Hauptseminar | Wintersemester 2004/05

Semantik der UML V2.0

Manfred Broy

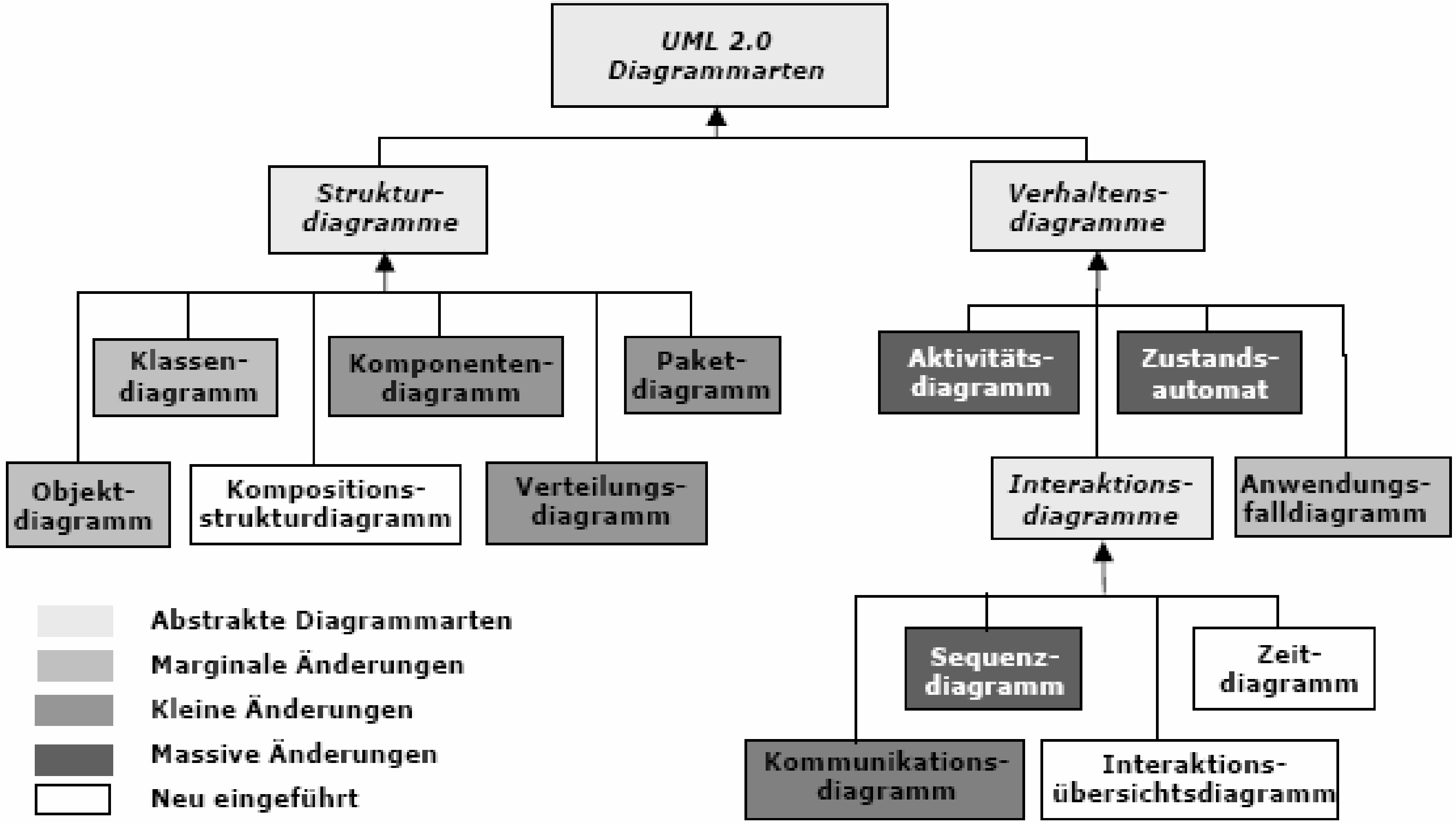
Thema:	UML 2.0 Interaktionen: Semantik und Verfeinerung
Datum:	14.01.2005
Betreuer	Graubmann, Peter
Referent:	Xie, Fei
Email:	xie@in.tum.de

