

Verhaltensorientierte Modellierung 2

Modul 11 (v1.0)

Kanonikvorlesung: Foundations of Computing

Heiko Mantel

MAIS, TU Darmstadt, WS11/12

Übersicht: Modul 11

Modulare Modellierung

- ☐ Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- ☐ Modellierung synchroner Ausführung
- ☐ Modellierung asynchrone Ausführung

Kommunikation zwischen Komponenten

- ☐ shared memory
- ☐ message passing

Nebenläufige Ausführung

Varianten nebenläufiger Ausführung

- ☐ synchrone Ausführung
- ☐ asynchrone Ausführung

Synchrone, nebenläufige Ausführung

- ☐ Die Systemkomponenten führen jeweils gleichzeitig einen Berechnungsschritt durch. Ein Berechnungsschritt des Systems beinhaltet also einen Schritt jeder Systemkomponente.
- ☐ Beispiel: getaktete Hardwaregatter

Asynchrone, nebenläufige Ausführung

- ☐ Die Systemkomponenten führen unabhängig voneinander ihre Berechnungsschritte durch. Ein Berechnungsschritt des Systems beinhaltet also einen Schritt einer Systemkomponente.
- ☐ Beispiel: verteiltes System

Modellierung Nebenläufiger Systeme 1

Anforderung

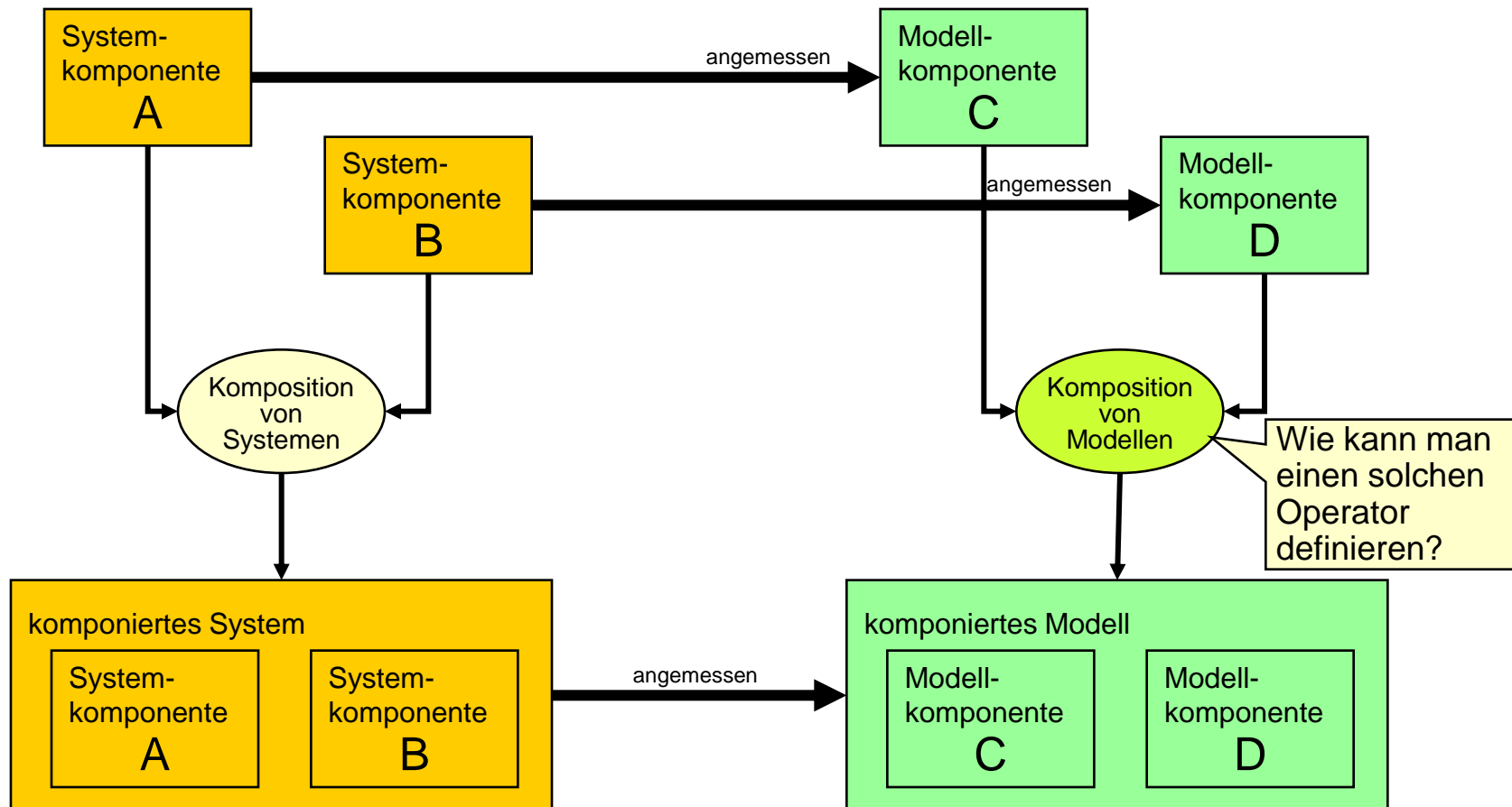
- ☐ Es sollte möglich sein, die Modularität eines nebenläufigen Systems bei der Modellierung auszunutzen.
- ☐ Vorteile
 - ☐ Die Modellierung wird strukturiert.
 - ☐ Die Komplexität des Modellierens wird reduziert.

