

Kurs OMSI im WiSe 2013/14

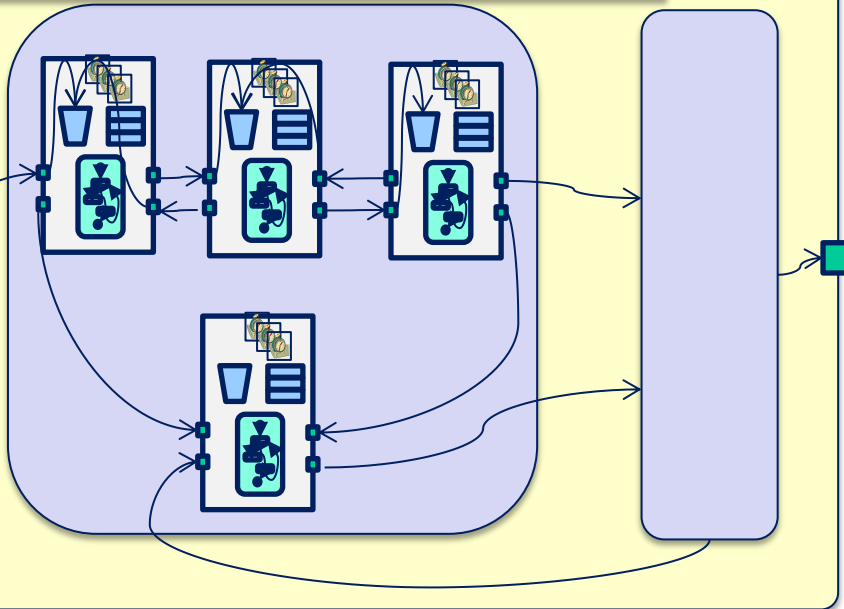
Objektorientierte Simulation mit ODEMx

Prof. Dr. Joachim Fischer
Dr. Klaus Ahrens
Dipl.-Inf. Ingmar Eveslage

fischer|ahrens|eveslage@informatik.hu-berlin.de

Herausforderung: System als Agent-Ensemble

- Agenten kommunizieren asynchron



UML-Probleme

- **Pool-bearbeitung:**
sog. semantischer Variationspunkt
(Reihenfolge, ...)
- **Adressierung** von Signalen
(nicht komplett gelöst,
Bekanntmachung der Agenten/Ports)
- **Übertragung** von Signalen
(ungelöst: Zeitverbrauch, Sicherheit)
- semantischer Variationspunkt
Offenheit bzgl. **Action-Sprache**
Datentypen, Anweisungen
- Do-Aktivität kann sowohl zeitdiskret als
auch **zeitkontinuierlich** sein

ODEMx: als geeignete Basis für eine Agent-Implementierung

Klasse Process ~ Basis für UML/SDL-Agent



1. Einführung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Beispiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Einordnung von UML
8. Klassifikation dynamischer Systeme
9. M&S eines Niedertemperaturofens

Verhaltens- und Zustandsgrößen

Modellierungsaspekte realer oder gedachter Phänomene

- Existenz/Substanz (Ausdehnung in Raum und Zeit)
repräsentiert durch statische und dynamische Objekt-Strukturen
- Verhaltensgrößen (messbare Eigenschaften)
repräsentiert durch Werte der Objektattribute
- Veränderung der Substanz (dynamisches Verhalten)
repräsentiert durch interagierende Objekte aktiver Klassen in Abhängigkeit einer Modellzeit bei Nutzung von Objekten passiven Klassen

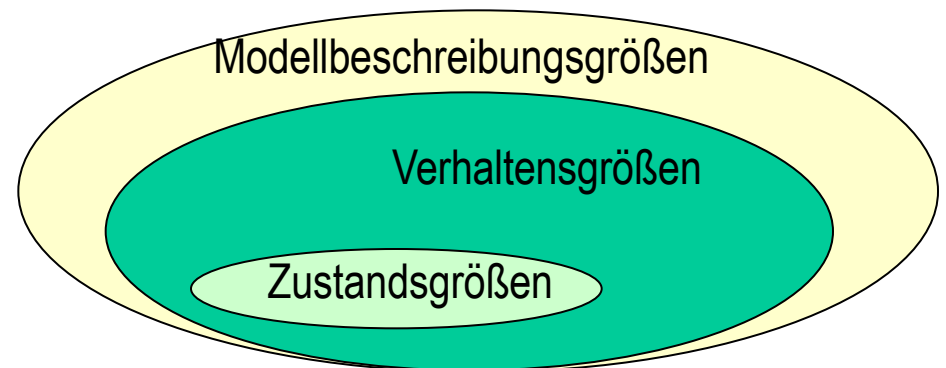
essentielle Verhaltensgrößen

- nicht immer beobachtbar
- **Zustandsgrößen** als ausgezeichnete Verhaltensgrößen
(spielen eine zentrale Rolle bei der Modellierung)

Zustandsgrößen

... sind

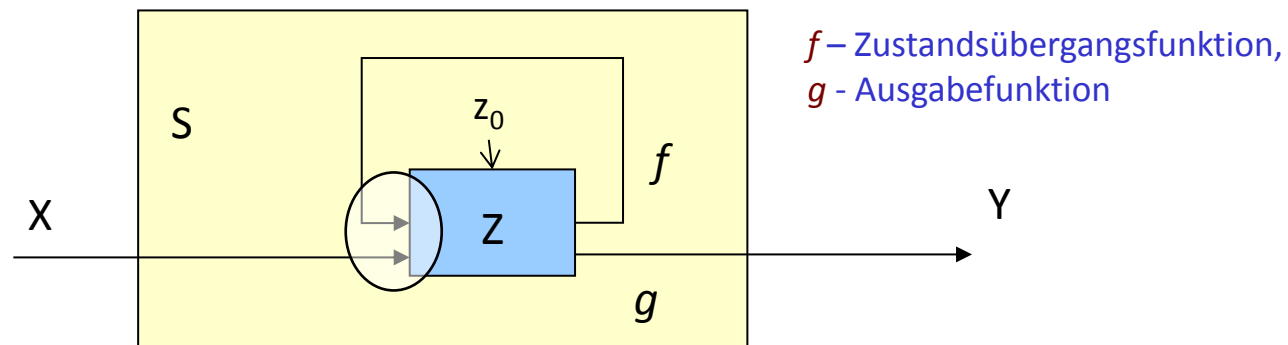
- Modellbeschreibungsgrößen, aus denen sich der Zustand eines Systems **vollständig** ergibt (Gedächtnis eines Systems)
→ Basis der Verhaltensbeschreibung
- Zustandsgrößen sind voneinander **unabhängig**
→ eine Zustandsgröße kann nicht als Kombination anderer Zustandsgrößen dargestellt werden
- sind **nicht immer eindeutig** definierbar
- sind i. Allg. strukturierte Größen



Zustandsänderungen (Prinzip)

- Sei Z n -dimensionaler Zustandsvektor (Zustand z = Belegung von Z), der Zustandsgrößen eines (Teil-)Systems S zu diesem Zeitpunkt beschreibt
- der (neue) **Zustand** ergibt sich aus dem bisherigen (**aktuellen**) Zustand bei Berücksichtigung von “Zuwachs” und “Reduktion (negativer Zuwachs)” für die Zustandsgrößen im betrachteten Zeitraum des Zustandswechsels
- ausgehend von einem ausgezeichneten Anfangszustand z_0

