Verhaltensorientierte Modellierung 1

Modul 10 (v1.0)

Kanonikvorlesung: Foundations of Computing

Heiko Mantel

MAIS, TU Darmstadt, WS11/12

Motivation

Erinnerung

Modelle sind absichtlich nicht originalgetreu. Bestimmte Aspekte der Realität werden durch ein Modell hervorgehoben.

Modellierung von Informationssystemen

Beispiele für Aspekte die in einer Modellierung betont werden können:

- **Daten**
  - Welche Datentypen verarbeitet das System?
- **Architektur**
  - Wie ist das System intern aufgebaut?
- **Verhalten**
  - Wie kann sich das System verhalten?

Fokus dieses Moduls

- Verhaltensorientierte Modellierung

Übersicht: Modul 10

Verhaltensorientierte Modellierung

- Zustände, Ereignisse und Transitionen
- Transitionssysteme
- Spuren und Historien

Modulare Modellierung

- Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- Modellierung synchroner Ausführung
- Modellierung asynschroner Ausführung

Verhaltensorientierte Modellierung

Man unterscheidet:

Zustände

- Ein Zustand ist eine Momentaufnahme eines Systems während einer Ausführung.

Ereignisse

- Ein Ereignis ist eine Gegebenheit, die einen Zustandsübergang auslösen kann.

Transitionen

- Eine Transition ist ein Übergang von einem Zustand in einen Zustand, der durch ein Ereignis ausgelöst wird.

Spuren und Historien

- Spuren und Historien modellieren mögliche Abläufe eines Informationssystems.

Zustände

Was ist ein Zustand?

- Ein **Zustand** ist eine Momentaufnahme eines Systems während einer Ausführung.

Wie können die Zustände eines Systems modelliert werden?

- Durch eine Menge S. (Vergleiche Vorgehen mit Modul 2)

Wie kann die Menge S definiert werden:

- Die Menge kann unterspezifiziert bleiben, d.h. man gibt nicht an, welche Elemente die Menge enthält.
- Die möglichen Zustände werden durch Symbole modelliert und S wird als Menge dieser Symbole definiert.
- Die möglichen Zustände werden durch andere mathematische Konzepte (z.B. Funktionen) modelliert und S wird als Menge dieser Konzepte definiert.

Beispiel

Sei VAR eine Menge von Programmvariablen und VAL die Menge der möglichen Werte. Dann können die Programmzustände durch den Funktionenraum $VAR \rightarrow VAL$ modelliert werden.

Ereignisse (1)

Was ist ein Ereignis?

- Ein **Ereignis** ist eine Gegebenheit, die einen Zustandsübergang auslösen kann.

Wie können Ereignisse modelliert werden?

- Durch eine Menge E. (vergleiche Vorgehen in Modul 2)

Wie kann die Menge E definiert werden:

- Die Menge kann unterspezifiziert bleiben, die möglichen Ereignisse können durch Symbole modelliert werden, oder die möglichen Ereignisse können durch andere mathematische Konzepte modelliert werden.

Beispiel

Sei CH eine Menge von Kommunikationskanälen, USER eine Menge von Nutzern und MSG eine Menge von Nachrichten. Dann kann das Verschicken von Nachrichten zwischen zwei Nutzern über Kanäle durch folgende Menge modelliert werden:

- $E = \{ (c,u,v,m) \mid c \in CH \ \wedge\ u,v \in USER \ \wedge\ m \in MSG \}$.

Anstatt (c,u,v,m) schreiben wir auch **c.u.v.m**.

Ereignisse (2)

Bei einer konkreten Modellierung zu beachten:

- Ereignisse können unterschiedlich detailliert modelliert werden.

Beispiele: Übertragung einer Nachricht

- **Möglichkeit A:**
  Die gesamte Übertragung wird durch ein Ereignis modelliert.
  - z.B. $E = \{ (c,u,v,m) \mid c \in CH \ \wedge \ u,v \in USER \ \wedge \ m \in MSG \}$.
- **Möglichkeit B:** (detaillierter als A)
  Senden und Empfangen werden durch Ereignisse modelliert.
  - z.B. $E = \{ send(c,u,v,m), recv(c,u,v,m) \mid c \in CH \ \wedge \ u,v \in USER \ \wedge \ m \in MSG \}$.
- **Möglichkeit C:** (detaillierter als A und B)
  Senden, Transport und Empfangen werden getrennt modelliert.
  - z.B. $E = \{ send(c,u,v,m), trans(c,u,v,m), recv(c,u,v,m) \mid c \in CH \ \wedge \ u,v \in USER \ \wedge \ m \in MSG \}$.