Voraussetzung

- ☐ Man braucht eine Definition der Komposition von Modellen von Systemkomponenten.
- ☐ Es muss klar sein, in welchen Fällen man mit dieser Form der Komposition aus Modellen der Systemkomponenten, die angemessen sind, ein Modell des zusammengesetzten Systems erhält, das wiederum angemessen ist.

Modellierung Nebenläufiger Systeme 2

Modulare Modellierung von nebenläufigen Systemen



Modellierung Nebenläufiger Systeme 3

Auf den folgenden Folien werden eingeführt:

- ☐ Synchrone Produktkomposition
 - ☐ Eine Form der Komposition, die eine angemessene Modellierung von synchronen, nebenläufigen Systemen ergibt, wenn die Modellierungen der Komponenten angemessen sind und die Komponenten nicht miteinander kommunizieren.
- ☐ Asynchrone Produktkomposition
 - ☐ Eine Form der Komposition, die eine angemessene Modellierung von asynchronen, nebenläufigen Systemen ergibt, wenn die Modellierungen der Komponenten angemessen sind und die Komponenten nicht miteinander kommunizieren.

Vereinfachung

- ☐ Zunächst beschränken wir uns auf Systeme, deren Komponenten nicht miteinander kommunizieren.
- ☐ Diese Vereinfachung wird im nächsten Abschnitt dieses Moduls aufgehoben.

Übersicht: Modul 11

Modulare Modellierung

- ☐ Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- ☐ Modellierung synchroner Ausführung
- ☐ Modellierung asynchrone Ausführung

Kommunikation zwischen Komponenten

- ☐ shared memory
- ☐ message passing

Synchrone Nebenläufigkeit 1

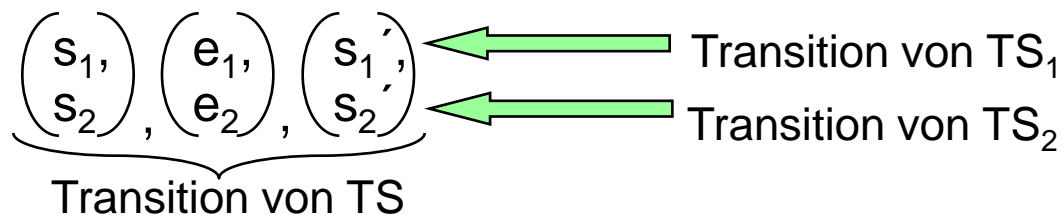
Definition

Seien $TS_1 = (S_1, S0_1, E_1, \rightarrow_1)$ und $TS_2 = (S_2, S0_2, E_2, \rightarrow_2)$ zwei Transitionssysteme. Die **synchrone Produktkomposition** von TS_1 und TS_2 ist das Transitionssystem $TS = (S, S0, E, \rightarrow)$ wobei