Zeit $t_0 \quad t_1 \quad t_2 \quad t_3$
 $z_0 \rightarrow z_1 \rightarrow z_2 \rightarrow z_3 \rightarrow \dots$ Zustandsentwicklung in Abhängigkeit der Zeit



f – Zustandsübergangsfunktion,
 g – Ausgabefunktion

Eingabe X

Zuwachs
im Zeitintervall $(t_k, t_{k+1}]$

Änderung des Zustandsvektors Z

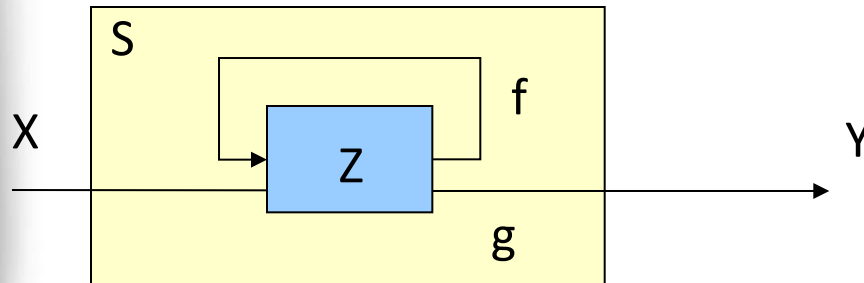
im Zeitintervall $[t_k, t_{k+1}]$
in Abhängigkeit von $x(t_{k+1}), z(t_k)$

Ausgabe Y

äußere Reaktion
des Systems auf die Eingabe
im Zeitintervall $(t_k, t_{k+1}]$

Allgemeine (Teil-)Systemdefinition

dient mehr der Klassifikation von Verhaltensmodellen

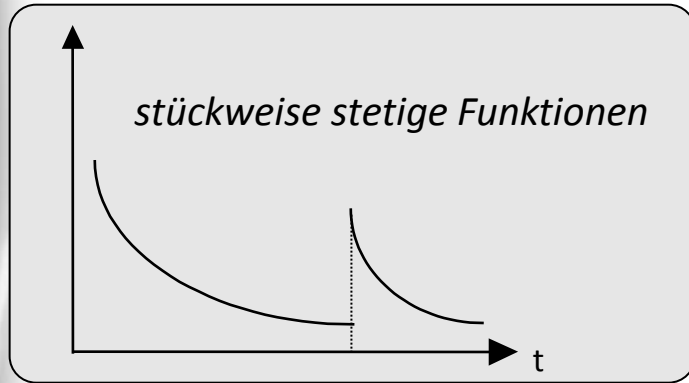


$$S = (Z, z_0, X, Y, f, g, \text{time})$$

- Z Menge der möglichen Zustände
- $z_0 \in Z$ Anfangszustand
- X Menge der möglichen Eingaben
- Y Menge der möglichen Ausgaben
- time Zeitbasis als (T, \leq, t_0) mit
 - Menge möglicher Zeitpunkte T ,
 - einer Ordnungsrelation \leq und
 - einem minimalen Element t_0
- f $Z \times X \times T \rightarrow Z$ als Zustandsübergangsfunktion
- g $Z \times X \times T \rightarrow Y$ als Ausgabefunktion

Arten von Zustandsänderungen

zeitkontinuierliche
Zustandsänderung



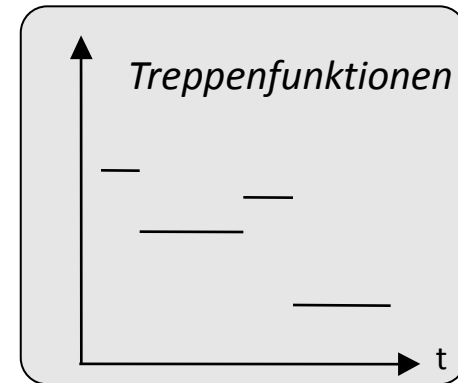
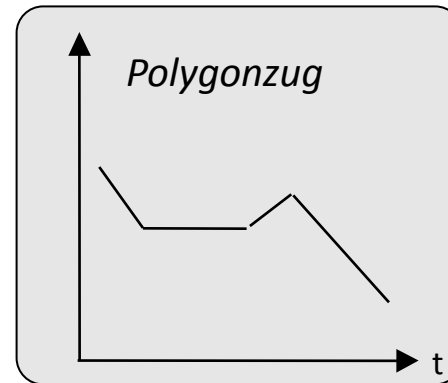
$$z'(t) = f(z(t), x(t), t)$$

mit $z(t) \in Z, x(t) \in X, t \in T$

Differentialgleichungen

numerische
Lösungsverfahren

zeitdiskrete
Zustandsänderung



$$z(t_{n+1}) = f(z(t_n), x(t_{n+1}), t_{n+1})$$

mit $z(t_n) \in Z, x(t_{n+1}) \in X, t_{n+1} \in T$

Differenzgleichungen
zelluläre Automaten

Ereignissimulationen

Prozesssimulation

Objektorientierte Simulation mit ODEMx

Verhaltensklassen zeitkontinuierlicher Art

Differentialgleichungen

Klassifikationsmöglichkeiten

n-ter Ordnung (erster Ordnung)

partielle
Differentialgleichungen

gewöhnliche
Differentialgleichungen

nichtlineare
Differentialgleichungen

lineare
Differentialgleichungen

Randwertaufgaben

Anfangswertaufgaben

- zeitkontinuierliche Prozesse im Beschreibungsmodell
- Approximation durch zeitdiskrete Prozesse (Anwendung numerischer Integrationsverfahren)
- zeitdiskrete Prozesse als Folgen von Ereignisrealisierungen im Simulationsmodell

ODEMx unterstützt:

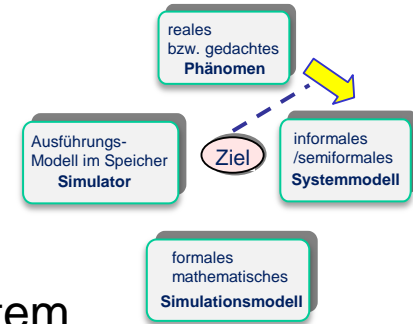
Gewöhnliche DGL-Systeme 1.Ordnung als AWA (linearer und nichtlinearer Art)

1. Einführung

1. Systemsimulation – was ist das?
2. Ein Blick zurück in die Anfänge
3. Modelle und Originale
4. Modellierungssprachen, Simulationsumgebungen
5. Beispiele aus der aktuellen Forschung
6. Paradigma der objektorientierten Modellierung
7. Einordnung von UML
8. Klassifikation dynamischer Systeme
9. M&S eines Niedertemperaturofens
10. Scheduler für zeitdiskrete und zeitkontinuierliche Systemmodelle

1. Schritt: Problemanalyse

- allgemein -



mit informaler Darstellung des Phänomens als System
(aus systemtheoretischer Sicht)