Viele weitere Modellierungen sind möglich.

Transitionen/Zustandsübergänge

Was ist eine Transition?

Eine **Transition** ist ein Übergang von einem Zustand in einen Zustand, der durch ein Ereignis ausgelöst wird.

Wie können Transitionen modelliert werden?

Durch Tupel der Form (s,e,s´) wobei

- $e \in E$ das Ereignis ist, das die Transition auslöst;
- $s \in S$ der Zustand vor Geschehen von e ist; und
- $s´ \in S$ der Zustand nach Geschehen von e in s ist.

Modellierung der Transitionen eines Systems

Die möglichen Transitionen des Systems werden durch eine Menge von Tupeln (s,e,s´) modelliert, wobei jedes Tupel eine mögliche Transition modelliert.

Konvention

Transitionen werden auch als **Zustandsübergänge** bezeichnet.

Transitionssysteme 1

Definition

Ein **Transitionssystem** ist eine Tupel (S, S0, E, $\rightarrow$) wobei

- **S** die Menge der Zustände,
- **S0** $\subseteq$ S die Menge der Anfangszustände,
- **E** die Menge der Ereignisse und
- $\rightarrow \subseteq S \times E \times S$ die Transitionsrelation ist.

Notation

Anstatt $(s,e,s') \in \rightarrow$ schreiben wir auch **s-e$\rightarrow$s´** .

Übung: In Modul 0 wurde eine formale Modellierung der Denksportaufgabe „Überquerung eines Flusses" entwickelt. Modelliere das Szenario aus der Denksportaufgabe durch ein Transitionssystem. Geeignete Elemente der Modellierung aus Modul 0 können dabei übernommen werden.

Transitionssysteme 2

Beispiel

Wir modellieren einen Kommunikationskanal, über den Nutzer u Nachrichten aus MSG an den Nutzer v verschicken kann, durch ein Transitionssystem CHANNEL(u,v) = (S, S0, E, $\rightarrow$) wobei

- $S = MSG^*$
- $S0 = \{ () \}$
- $E = \{ send(u,v,m), recv(u,v,m) \mid m \in MSG \}$
- $\rightarrow = \{ (s, send(u,v,m), s') \in S \times E \times S \mid s' = (m).s \}$
  $\cup \{ (s, recv(u,v,m), s') \in S \times E \times S \mid s = s'.(m) \}$

Konkatenation von Folgen (siehe Folie 13)

In dieser Modellierung ist ein Zustand eine Folge von Nachrichten. Der modellierte Kommunikationskanal hat folgende Eigenschaften:

- **FIFO**: Wird eine Nachricht m1 vor einer Nachricht m2 verschickt, so wird m2 nie vor m1 empfangen.
- **zuverlässige Übertragung**: Es gehen keine Nachrichten verloren und Nachrichten, die empfangen werden, wurden auch verschickt.

Es besteht aber **keine** Garantie, dass eine Nachricht, die verschickt wird, auch empfangen wird. (Wie begründet sich das im Modell?)

Spuren 1

Definition

Eine **Spur** (engl.: **trace**) ist eine **endliche** Folge von Ereignissen und Zuständen. Wenn die Folge nur Ereignisse enthält, so nennen wir die Spur auch **Ereignisspur**. Enthält die Folge nur Zustände, so nennen wir die Spur auch **Zustandsspur**.

Beispiel

Die Ereignisspur

- (ch1.u1.u2.m1, ch2.u2.u3.m1, ch1.u1.u2.m2)

kann wie folgt interpretiert werden:

- Nutzer u1 schickt die Nachricht m1 über den Kanal ch1 an u2.
- Nutzer u2 leitet m1 über den Kanal ch2 an Nutzer u3 weiter.
- Nutzer u1 schickt die Nachricht m2 über den Kanal ch1 an u2.

Spuren 2

Beispiel (Fortsetzung)

Die Spur

- (s0, ch1.u1.u2.m1, s1, ch2.u2.u3.m1, s2, ch1.u1.u2.m2, s3)

kann wie folgt interpretiert werden:

- Nutzer u1 schickt die Nachricht m1 im Zustand s0 über den Kanal ch1 an u2 und dieses resultiert im Zustand s1.
- Nutzer u2 leitet m1 im Zustand s1 über den Kanal ch2 an u3 weiter und dieses resultiert im Zustand s2.
- Nutzer u1 schickt die Nachricht m2 im Zustand s2 über den Kanal ch1 an u2 und dieses resultiert im Zustand s3.

Beobachtung

In obigem Beispiel werden Zustände durch Symbole benannt. Die Struktur von Zuständen wird nicht detailliert modelliert.