Gliederung

- Geschichte und Diagrammarten der UML 2.0
- Notation der Sequenzdiagramme
  - Basis-Sequenzdiagramme
  - Strukturierte Sequenzdiagramme
- Syntax und Semantik der Interaktionen
- Verbesserung der Negation
- Zusammenfassung

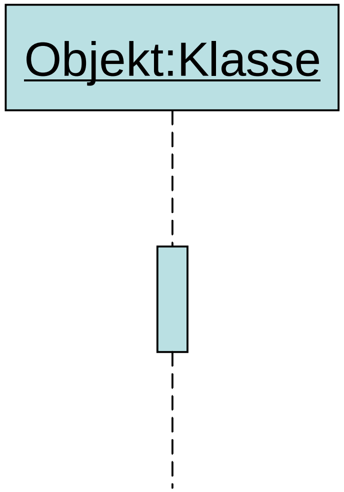
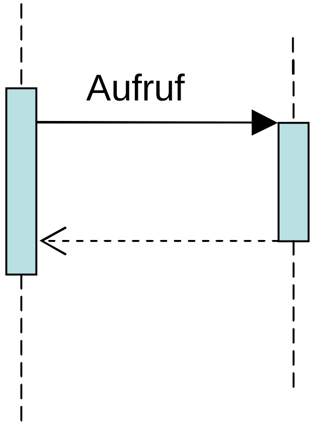
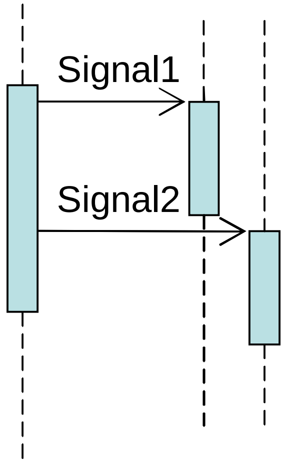
Geschichte und Diagrammarten

- Diagrammarten der UML 2.0



Notation der Sequenzdiagramme

F Notation der Basis-Sequenzdiagramme

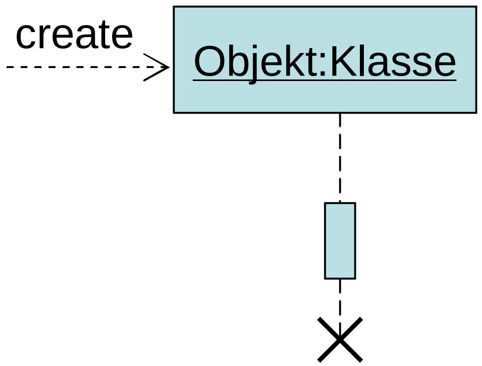
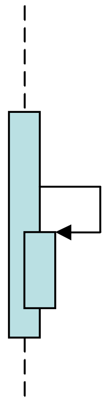
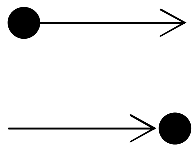
Inhalt	Klasse, Objekt	Synchrone Nachricht (Aufruf)	Asynchrone Nachricht
Symbole			

Notation der Sequenzdiagramme

- Die Syntax der Nachrichten ist wie folgend definiert.
  - messageident ::= [attribute =] signal-or-operation-name [(arguments)][: return-value] | '*'
  - arguments ::= argument [, arguments]
  - argument ::= [parameter-name=]argument-value | attribute= out-parameter-name [:argument-value]| -
- Beispiele der Syntax
  - *mymessage(14, - , 3.14, "hello")*
  - *v=mymsg(16, variab):96*
  - *mymsg(myint=16)*

Notation der Sequenzdiagramme

- **F** Notation der Basis-Sequenzdiagramme

Inhalt	Objekt erzeugen, zerstören	Rekursion	Gefundene & verlorene Nachrichten
Symbole			