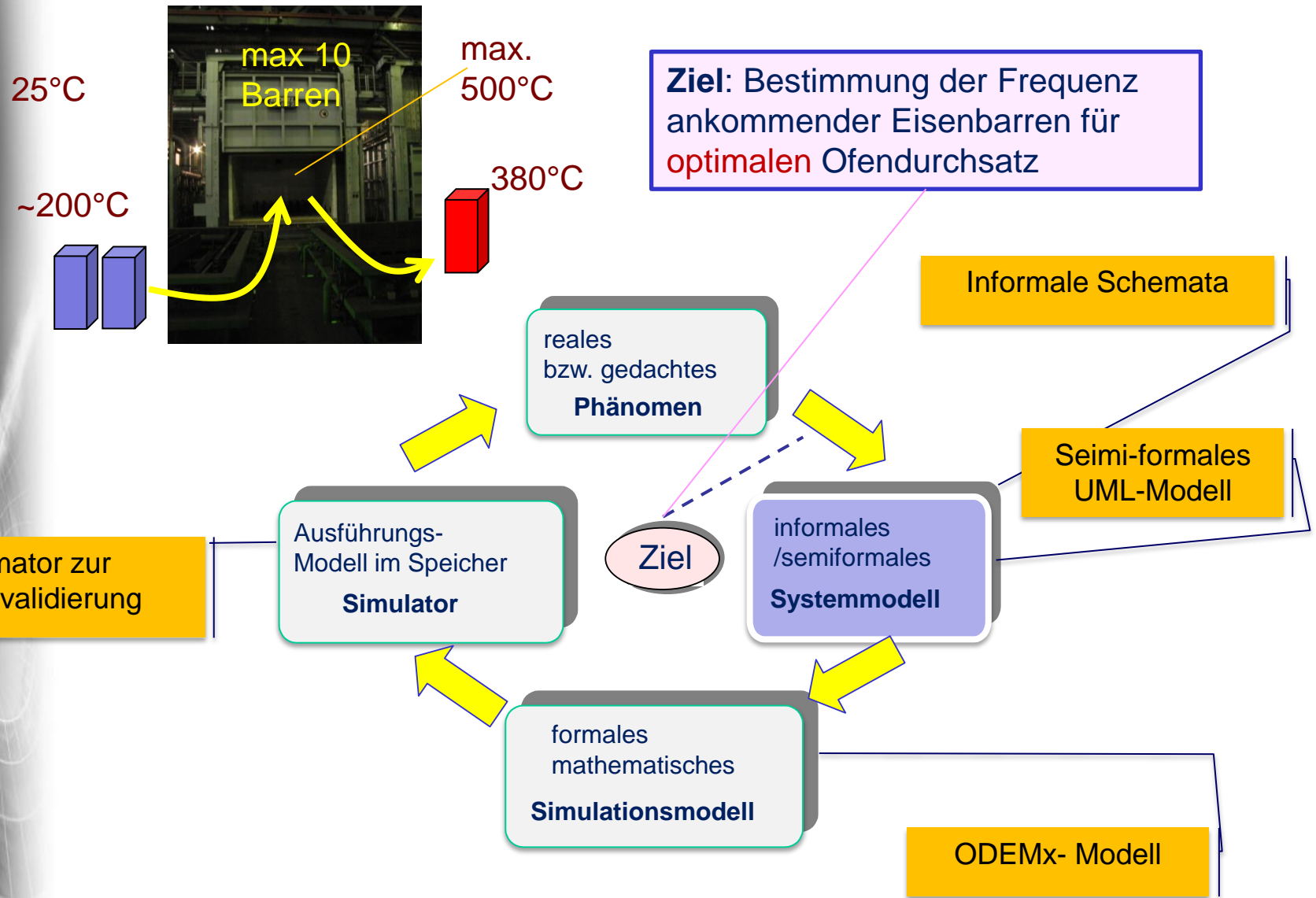
bei Identifikation von

- Systemelementen und Systemumgebung
- Relationen (Wechselwirkungen) zwischen Systemelementen untereinander und zur Umgebung

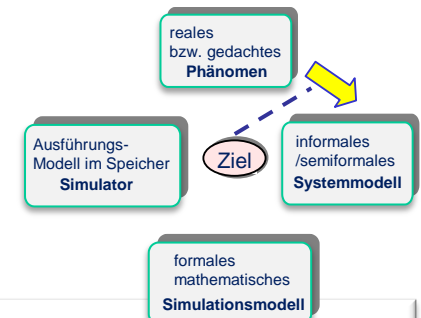
zur Erbringung des bereits bestimmten

- Systemzwecks / Untersuchungsziels

Beschickung eines Niedrigtemperaturofens



2. Schritt: Problemanalyse - Informale Darstellung -

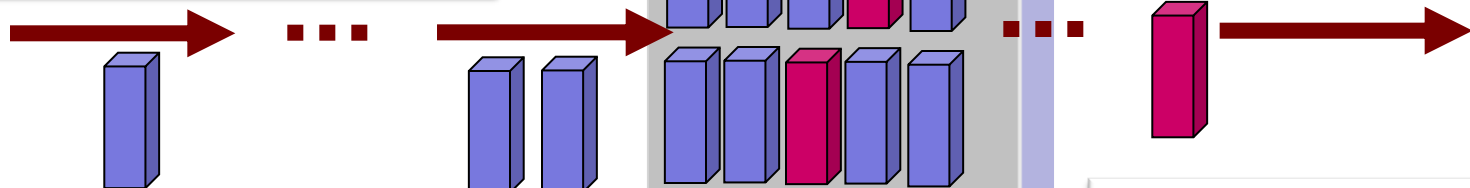


(5) Änderung der Ofentemperatur
in Abhängigkeit der Belegung
(maximal 500°C)

(4) Erwärmung der Barren
auf Zieltemperatur von mind. 380 °C

(1) stochastische Ankunftszeit
2-10 min,
stochastische Anfangstemperatur
~200 °C

(6) regelmäßige
Temperaturkontrolle
alle 2 min

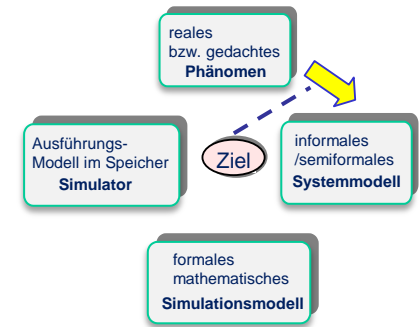


(2) evtl. Abkühlung
auf Umgebungstemperatur
25 °C möglich

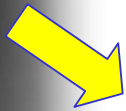
(3) Beschickung
Ofen hat begrenzte
Aufnahmekapazität
von 10 Barren

(7) individuelle Entnahme

3. Schritt: partielle Modellformalisierung - UML-Notation -



informal beschriebenes
Systemmodell



Assoziationen

passive Klassen

aktive Klassen

- **Struktur** (Objekt- und Klassendiagramm) und
- **Verhalten** als Ensemble asynchron kommunizierender Automaten
Zustandsdiagramm