Übung: Modelliere obiges Systemverhalten unter Verwendung von strukturierten Zuständen, die für jeden Nutzer angeben, welche Nachrichten bisher gesendet und empfangen wurden.

Spuren 3

Definition

Die **Konkatenation** zweier Spuren t1, t2 $\in$ (S$\cup$E)* ist die wie folgt rekursiv definierte Spur **t1.t2** $\in$ (S$\cup$E)*:

- t1.t2 = t1 wenn t2 = () und
- t1.t2 = t2 wenn t1 = () und
- t1.t2 = (t1.t2´, q) wenn t1 $\neq$ () und t2 = (t2´,q).

Beispiel

Sei t = (s0, ch1.u1.u2.m1, s1, ch2.u2.u3.m1, s2, ch1.u1.u2.m2, s3). Dann gilt

- (ch1.u1.u2.m1, ch2.u2.u3.m1 , ch1.u1.u2.m2)
  = (ch1.u1.u2.m1, ch2.u2.u3.m1) . (ch1.u1.u2.m2)

Spuren 4

Definition

Die **Projektion** einer Spur $t \in (S \cup E)^*$ nach einer Menge $Q \subseteq S \cup E$ ist die wie folgt rekursiv definierte Spur **$t \upharpoonright Q$** $\in (S \cup E)^*$:

- $t \upharpoonright Q = ()$ wenn $t = ()$;
- $t \upharpoonright Q = (t' \upharpoonright Q, q)$ wenn $t = t'.(q)$ und $q \in Q$; und
- $t \upharpoonright Q = t' \upharpoonright Q$ wenn $t = t'.(q)$ und $q \notin Q$.

Beispiel

Für $Q = \{ ch.u.v.m \in E \mid u = u1 \}$ gilt:

- (s0, ch1.u1.u2.m1, s1, ch2.u2.u3.m1, s2, ch1.u1.u2.m2, s3) $\upharpoonright$ Q = (ch1.u1.u2.m1, ch1.u1.u2.m2)

Für $Q' = \{ ch.u.v.m \in E \mid m = m1 \}$ gilt:

- (s0, ch1.u1.u2.m1, s1, ch2.u2.u3.m1, s2, ch1.u1.u2.m2, s3) $\upharpoonright$ Q´ = (ch1.u1.u2.m1, ch1.u2.u3.m1)

Für $Q'' = \{ ch.u.v.m \in E \mid m = m2 \}$ gilt:

- (s0, ch1.u1.u2.m1, s1, ch2.u2.u3.m1, s2, ch1.u1.u2.m2, s3) $\upharpoonright$ Q´´ = (ch1.u1.u2.m2)

Spuren von Transitionssystemen 1

Definition

Sei TS = (S, S0, E, →) ein Transitionssystem. **Die durch TS induzierte Menge von Spuren Traces(TS)** ⊆ (S∪E)* ist die kleinste Menge, so dass

- (s) ∈ Traces(TS) wenn s ∈ S0 und
- t.(s,e,s´) ∈ Traces(TS) wenn t.(s) ∈ Traces(TS) und s-e→s´.

Beispiel

Für die Modellierung auf Folie „Transitionssysteme 2“ gilt z.B.

- (()) ∈ Traces(CHANNEL(u,v))
- ((), send(u,v,m1), (m1), send(u,v,m2), (m2,m1), recv(u,v,m1), (m2)) ∈ Traces(CHANNEL(u,v))

Spuren von Transitionssystemen 2

Definition

Die durch ein Transitionssystem TS = (S, S0, E, $\rightarrow$) **induzierte Menge von Zustandsspuren S-Traces(TS)** $\subseteq S^*$ und die durch TS **induzierte Menge von Ereignisspuren E-Traces(TS)** $\subseteq E^*$ sind wie folgt definiert

- S-Traces(TS) = { t↾S | t $\in$ Traces(TS) }
- E-Traces(TS) = { t↾E | t $\in$ Traces(TS) }

Beispiel

Für die Modellierung auf Folie „Transitionssysteme 2“ gilt z.B.

- ((\)) $\in$ S-Traces(CHANNEL(u,v))
- ((), (m1), (m2,m1), (m2)) $\in$ S-Traces(CHANNEL(u,v))

und

- () $\in$ E-Traces(CHANNEL(u,v))
- (send(u,v,m1), send(u,v,m2), recv(u,v,m1)) $\in$ E-Traces(CHANNEL(u,v))

Historien

Definition

Eine **Historie** (engl.: **history**) ist eine **unendliche** Folge von Ereignissen und Zuständen. Wenn die Folge nur Ereignisse enthält, so nennen wir die Historie auch **Ereignishistorie**. Enthält die Folge nur Zustände, so nennen wir die Historie auch **Zustandshistorie**.