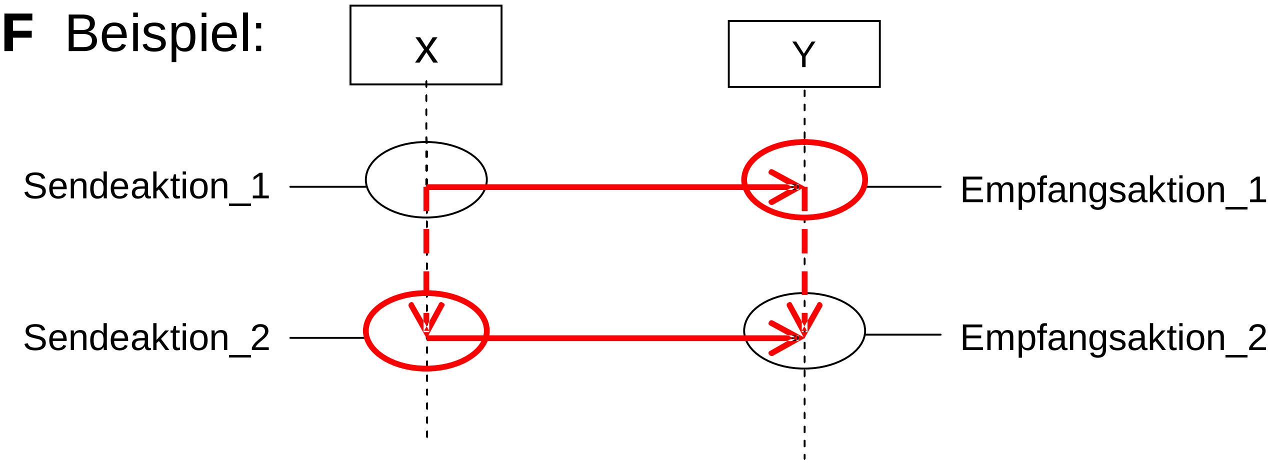
Notation der Sequenzdiagramme

- Die Regeln der zeitlichen Abläufe

  1. Die Sendaktion einer Nachricht findet stets vor der Empfangsaktion für diese Nachricht statt.

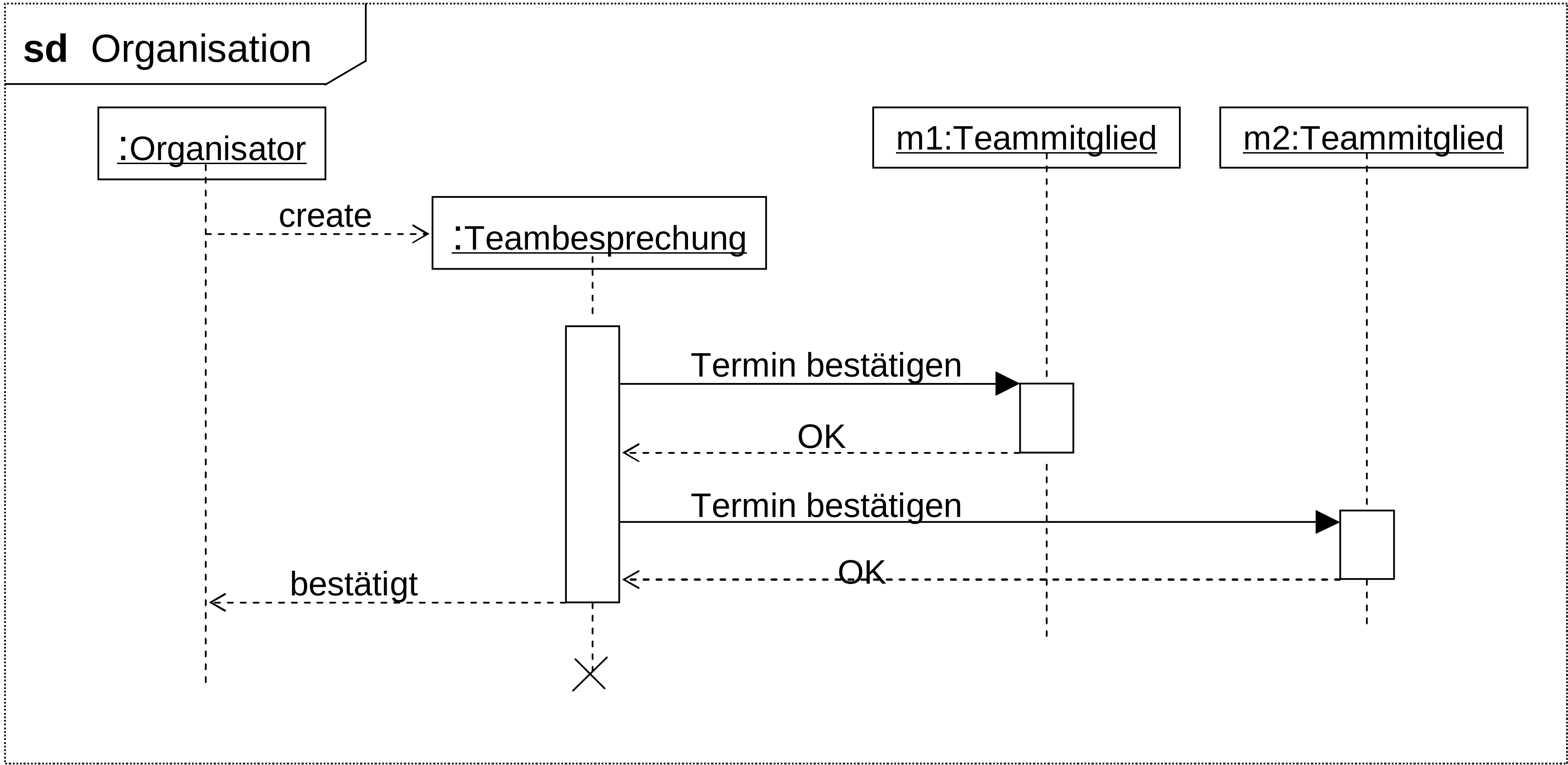
  2. Die Abfolge von Sende- und Empfangsaktionen auf der gleichen Lebenslinie ist total geordnet.