bei Identifikation

- verwendeter **Modellklassen**
- und von **Zustands-** und **Bewertungsgrößen**

bei (zunächst noch) verbaler **Verhaltensbeschreibung**

lifeCycle

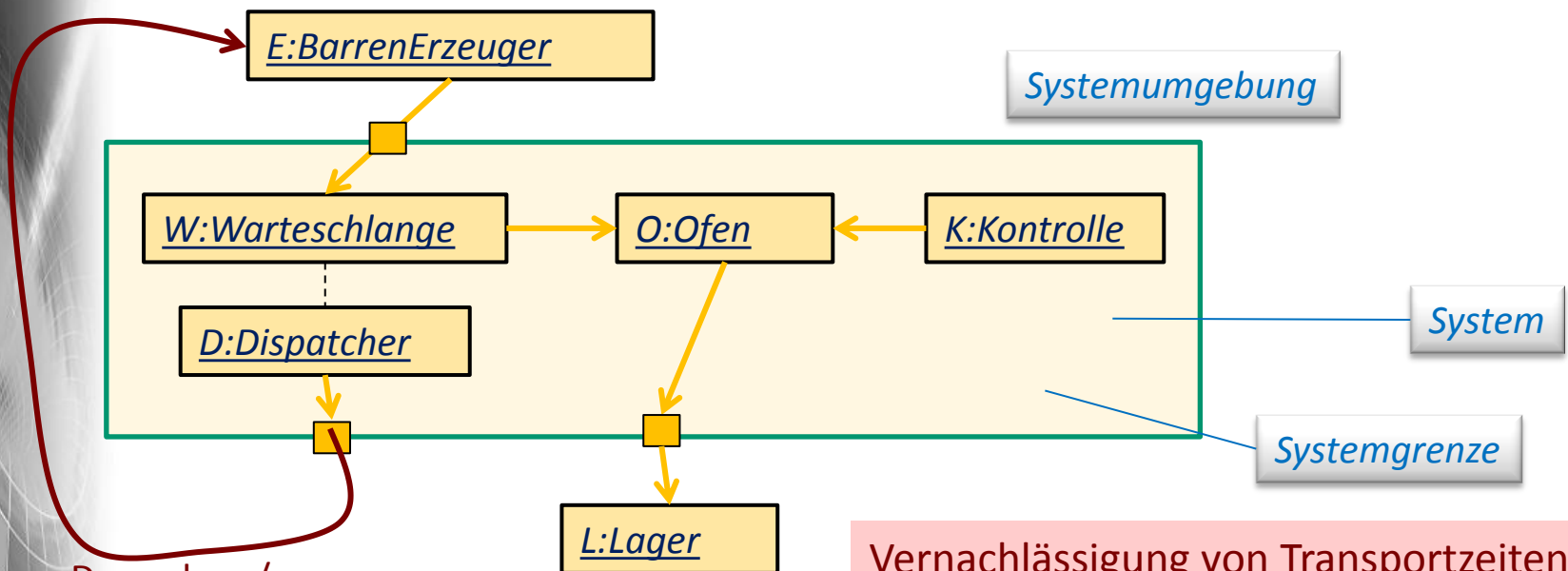
zeitdiskrete
Zustandsänderungen

zeitkontinuierliche
Zustandsänderungen

System (Umgebung, Grenze, Systemkomponenten)

Was ist zu prüfen?

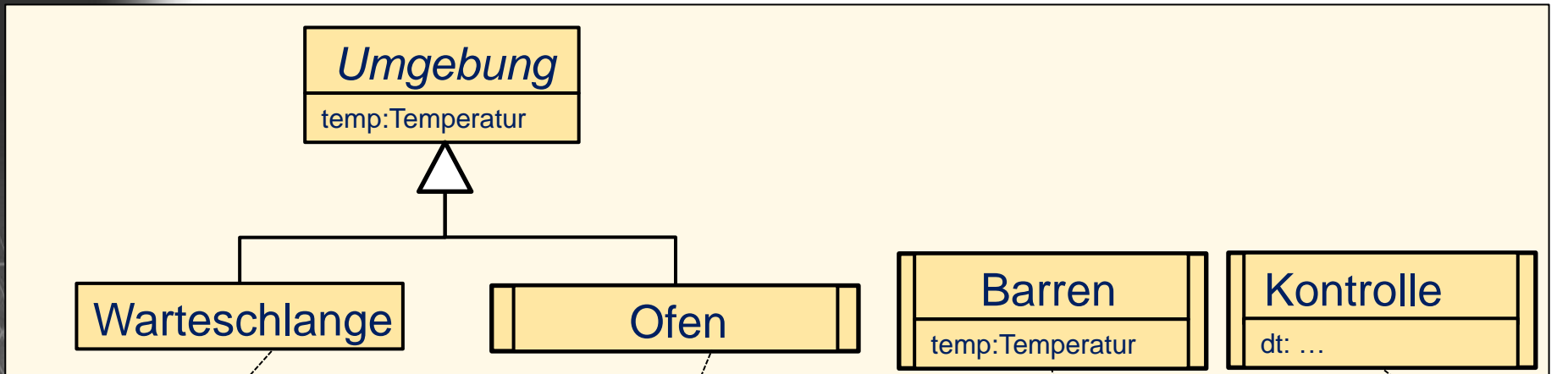
- können die Systemelemente in ihrer Wechselwirkung die Systemfunktionalität erbringen?
- Sind die Wechselwirkungen mit der Umgebung rückkopplungsfrei?



Drosselung/
Beschleunigung
der Barrenproduktion

würde Ausweitung der Systemgrenzen notwendig machen

Klassendiagramm, verbale Verhaltensspezifikation



kein eigenständiger **Lebenslauf**
temp ist mit 25°C konstant

FIFO-Warteschlange von Barren

Lebenslauf
 Temperaturänderung
 eines individuellen Ofens

Liste von Barren

Lebenslauf beschreibt
 Temperaturänderung
 eines individuellen Barrens

Lebenslauf beschreibt
 regelmäßige
 Tempprüfung

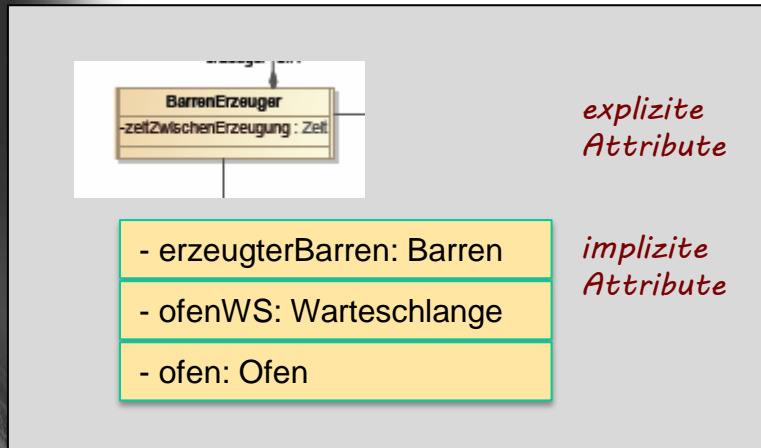
temp ändert sich in Abhängigkeit

- des momentanen Wertes von **temp** und
- der momentanen Temperaturen **aller** aktuell eingelagerten Barren

