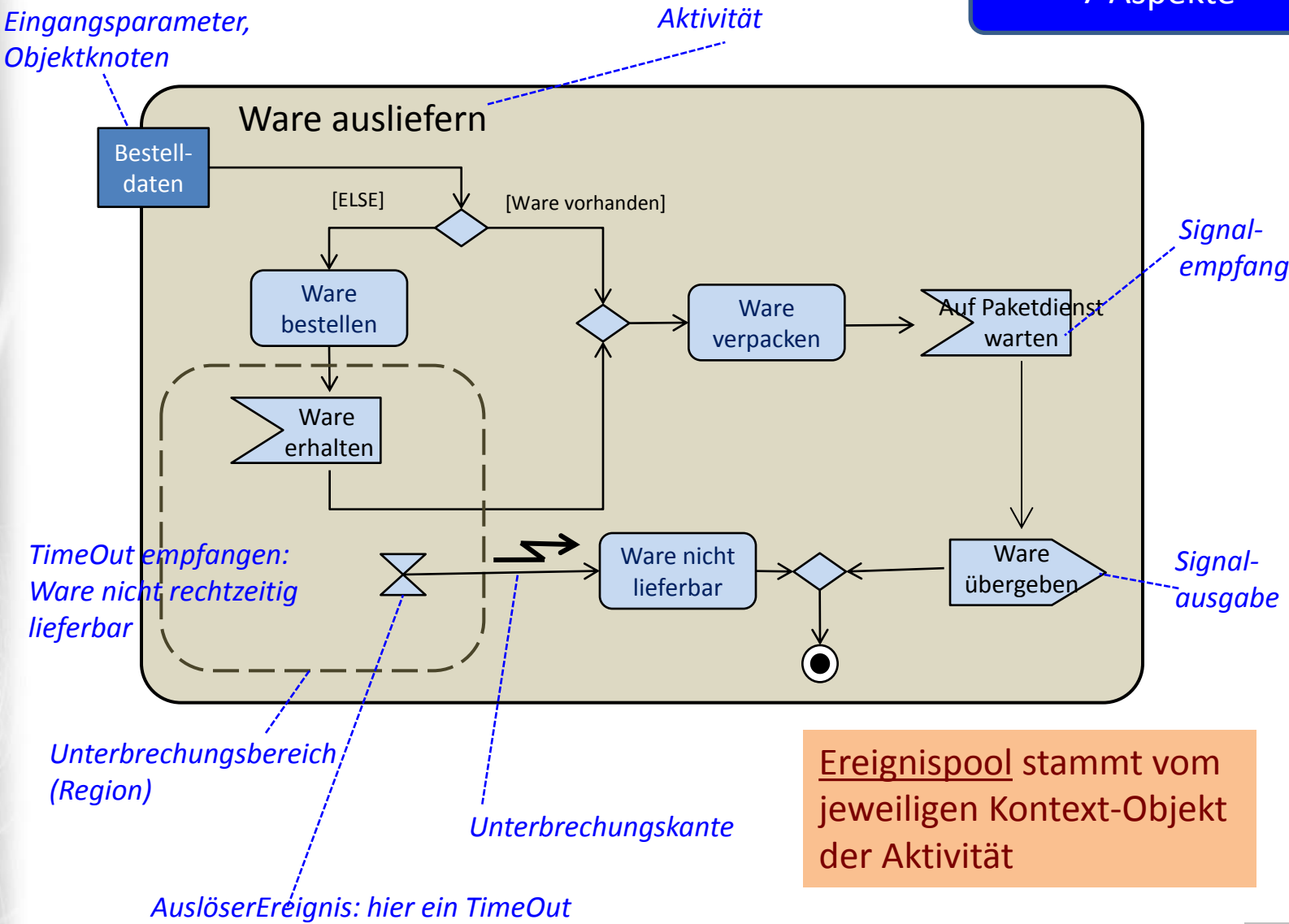
7 Aspekte

- Elementarer Schritt: **Aktion**
- **hierarchische Kompositionen**:
  ein Aktionsknoten kann eine Aktivität aufrufen
  (ähnlich zu: Funktion – Anweisung)



Beispiel: komplexere Aktivität (Forts.10)