- Beispiel:



Notation der Sequenzdiagramme

- Beispiel des Basis- Sequenzdiagramms



Notation der Sequenzdiagramme

F Strukturierte Sequenzdiagramme

- Motivation
- Zwei Komponenten

1. Referenzen
   - Einen Verweis auf ein anderes Sequenzdiagramm
2. Operatoren
   - Ähnliche Abläufe innerhalb eines Sequenzdiagramms darzustellen
   - Namen: *seq, alt, opt, break, par, strict, loop, region, neg, assert, ignore, consider*

Syntax und Semantik der Interaktionen

- Abstrakte Syntax der Interaktionen

Interaction ::= Basic

| CombinedFragment

CombinedFragment ::= opt(Interaction)

|strict(Interaction, Interaction)

| seq(Interaction, Interaction)

| par(Interaction, Interaction)

| loop(Nat, (Nat|∞), Interaction)

| ignore(Messages, Interaction)

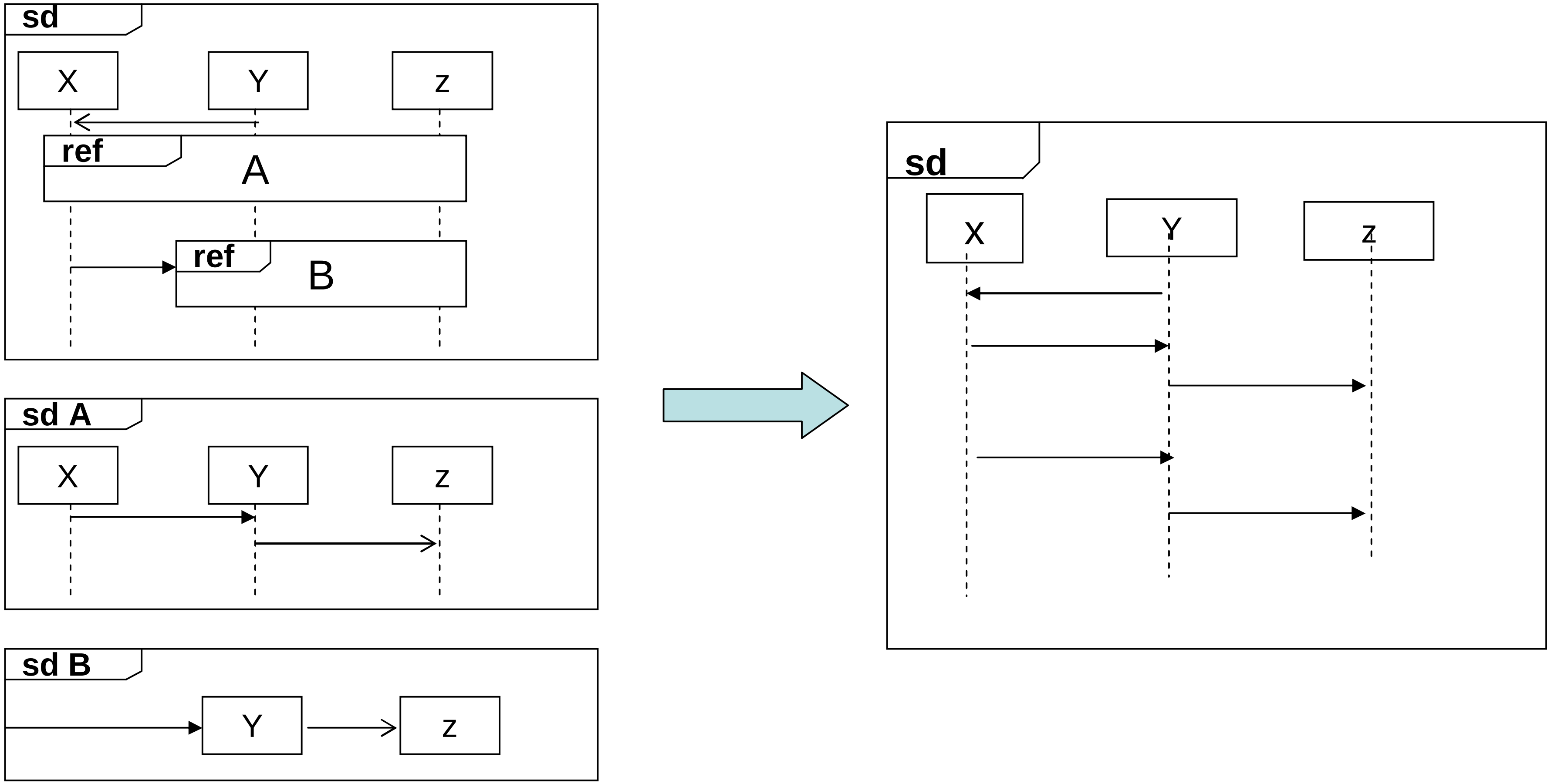
| alt(Interaction, Interaction)