bei leerem Ofen wird **temp** sich asymptotisch 500 °C annähern

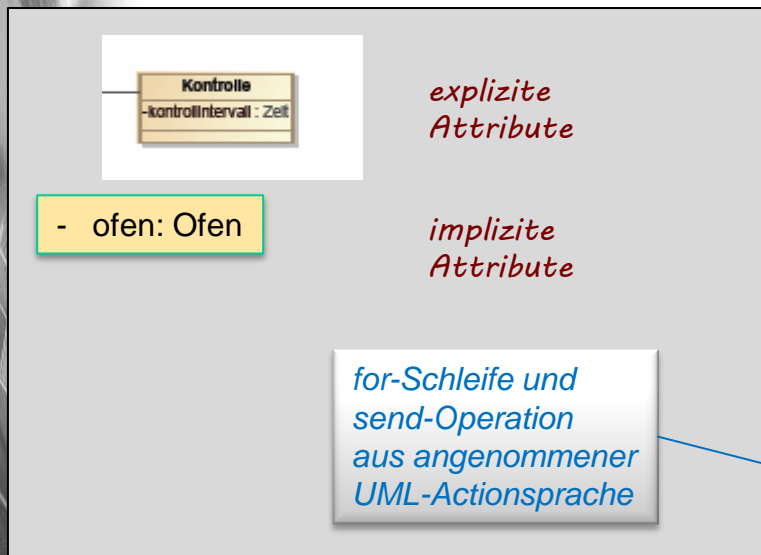
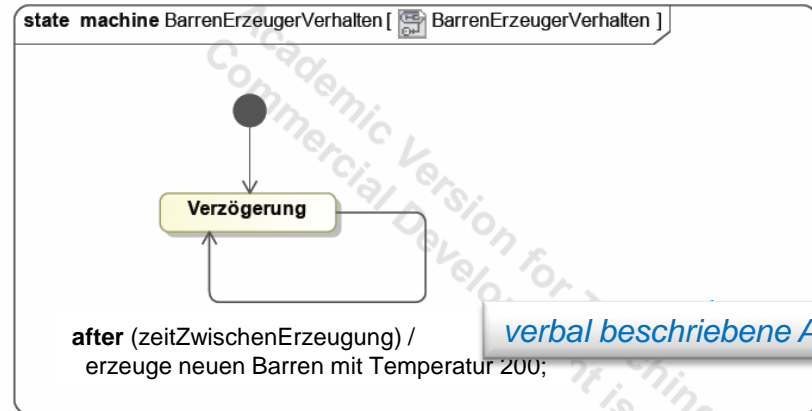
- 1) Objekt lagert sich entweder im Ofen oder in der Warteschlange ein
- 2) **temp** ändert sich in Abhängigkeit
 - des momentanen Wertes von **temp** und
 - der momentanen Temperatur der jeweils aktuellen polymorphen Umgebung:
 - a) außerhalb oder
 - b) innerhalb des Ofens
- 3) Objekt verlässt Ofen, sobald es von der Kontrolle erkannt worden ist

Verhaltensdiagramm (BarrenErzeuger, Kontrolle)



explizite Attribute

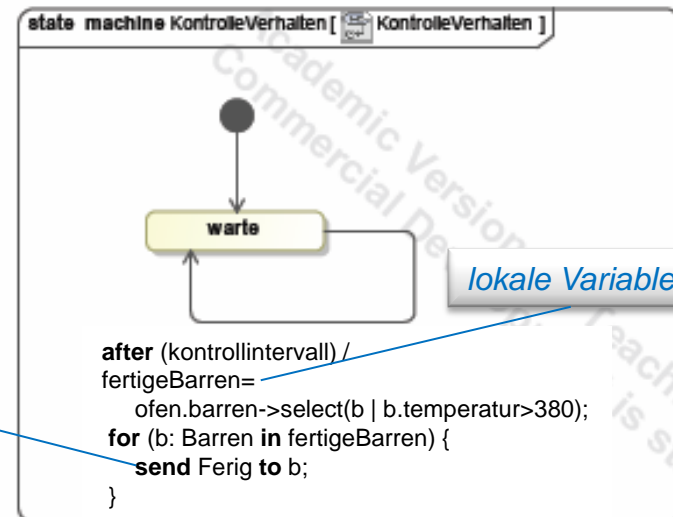
implizite Attribute



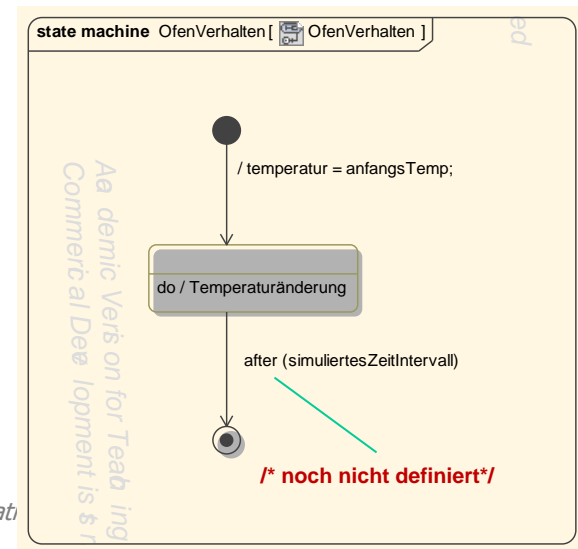
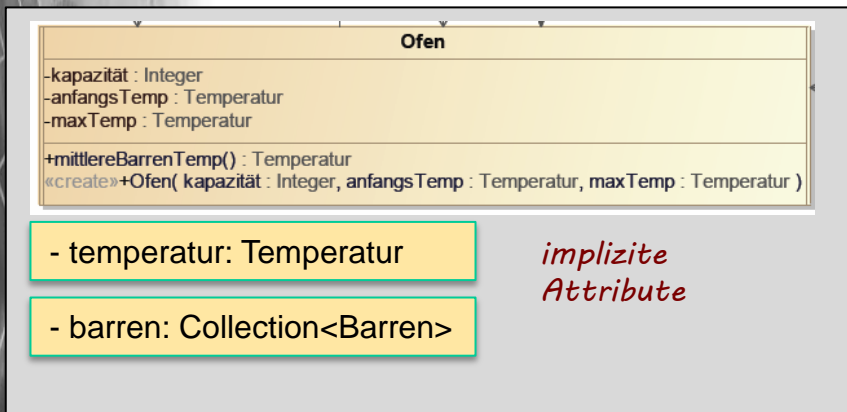
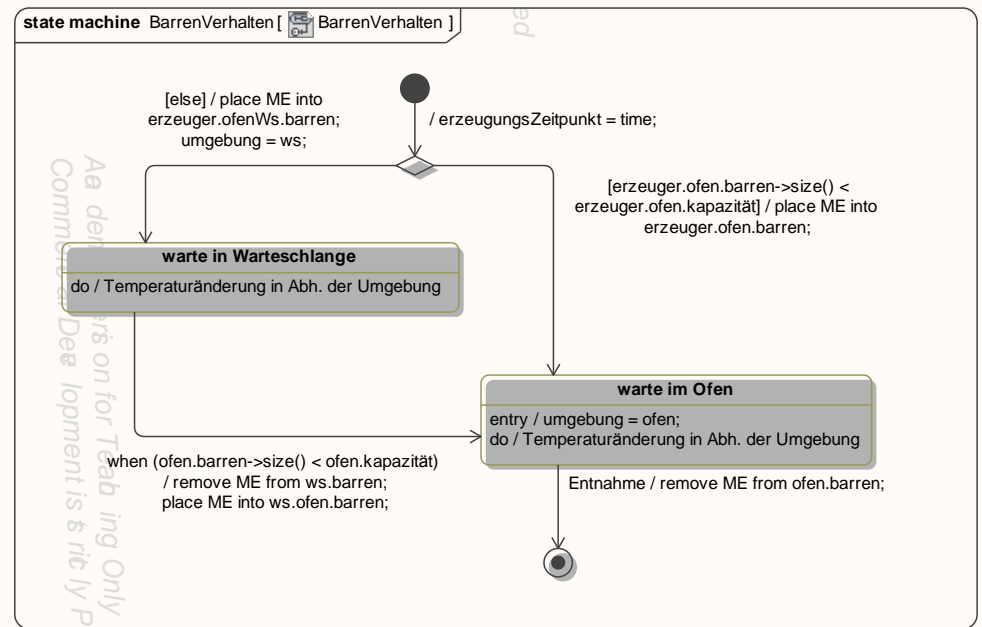
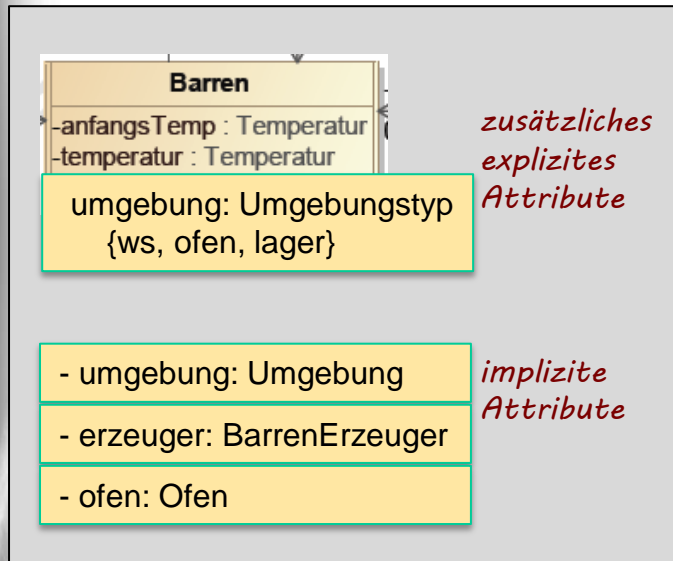
explizite Attribute