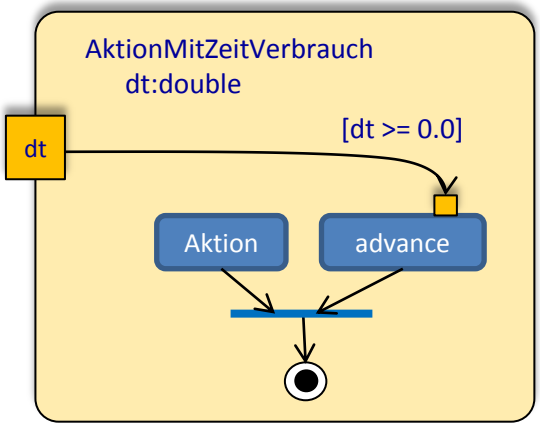
7 Aspekte

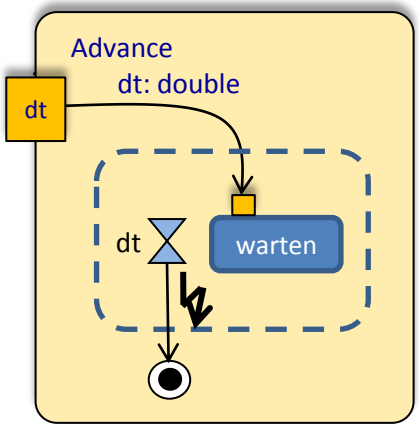


Ereignispool stammt vom jeweiligen Kontext-Objekt der Aktivität

Vorbereitung für ein komplettes Beispiel

- Quantifizierung von Zeitdauern für Aktionen/Aktivitäten

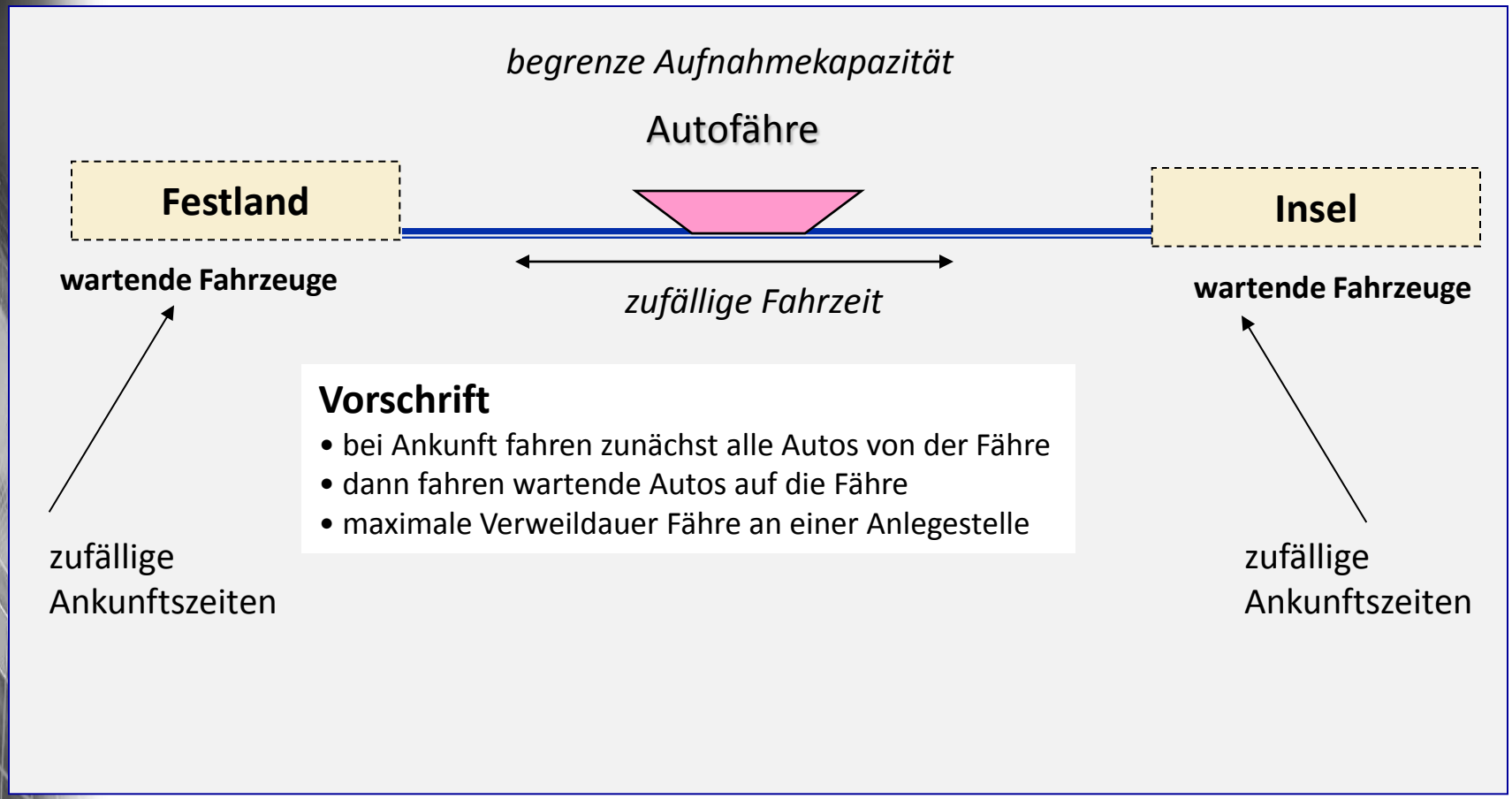




mit Hilfe des Konzeptes Parametrisierter Classifier als Schablone für beliebige (aber sehr elementare Aktivitäten)

Beispiel: Autofähre

- Ziel: Darstellung als Aktivitätsmodell in UML



Aktivitätsdiagramme

1. Allgemeine Charakterisierung
2. Kontrollflussknoten: Typen
   - Initialknoten, Aktivitätsendknoten, Ablaufendknoten, Entscheidungsknoten
   - Vereinigungsknoten, Parallelisierungsknoten, Synchronisationsknoten
3. Objektknoten: Typen
4. Objektfluss-Steuerung
5. Strukturierte Aktivitätsknoten
6. Aktionen

Start und Ende von Aktivitäten

- **Vorbedingung**: Im Allg. müssen an **allen eingehenden Kanten** eines Knotens die erforderlichen Kontroll- u. DatenToken “angeboten” werden
- **Ausführungsbeginn**: Sobald sie allesamt bereit stehen (Synchronisation), “fließen” sie in den Knoten ein, womit dieser zur Ausführung kommt.
- **Verbleib**: Die Token verbleiben in dem Knoten, solange dessen Ausführung dauert **plus der Zeit**, die evtl. benötigt wird, bis der **nächstfolgende Knoten** die Token akzeptiert.
- **Verzögerung**: Eine Überwachungsbedingung kann z.B. eine Weitergabe der Token verhindern. Damit kann es zur Ansammlung von Token im Vorgängerknoten kommen
- in die Ausführung können Aktionen einbezogen werden, die auf den Eintritt spezifischer **Ereignisse** warten
- **Aufenthaltsort**: Token können nur in Aktions- oder Objektknoten verweilen