| neg(Interaction)

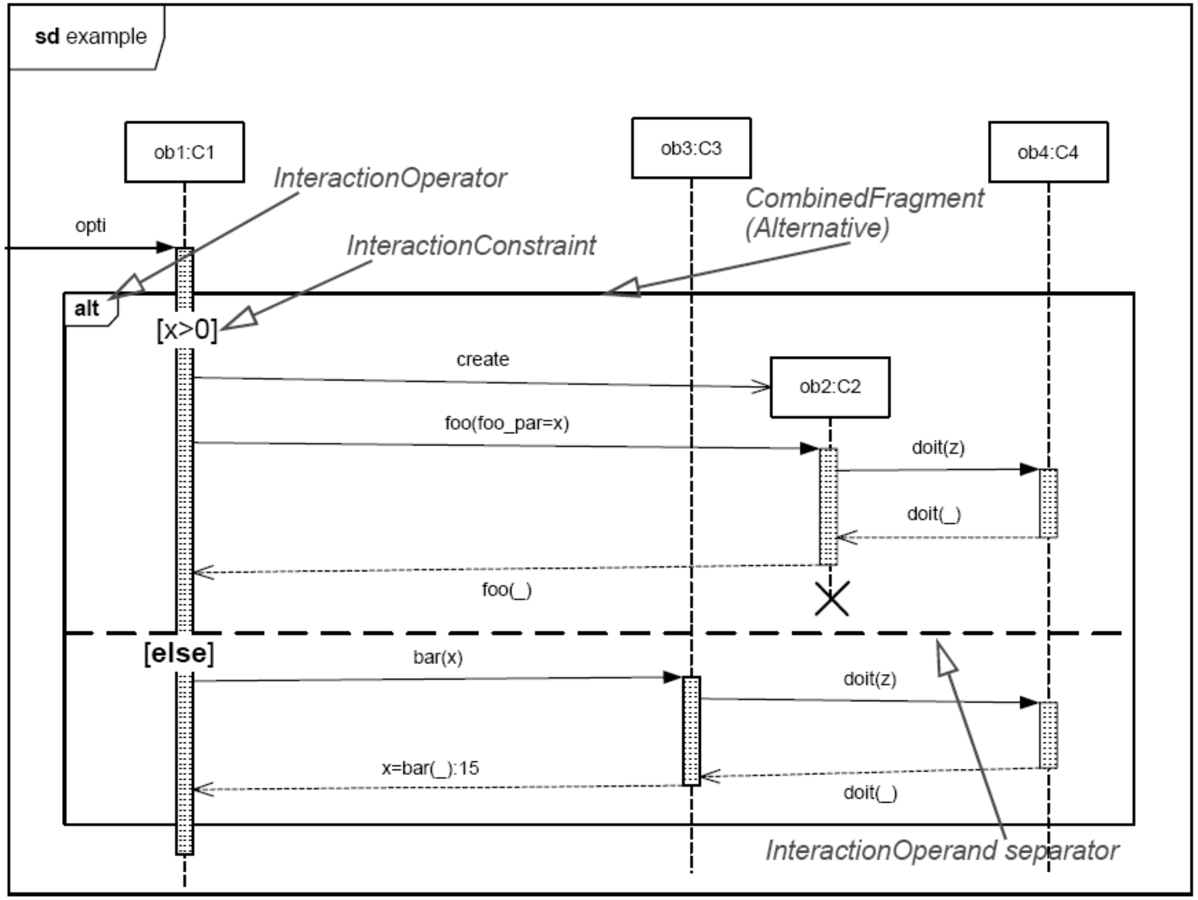
| assert(Interaction)