implizite Attribute

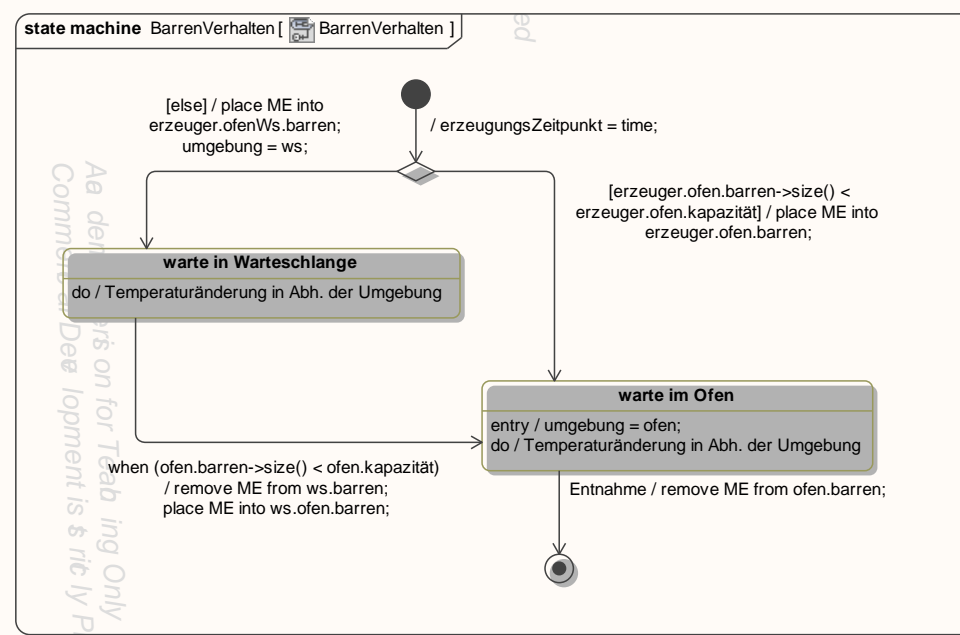
for-Schleife und send-Operation aus angenommener UML-Actionsprache



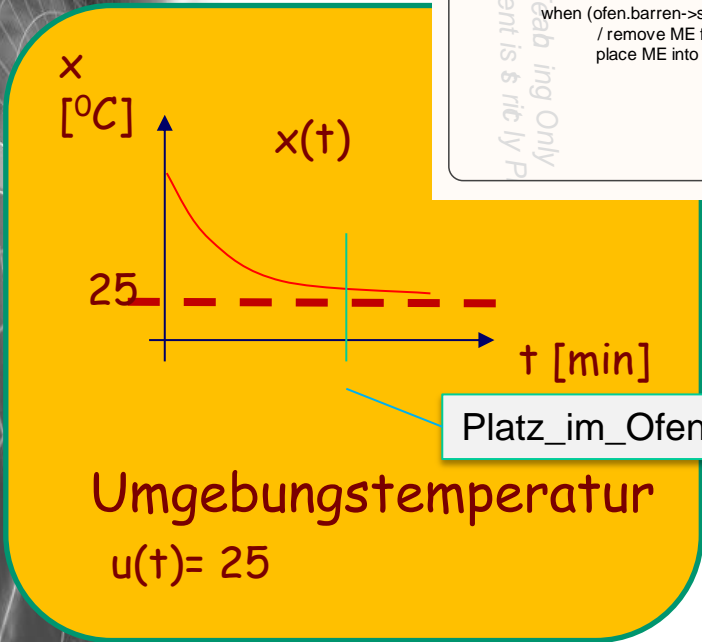
Verhaltensdiagramm (Barren, Ofen)