- ☐ $S = S_1 \times S_2$
- ☐ $S0 = S0_1 \times S0_2$
- ☐ $E = E_1 \times E_2$
- ☐ $\rightarrow = \{ ((s_1, s_2), (e_1, e_2), (s_1', s_2')) \in S \times E \times S \mid (s_1, e_1, s_1') \in \rightarrow_1 \wedge (s_2, e_2, s_2') \in \rightarrow_2 \}$

Beobachtungen

- ☐ Jedes Paar von Anfangszuständen von TS_1 und TS_2 ist ein Anfangszustand von TS .
- ☐ Damit TS eine Transition machen kann, müssen TS_1 und TS_2 entsprechende Transitionen machen:



Synchrone Nebenläufigkeit 2

Theorem

Seien $TS_1 = (S_1, S_{0_1}, E_1, \rightarrow_1)$ und $TS_2 = (S_2, S_{0_2}, E_2, \rightarrow_2)$ zwei Transitionssysteme. Sei $TS = (S, S_0, E, \rightarrow)$ das synchrone Produkt von TS_1 und TS_2 .

□ Wenn $\rightarrow_2 \neq \emptyset$ dann gilt

$$\begin{aligned} \rightarrow_1 = \{ (s_1, e_1, s_1') \in S_1 \times E_1 \times S_1 \\ \mid \exists s_2, s_2' \in S_2: \exists e_2 \in E_2: ((s_1, s_2), (e_1, e_2), (s_1', s_2')) \in \rightarrow \} \end{aligned}$$

□ Wenn $\rightarrow_1 \neq \emptyset$ dann gilt

$$\begin{aligned} \rightarrow_2 = \{ (s_2, e_2, s_2') \in S_2 \times E_2 \times S_2 \\ \mid \exists s_1, s_1' \in S_1: \exists e_1 \in E_1: ((s_1, s_2), (e_1, e_2), (s_1', s_2')) \in \rightarrow \} \end{aligned}$$

Beobachtungen für TS_1

- Für jede mögliche Transition $((s_1, s_2), (e_1, e_2), (s_1', s_2'))$ von TS erhält man eine mögliche Transition von TS_1 indem man die drei Paare jeweils durch ihre ersten Elemente ersetzt, d.h. $s_1 - e_1 \rightarrow_1 s_1'$.
- Wenn $\rightarrow_2 \neq \emptyset$ gilt, dann gibt es für jede mögliche Transition (s_1, e_1, s_1') von TS_1 mindestens eine Transition von TS , die folgende Form hat: $(s_1, s_2) - (e_1, e_2) \rightarrow (s_1', s_2')$.

Asynchrone Nebenläufigkeit 1

Definition

Seien $TS_1 = (S_1, S_{0_1}, E_1, \rightarrow_1)$ und $TS_2 = (S_2, S_{0_2}, E_2, \rightarrow_2)$ zwei Transitionssysteme. Die **asynchrone Produktkomposition** von TS_1 und TS_2 ist das Transitionssystem $TS = (S, S_0, E, \rightarrow)$ wobei

- ☐ $S = S_1 \times S_2$
- ☐ $S_0 = S_{0_1} \times S_{0_2}$

- ☐ $E = E_1 \cup E_2$
- ☐ $\rightarrow = \{ ((s_1, s_2), e, (s_1', s_2')) \in S \times E \times S \mid ((s_1, e, s_1') \in \rightarrow_1 \wedge e \in E_1 \wedge s_2' = s_2) \vee ((s_2, e, s_2') \in \rightarrow_2 \wedge e \in E_2 \wedge s_1' = s_1) \}$

Beachte die Unterschiede zur synchronen Komposition!