- **Kontrollknoten** geben die Token immer sofort weiter (bis auf Ausnahmen)
- Start und Ende von Aktivitäten werden zum einen durch **Initialknoten** und zum anderen durch **Aktivitätsend-** und **Ablaufendknoten** festgelegt

Initialknoten: ●

- Sobald eine Aktivität aufgerufen wird (alle Parameter liegen vor), werden an **allen ausgehenden Kanten eines Initialknotens** KontrollToken zur Verfügung gestellt, womit die Aktivität gestartet wird

semantisch äquivalent

- Eine Aktivität kann **mehrere Startknoten** besitzen, womit nebenläufige Abläufe (synchron) gestartet werden

- Eine Aktivität muss aber **nicht unbedingt** einen Initialknoten besitzen.
  1. Ersatz:
     Parameter-Token können den Start auslösen.
  2. Ersatz: **alle** Knoten, die über **keine Eingangskanten** verfügen, werden mit Kontrollknoten versorgt.

Unterschied zum expliziten Einsatz von Initialknoten:
keine Angabe von Überwachungsbedingungen möglich

Aktivitätsendknoten: ◉

- … erzwingt das **Ende aller** Abläufe einer Aktivität

- **Sonderfall:** hat eine Aktivität mehrere Endknoten,
  beendet derjenige Ablauf das Ende der Aktivität,
  dessen KontrollToken einen Aktivitätsendknoten als erster erreicht

- **Ende einer Aktivität bedeutet**,
  a) Nichtweiterausführung **aller** laufenden Aktionen (Atomarität von Aktionen).
  b) Kein Start momentan startbarer Aktionen
  c) Löschen **aller** KontrollToken.
  d) Löschung **aller** DatenToken, die bereits als Ausgabeparameter vorliegen mit anschließender Belegung der Parameter mit NULL-DatenToken
  e) evtl. Rückgabe von DatenToken an Aufrufer der Aktivität (???)