Barren: Temperaturänderung

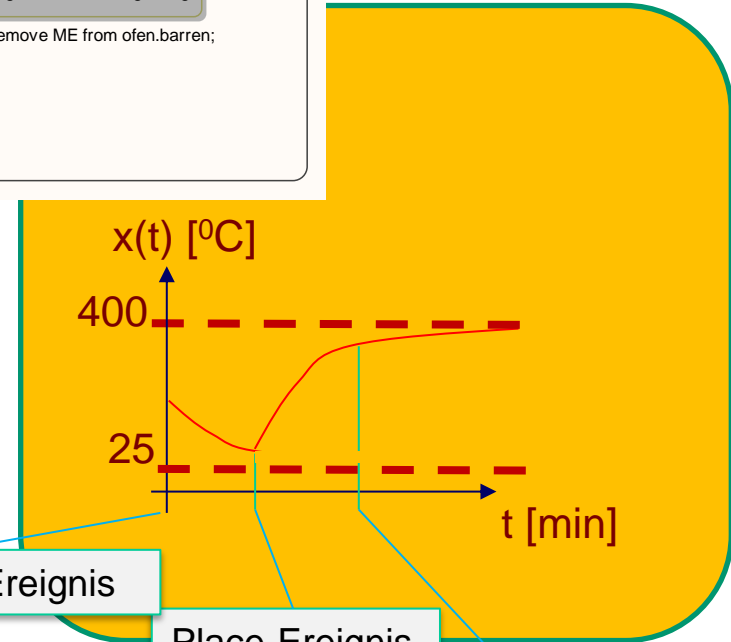


umg= ws



Platz_im_Ofen-Ereignis

umg= ofen

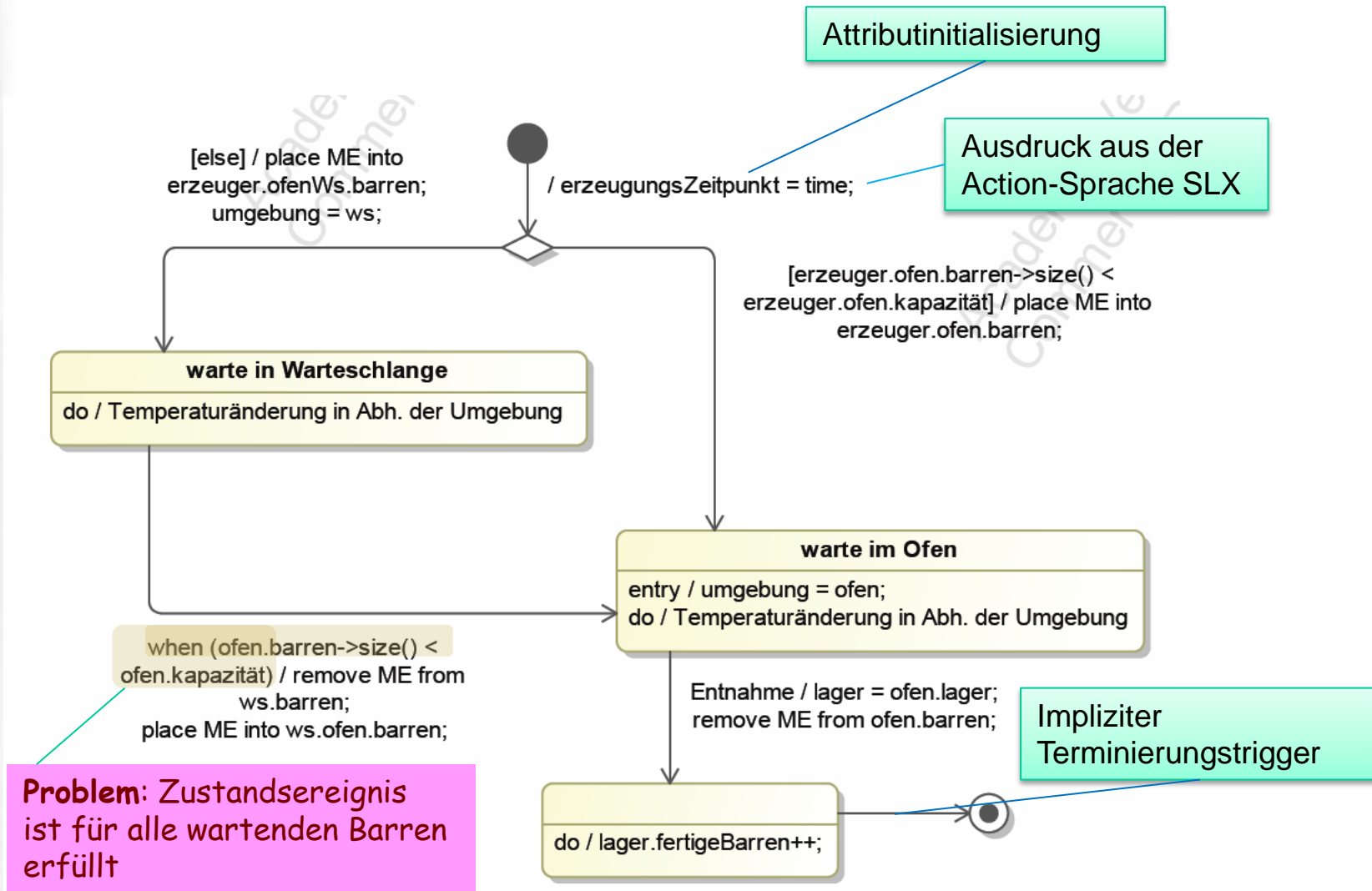


Erzeugungs-Ereignis

Place-Ereignis

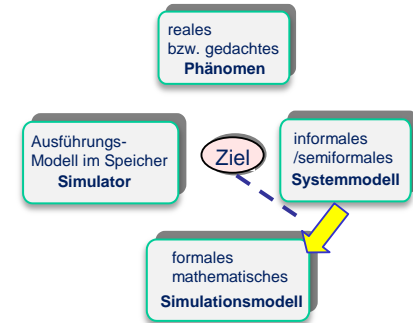
Remove-Ereignis

Modifiziertes Barrenverhalten



4. Schritt: Vervollständigung der Formalisierung

- Temperaturänderung als DGL -



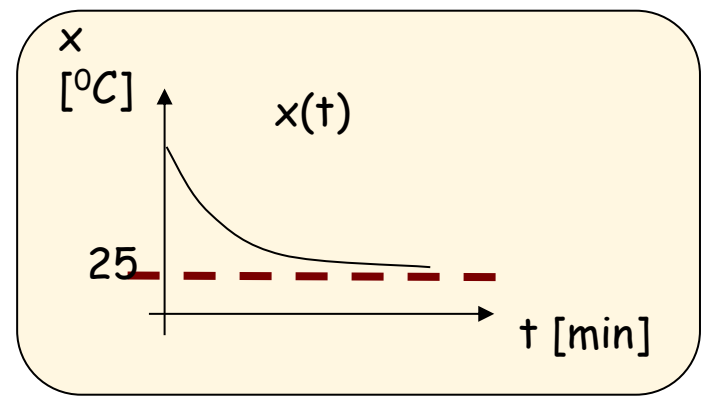
negativer Wert für $x'(t) \rightarrow$ Reduktion von $x(t)$
 Änderung von $x(t)$:
 • zunächst stark,
 • dann schwach,
 \rightarrow asymptotische Annäherung an den Wert 25

umg= WS

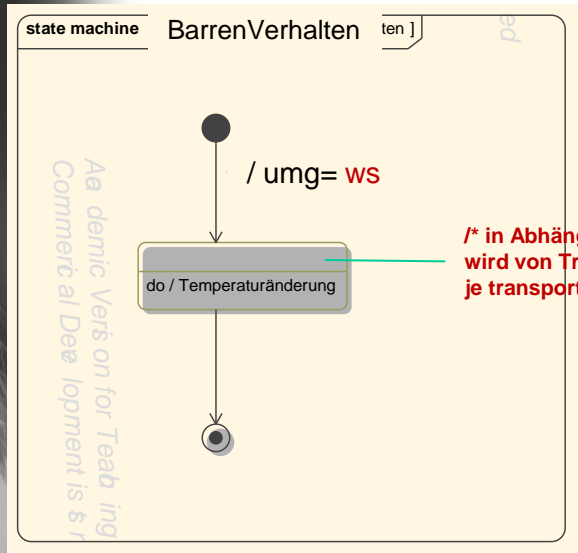
Barrentemperatur $x(t)$
 $x'(t) = \{ 25 - x(t) \} / 7$

25 - 200

Anfangstemperatur: ca. 200 °C



4. Schritt: Vervollständigung der Formalisierung - Temperaturänderung als DGL -



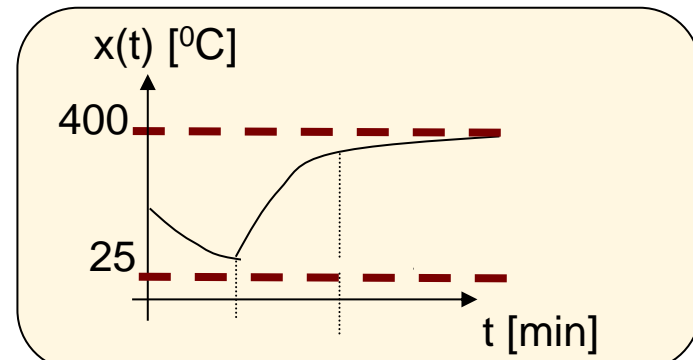
umg= **ofen**

Positiver Wert für $x'(t) \rightarrow$ Erhöhung von $x(t)$
 Änderung von $x(t)$:
 • zunächst stark,
 • dann schwach,
 \rightarrow asymptotische Annäherung an den Wert y

Barrentemperatur $x(t)$

$$x'(t) = \{ y(t) - x(t) \} / 7$$

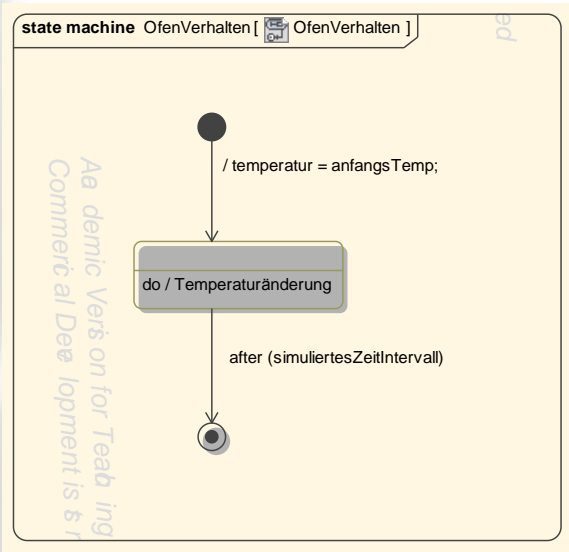
400 - 200



aber: Ofentemperatur $y(t)$ ist nicht konstant, eine explizite Darstellung gibt es nicht, deshalb kann man die Zeit, wie lange ein beliebiger Barren im Ofen benötigt, nicht vorab berechnen.