Bemerkung

Historien können zur Modellierung von nicht-terminierenden Ausführungen (unendliche Ausführungen) aber auch von terminierenden Ausführungen eingesetzt werden.

- siehe übernächste Folie

Historien von Transitionssystemen 1

Definition

Sei TS = (S, S0, E, $\rightarrow$) ein Transitionssystem. **Die durch TS induzierte Menge von Historien Hist(TS)** $\subseteq (S \cup E)^\infty$ ist

- Hist(TS) = { h: $\mathbb{N} \rightarrow (S \cup E)$ | $h(0) \in S0$ $\wedge \forall n \in \mathbb{N}$: $h(2{*}n)$-$h(2{*}n+1) \rightarrow h(2{*}n+2)$ }

Definition

Die durch ein Transitionssystem TS = (S, S0, E, $\rightarrow$) **induzierte Menge von Zustandshistorien S-Hist(TS)** $\subseteq S^\infty$ und die durch TS **induzierte Menge von Ereignishistorien E-Hist(TS)** $\subseteq E^\infty$ sind wie folgt definiert

- S-Hist(TS) = { h: $\mathbb{N} \rightarrow S$ | $\exists h' \in$Hist(TS): $\forall n \in \mathbb{N}$: $h(n) = h'(2{*}n)$ }
- E-Hist(TS) = { h: $\mathbb{N} \rightarrow E$ | $\exists h' \in$Hist(TS): $\forall n \in \mathbb{N}$: $h(n) = h'(2{*}n+1)$ }

Historien von Transitionssystemen 2

Bemerkung

Historien eignen sich zur Modellierung von Ausführungen, die nicht terminieren. Sie können aber auch zur Modellierung von Ausführungen eingesetzt werden, die nach einer endlichen Zahl von Schritten terminieren. Dafür kann man z.B. wie folgt vorgehen:

- Man führt ein neues Ereignis $\surd$ ein, das Terminierung modelliert.
- Wenn eine Ausführung terminiert, das heißt wenn das Ereignis $\surd$ an der Stelle n zum ersten Mal auftritt ($hist(n) = \surd$), dann gilt für alle $n' > n$: $hist(n') = hist(m)$ oder $hist(n') = \surd$. Das heißt, andere Ereignisse als $\surd$ können nicht mehr auftreten. Der letzte Zustand in der Historie steht an der Stelle $n-1$. Die Interpretation hiervon ist, dass der Zustand unverändert bleibt..

Übersicht: Modul 10

Verhaltensorientierte Modellierung

- Zustände, Ereignisse und Transitionen
- Transitionssysteme
- Spuren und Historien

Modulare Modellierung

- Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- Modellierung synchroner Ausführung
- Modellierung asynschroner Ausführung