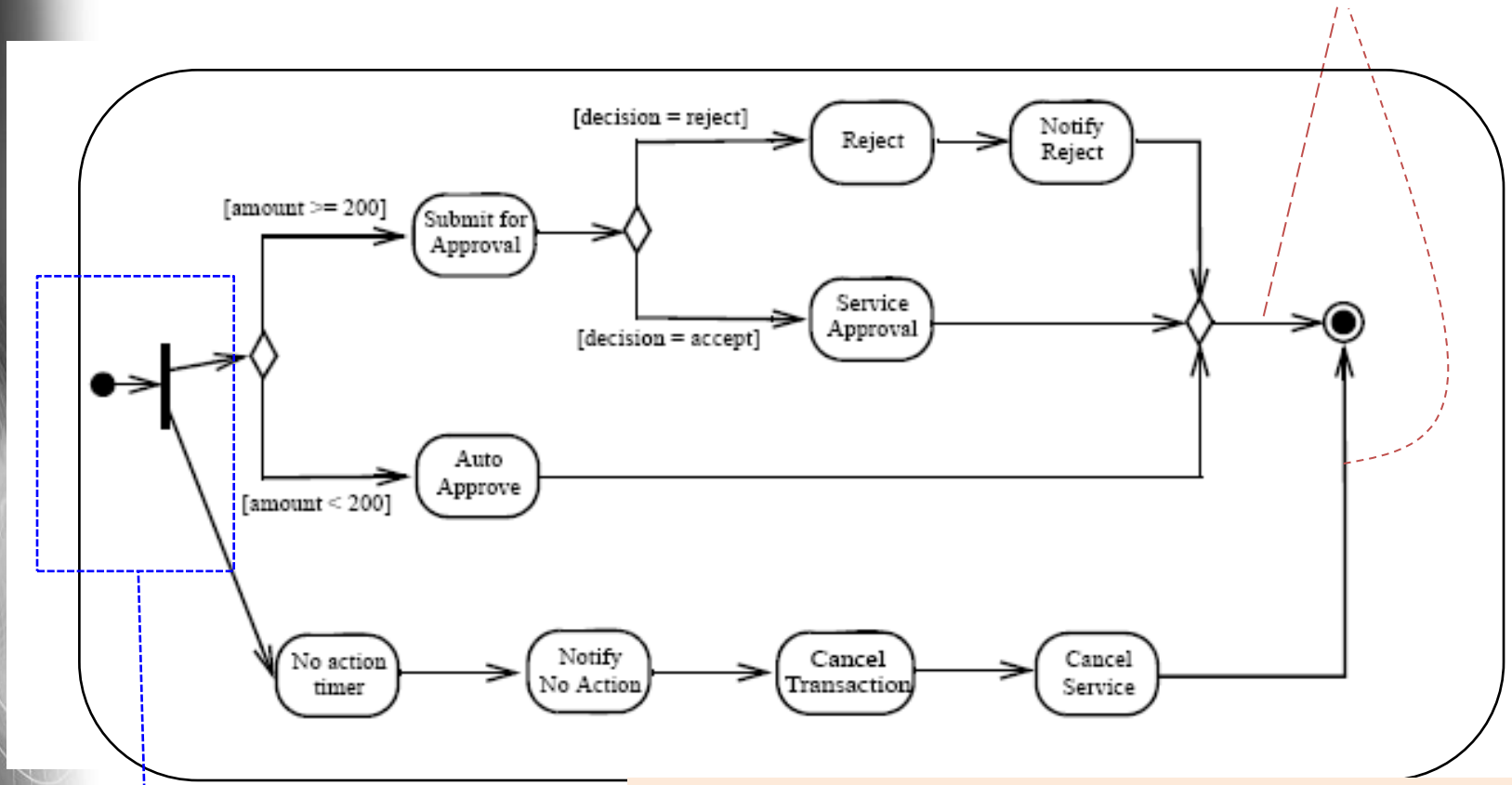
- Repräsentiert die Aktivität den **Lebenszyklus** eines Objektes
  (als Top-Level-Aktivität, die der Instanz des Kontext-Classifiers zugeordnet ist),
  so wird auch das Kontext-Objekt gelöscht.