4. Schritt: Vervollständigung der Formalisierung

- Temperaturänderung als DGL -



Energiezufuhr: max. 500°C

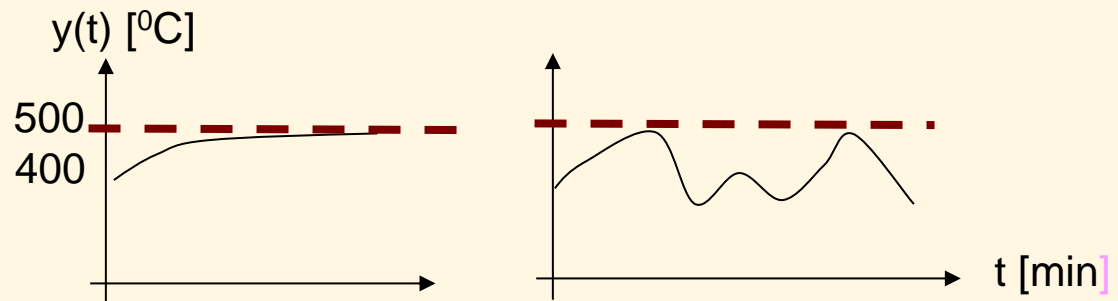
n(t): Anzahl von Barren
im Ofen

m(t): mittlere Temperatur
der Barren im Ofen

Anfangstemperatur: 400°C

Ofentemperatur y als $y(t)$

$$y'(t) = 500 - y(t) - n(t) \cdot \{y(t) - m(t)\}$$



4. Schritt: Modellformalisierung (nun abstrakt vollzogen)

- Festlegung der Art der Zustandsänderung → Verhaltensklassen

von

- Struktur (Objekt- und Klassenstruktur) und
- Verhalten

bei Identifikation

- verwendeter Modellklassen
- und von Zustands- und Bewertungsgrößen

bei Bestimmung

- der Verhaltensfunktionen/-Gleichungen

passive Klassen

aktive Klassen

UML bietet
Zustandsmaschine
zur Beschreibung
Aber mit semantischen
Päzisierungen

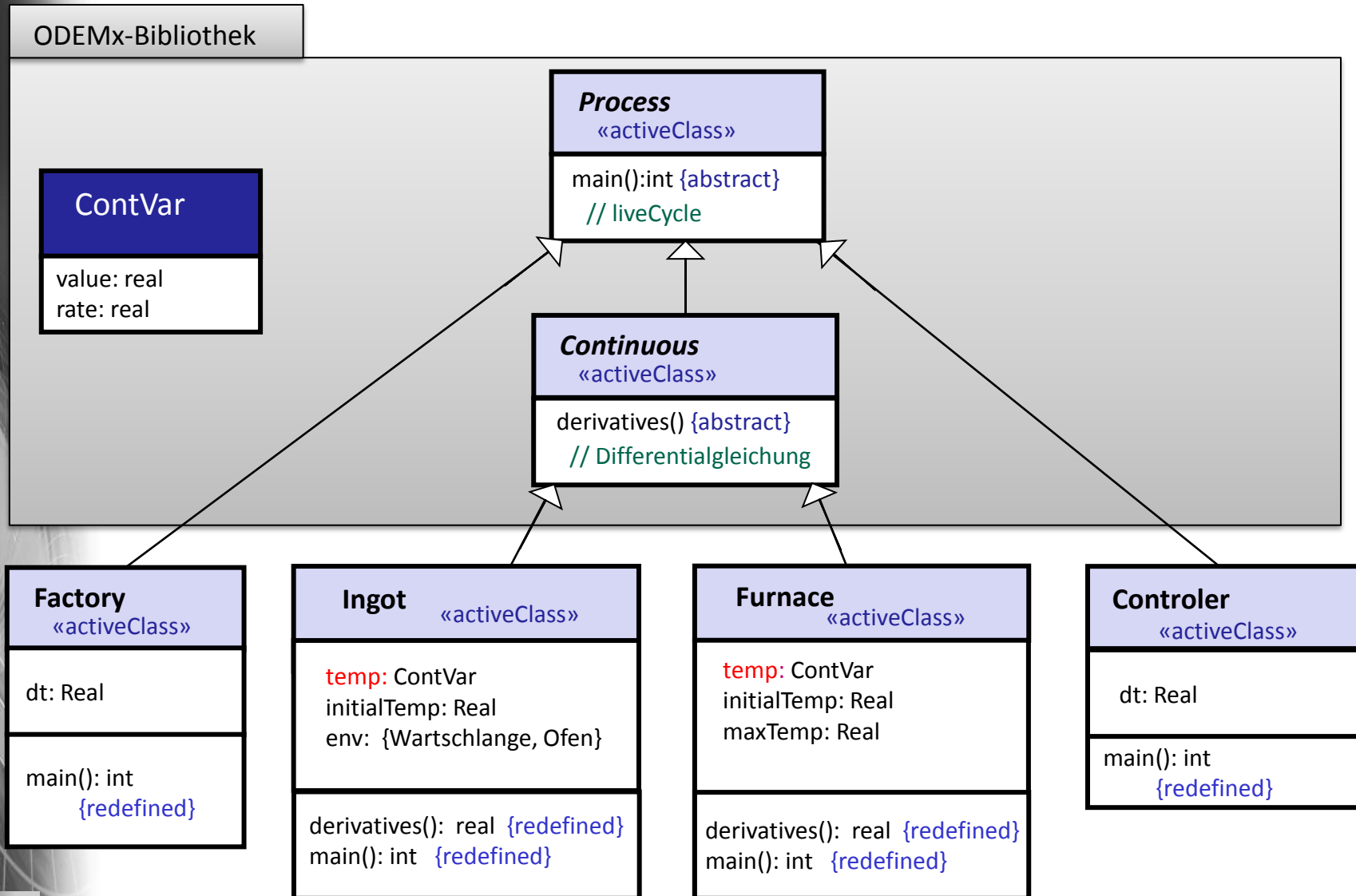
lifeCycle

zeitdiskrete
Zustandsänderungen

zeitkontinuierliche
Zustandsänderungen

6. Schritt: Modellformalisierung

- Identifikation von Modellgrößen der Strukturelemente



Laufzeitverwaltung auf einer Ein-Prozessormaschine

Vor.: alle statischen Systemstrukturelemente sind als Objekte generiert:
Objekte aktiver Klassen führen ihre Starttransitionen zur Modellzeit 0.0 aus.