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

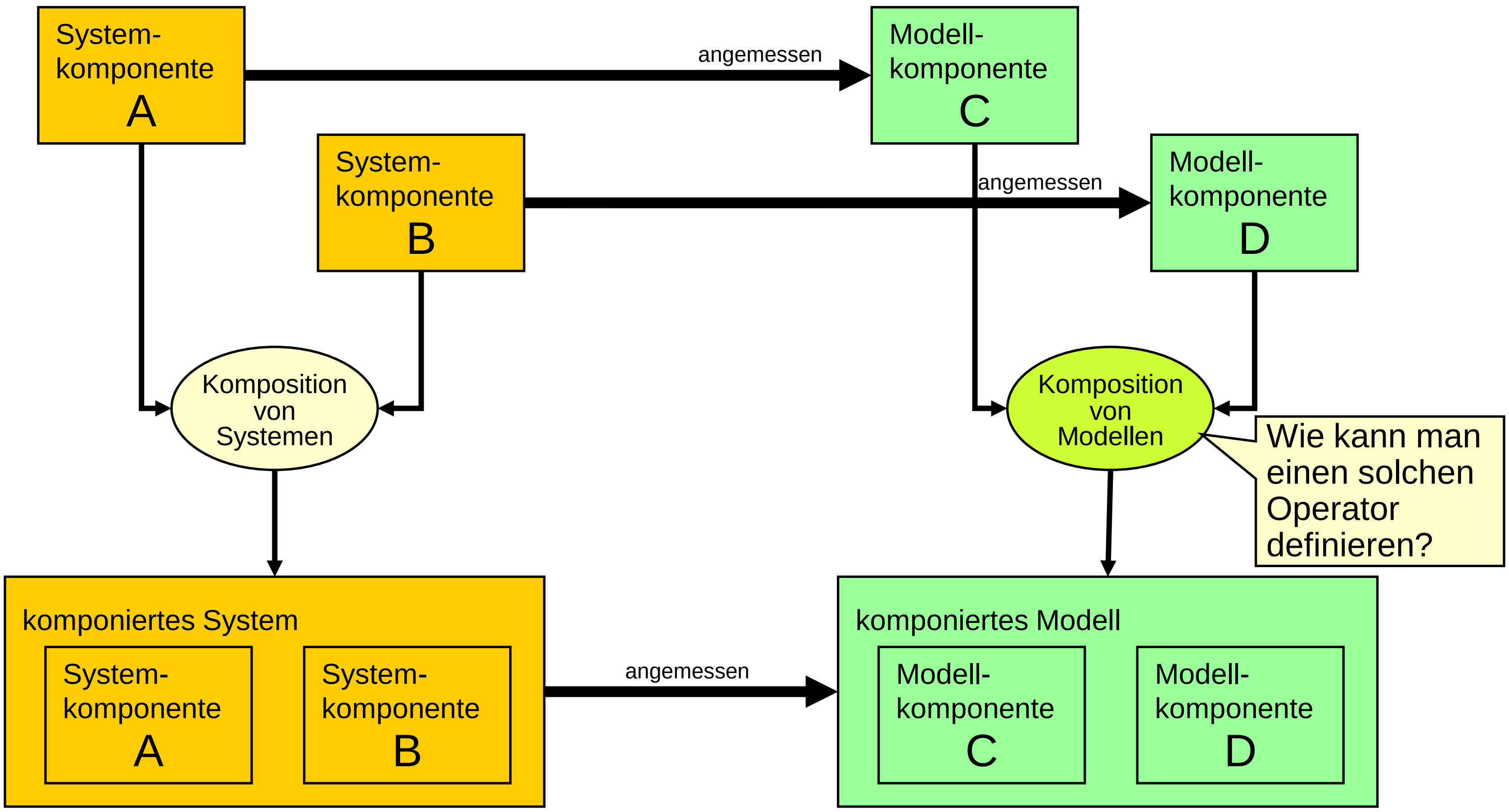
- Es sollte möglich sein, die Modularität eines nebenläufigen Systems bei der Modellierung auszunutzen.
- Vorteile
  - Die Modellierung wird strukturiert.
  - Die Komplexität des Modellierens wird reduziert.

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 Modul aufgehoben.

Übersicht: Modul 10

Verhaltensorientierte Modellierung

- Zustände, Ereignisse und Transitionen
- Transitionssysteme
- Spuren und Historien

Modulare Modellierung

- Komposition von Modellen von Komponenten

Nebenläufige Ausführung

- Modellierung synchroner Ausführung
- Modellierung asynschroner Ausführung

Synchrone Nebenläufigkeit 1

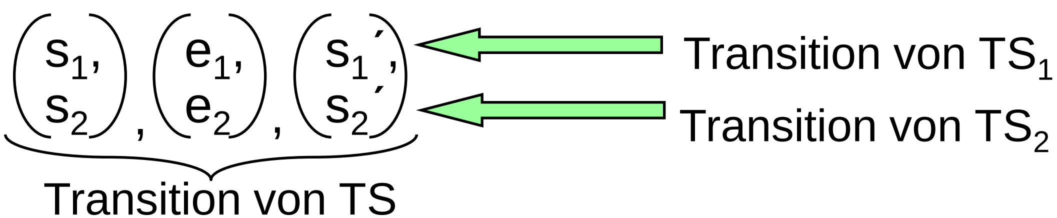
Definition

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:



Asynchrone 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 **asynchrone Produktkomposition** von TS_1 und TS_2 ist das Transitionssystem $TS = (S, S0, E, \rightarrow)$ wobei

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

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.

Nebenläufigkeit

… wird im nächsten Modul fortgesetzt …

Rückblick

Einige wesentliche Lernziele dieses Moduls

- Fähigkeit das Verhalten von Informationssystemen zu modellieren
  - Unterscheidung von Zuständen, Ereignissen und Transitionen in der Modellierung
- Verständnis der Spuren und Historien eines Transitionssystems
  - Wie sind die Menge der Spuren und Historien definiert?
- Was ist eine modulare Modellierung?
- Wie modelliert man nebenläufige Systeme?
  - Freiheitsgrad: synchrone versus asynchrone Ausführung
  - Welche Beziehungen bestehen zwischen den Transitionen der Systemkomponenten und den Transitionen des Gesamtsystems?