- Wurde die Aktivität als -**singleExecution**- spezifiziert,
  werden auch hier **alle** gestarteteten Abläufe durch das Erreichen eines Aktivitätsendknoten beendet

Aktivitätsendknoten (Beispiel)

Häufiger Fehler in UML-Büchern:

Annahme einer impliziten Synchronisation nebenläufiger Ausführungen



vielmehr gilt:

Das erste ankommende KontrollToken bricht alles ab

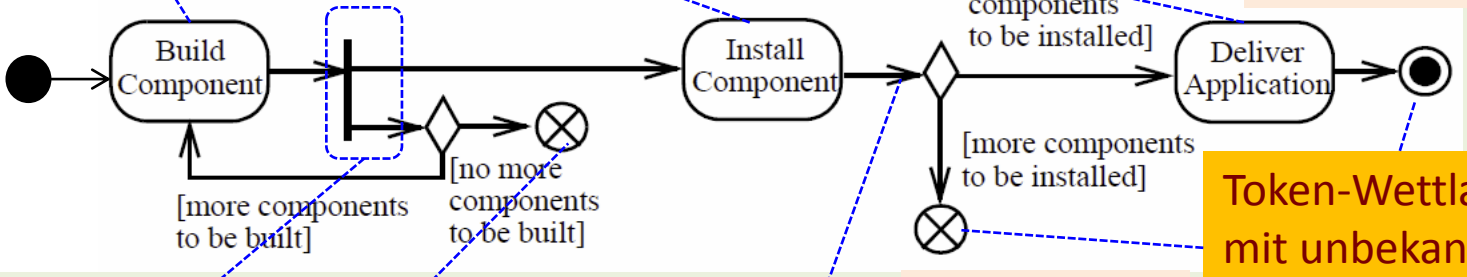
unterschiedliche Darstellungen möglich:

- *Startknoten mit zwei Ausgangskanten,*
- *zwei verschiedene Startknoten*

Ablaufendknoten: ⊗

unspezifizierter Zeitverbrauch,
auch Realationen untereinander nicht bekannt

- … erzwingt das **Ende** nur **eines Ablaufes** von mehreren einer Aktivität



wird 1x passiert

Token-Wettlauf mit unbekanntem Ausgang

wird 0 bis 50x passiert

jeder eingehende KontrollToken erzeugt zwei KontrollToken am Ausgang

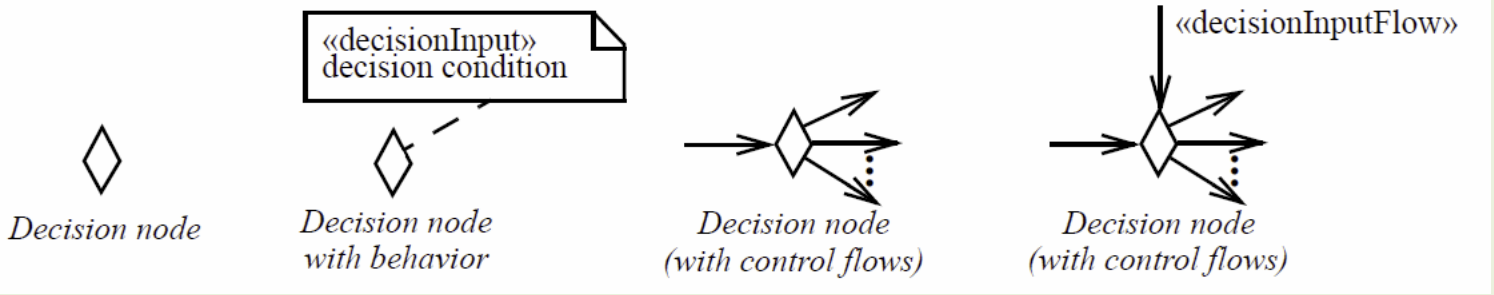
Das (zuletzt duplizierte) 100. KontrollToken erreicht Ablaufendknoten
➔
Es kommt zu keinem weiteren Aufruf von von BuildComponent

Ann.:
*von 100 Token ~ Komponenten, die **erstellt** worden sind, sind 50 durch InstallComponent Installiert (und mehr sollen nicht Installiert werden)*

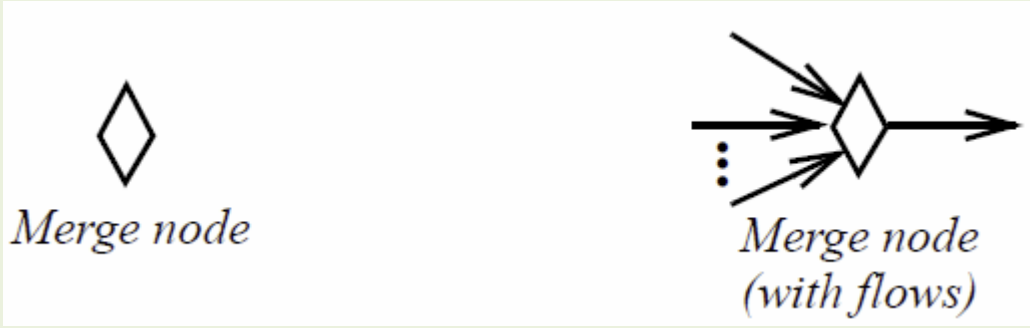
Ann.:
aus 50 Komponenten soll eine Anwendung enstehen