Beobachtungen

- ☐ Jedes Paar von Anfangszuständen von TS_1 und TS_2 ist ein Anfangszustand von TS .
- ☐ Damit TS eine Transition machen kann, muss entweder TS_1 oder TS_2 eine entsprechende Transition machen.

Asynchrone Nebenläufigkeit 2

Theorem

Seien $TS_1 = (S_1, S0_1, E_1, \rightarrow_1)$ und $TS_2 = (S_2, S0_2, E_2, \rightarrow_2)$ zwei Transitionssysteme. Sei $TS = (S, S0, E, \rightarrow)$ das asynchrone Produkt von TS_1 und TS_2 . **Wenn $E_1 \cap E_2 = \emptyset$** dann gilt:

- $\rightarrow_1 = \{ (s_1, e_1, s_1') \in S_1 \times E_1 \times S_1 \mid \exists s_2 \in S_2: ((s_1, s_2), e_1, (s_1', s_2)) \in \rightarrow \}$
- $\rightarrow_2 = \{ (s_2, e_2, s_2') \in S_2 \times E_2 \times S_2 \mid \exists s_1 \in S_1: ((s_1, s_2), e_2, (s_1, s_2')) \in \rightarrow \}$

Beobachtungen für TS_1

- Wenn $E_1 \cap E_2 = \emptyset$ gilt, dann erhält man für jede mögliche Transition $((s_1, s_2), e_1, (s_1', s_2))$ von TS mit $e_1 \in E_1$ eine mögliche Transition von TS_1 indem man die zwei Paare jeweils durch ihre ersten Elemente ersetzt, d.h. $s_1 - e_1 \rightarrow_1 s_1'$.
- Für jede mögliche Transition (s_1, e_1, s_1') von TS_1 mindestens eine Transition von TS , die folgende Form hat: $(s_1, s_2) - e_1 \rightarrow (s_1', s_2')$.

Übersicht: Modul 11

Modulare Modellierung

- ☐ Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- ☐ Modellierung synchroner Ausführung
- ☐ Modellierung asynchrone Ausführung

Kommunikation zwischen Komponenten

- ☐ shared memory
- ☐ message passing

Kommunikation

Wie kommunizieren Komponenten in einem verteilten System?

- ☐ gemeinsamer Speicher
- ☐ Austausch von Nachrichten

Gemeinsamer Speicher (engl.: shared memory)

- ☐ Eine Komponente aktualisiert den gemeinsamen Speicher. Alle Komponenten sehen anschließend den aktualisierten Speicher.
- ☐ Problematik: Wie geht man mit Schreibkonflikten um?
 - ☐ siehe Übungsblatt

Austausch von Nachrichten (engl.: message passing)

- ☐ Eine Komponente sendet eine Nachricht, und eine andere Komponente empfängt diese Nachricht.

Im Rest dieses Moduls werden wir nur asynchrone nebenläufige Systeme betrachten. Synchrone, nebenläufige Systeme werden in der Übung betrachtet.

Shared Memory 1

Definition

Zwei Transitionssysteme $TS_1 = (S_1, SO_1, E_1, \rightarrow_1)$ und $TS_2 = (S_2, SO_2, E_2, \rightarrow_2)$ heißen **speicherkomponierbar**, wenn sich die Zustände in S_1 und S_2 in einen lokalen und in einen globalen Anteil zerlegen lassen, d.h. wenn es SL_1 , SL_2 , und SG gibt, so dass $\square S_1 = SL_1 \times SG$ und $S_2 = SL_2 \times SG$.

Der Wertebereich SG modelliert den gemeinsamen Speicher der beiden Komponenten. Die Wertebereiche SL_1 und SL_2 modellieren den jeweiligen lokalen Speicher der Komponenten TS_1 und TS_2 .