Entscheidungsknoten, Vereinigungsknoten

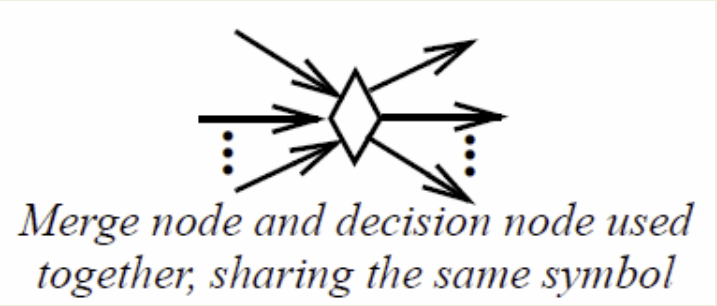
- mit syntaktischen Varianten



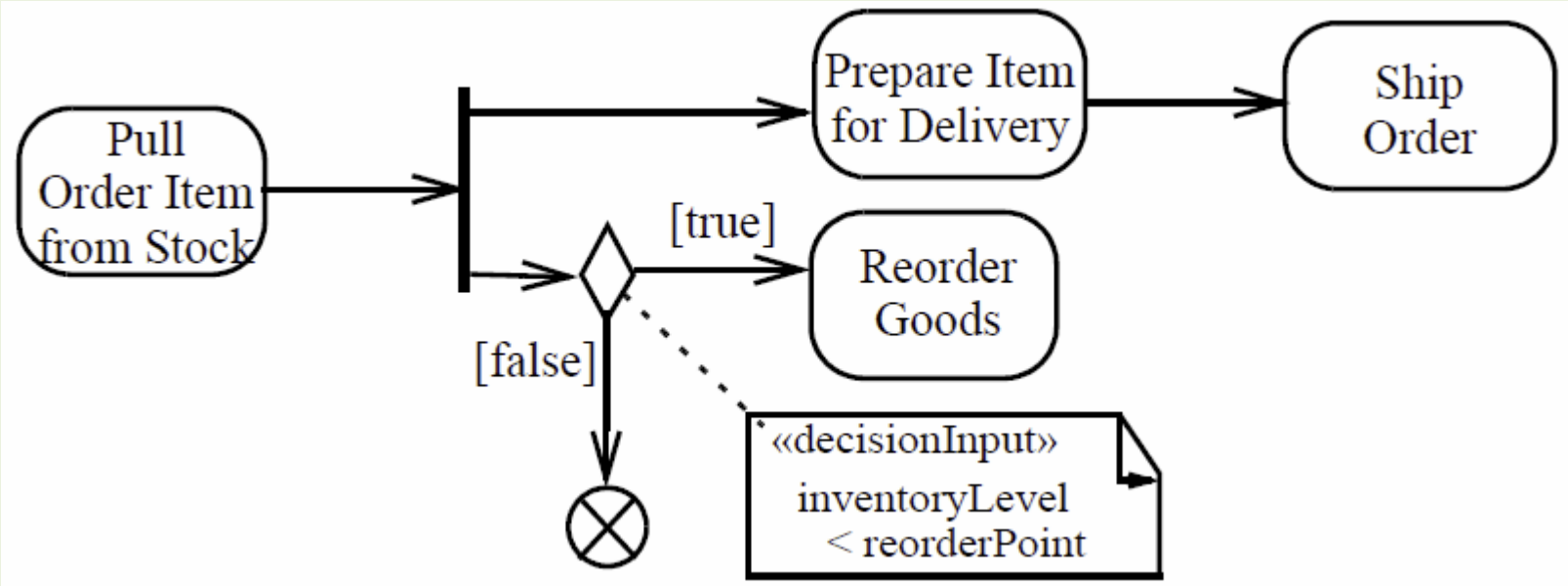
Schlüsselworte für Konzeptvariation



Keine Synchronisation!



DecisionInput-Spezifikation: Beispiel



Frohe Weihnachten

und einen guten Rutsch ins Neue Jahr