Veranschaulichung der geforderten Speicherstruktur



Shared Memory 2

Beispiel

Der Zustand eines Informationssystems sei durch die Werte einer Menge von Programmvariablen VAR gegeben. Um eine speicherkomponierbare Modellierung zu erhalten, partitionieren wir die Menge VAR in zwei disjunkte Teilmengen:

- $LVAR$ die Menge der lokalen Variablen und
- $GVAR$ die Menge der globalen Variablen.

Die Menge der Zustände kann jetzt wie folgt definiert werden:

$$\begin{aligned} \square S &= \{ (l, g) \mid l : LVAR \rightarrow VAL \wedge g : GVAR \rightarrow VAL \} \\ &= (LVAR \rightarrow VAL) \times (GVAR \rightarrow VAL). \end{aligned}$$

Somit hat die Menge der Zustände die auf der vorigen Folie verlangte Form. Die Komponente ist z.B. mit einer Komponente komponierbar, deren Zustände wie folgt definiert sind:

$$\begin{aligned} \square S' &= \{ (l, g) \mid l : \textcolor{teal}{LVAR}' \rightarrow VAL \wedge g : GVAR \rightarrow VAL \} \\ &= (\textcolor{teal}{LVAR}' \rightarrow VAL) \times (GVAR \rightarrow VAL). \end{aligned}$$

Shared Memory 3

Definition

Seien $TS_1 = (SL_1 \times SG, S0_1, E_1, \rightarrow_1)$ und $TS_2 = (SL_2 \times SG, S0_2, E_2, \rightarrow_2)$ zwei speicherkomponierbare Transitionssysteme. Die **shared-memory Komposition** von TS_1 und TS_2 ergibt das Transitionssystem $TS = (S, S0, E, \rightarrow)$ wobei

- $S = SL_1 \times SL_2 \times SG$
- $S0 = \{ (sl_1, sl_2, sg) \in SL_1 \times SL_2 \times SG \mid (sl_1, sg) \in S0_1 \wedge (sl_2, sg) \in S0_2 \}$
- $E = E_1 \cup E_2$
- $\rightarrow = \{ ((sl_1, sl_2, sg), e, (sl_1', sl_2', sg')) \in S \times E \times S \mid ((sl_1, sg), e, (sl_1', sg')) \in \rightarrow_1 \wedge e \in E_1 \wedge sl_2' = sl_2 \vee ((sl_2, sg), e, (sl_2', sg')) \in \rightarrow_2 \wedge e \in E_2 \wedge sl_1' = sl_1 \}$

Message Passing 1

Intuition der Modellierung

- ❑ Der Transport einer Nachricht m zwischen zwei Komponenten A und B wird durch das gleichzeitige Geschehen desselben Ereignisses in beiden Komponenten modelliert.

Beispiel

- ❑ **ch.u.v.m** modelliert den Transport einer Nachricht m von Nutzer u zu Nutzer v über den Kanal ch .
- ❑ Der direkte Transport der Nachricht von u zu v in einem Systemverhalten wird durch das gleichzeitige Auftreten von **ch.u.v.m** in beiden Komponenten, die das Verhalten der Nutzer modellieren, modelliert.

Veranschaulichung der Kommunikation im Beispiel:



Message Passing 2

Beispiel

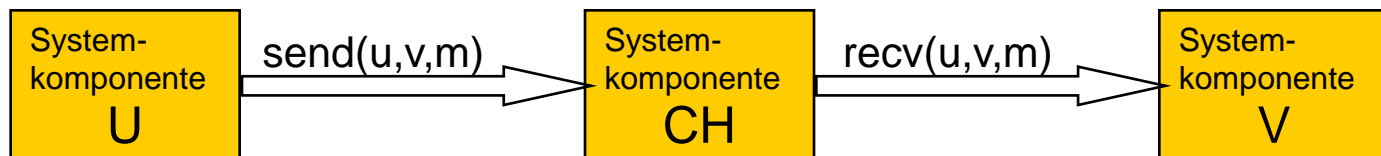
Der Austausch einer Nachricht m von u nach v über einen Kanal, der die Kommunikation verzögert, kann wie folgt modelliert werden:

- ❑ Eine Modellkomponente CH modelliert das Verhalten des Kommunikationskanals (siehe z.B. Folie „Transitionssysteme 2“).
- ❑ Die Modellkomponenten U und V modellieren das jeweilige Verhalten der Nutzer u und v .

Die Kommunikation zwischen u und v wird wie folgt modelliert:

- ❑ **send(u,v,m)** modelliert das Senden von m durch u .
 - ❑ gemeinsames Ereignis von U und CH
- ❑ **recv(u,v,m)** modelliert das Empfangen von m durch v .
 - ❑ gemeinsames Ereignis von CH und V

Veranschaulichung der Kommunikation im Beispiel:



Message Passing 3

Definition

Seien $TS_1 = (S_1, S0_1, E_1, \rightarrow_1)$ und $TS_2 = (S_2, S0_2, E_2, \rightarrow_2)$ zwei Transitionssysteme. Die **message-passing Komposition** von TS_1 und TS_2 ergibt das Transitionssystem $TS = (S, S0, E, \rightarrow)$ wobei

- ☐ $S = S_1 \times S_2$
- ☐ $S0 = S0_1 \times S0_2$
- ☐ $E = E_1 \cup E_2$
- ☐ $\rightarrow = \{ ((s_1, s_2), e, (s_1', s_2')) \in S \times E \times S \mid \begin{aligned} &((s_1, e, s_1') \in \rightarrow_1 \wedge e \in E_1 \setminus E_2 \wedge s_2' = s_2) \\ &\vee ((s_2, e, s_2') \in \rightarrow_2 \wedge e \in E_2 \setminus E_1 \wedge s_1' = s_1) \\ &\vee ((s_1, e, s_1') \in \rightarrow_1 \wedge (s_2, e, s_2') \in \rightarrow_2 \wedge e \in E_2 \cap E_1) \} \end{aligned}$

Beobachtungen

- ☐ Gemeinsame Ereignisse, d.h. Ereignisse in $e \in E_1 \cap E_2$, verursachen synchron einen Zustandsübergang in beiden Komponenten.
- ☐ Lokale Ereignisse, d.h. Ereignisse $e \in (E_1 \setminus E_2) \cup (E_2 \setminus E_1)$, hingegen verursachen nur einen Zustandsübergang in einer Komponente.

Klausur (Erinnerung)

Klausur

- ☐ Datum: 7. März 2012, 8:00-10:30
(bitte in TUCAN nach Änderungen schauen!)
- ☐ Ort: Zuordnung wird später bekannt gegeben.
- ☐ Regeln:
 - ☐ keine Unterlagen, elektronischen Hilfsmittel, ...
 - ☐ 1 beidseitig handbeschriebene DIN A4 Seite ist erlaubt
 - **keine Ausdrücke! keine Kopien!**
 - muss **deutlich lesbar** mit Name und Matrikelnummer **am oberen Rand** auf **beiden** Seiten markiert sein
 - ☐ mitzubringen sind
 - **Personalausweis** oder **Reisepass** mit Lichtbild
 - **Studierendenausweis** im Original
 - **Kugelschreiber/Füller**
 - **Uhr**
 - ☐ Papier wird gestellt (sowohl für Antworten als auch für Notizen)

Rückblick

Einige wesentliche Lernziele dieses Moduls

- ☐ Was ist eine modulare Modellierung?
- ☐ Wie modelliert man nebenläufige Systeme?
 - ☐ Freiheitsgrad: synchrone versus asynchrone Ausführung
 - ☐ Freiheitsgrad: shared-memory versus message-passing
 - ☐ Welche Beziehungen bestehen zwischen den Transitionen der Systemkomponenten und den Transitionen des Gesamtsystems?