Syntax und Semantik der Interaktionen

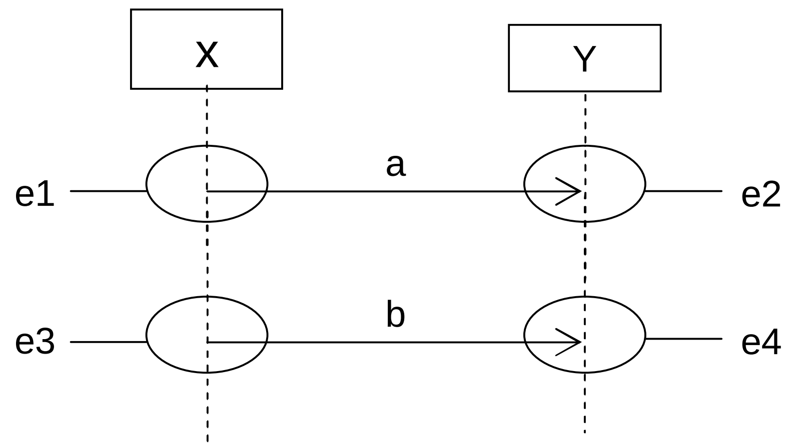
- Beispiel der Sequenzdiagramme (ref)



- Beispiel der Sequenzdiagramme (alt)



Syntax und Semantik der Interaktionen



Mögliche Abläufe: e1→e2→e3→e4 oder e1→e3→e2→e4

⇩

partiell geordnete, bezeichnete Multimenge (*pomset*)

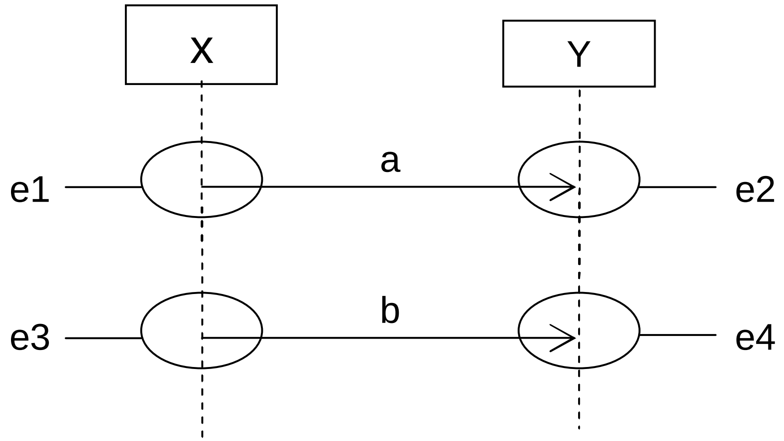
Syntax und Semantik der Interaktionen

F partiell geordnete, bezeichnete Multimenge (*pomset*)

- Eine Klasse $[(X, \leqslant_x, \lambda_x)]$
- Eine Spur: eine *pomset* und total geordnet
- *lin(p)*: alle mögliche Linearisierung für eine *pomset* p
- skip: Ein leere *pomset*, das durch (Φ,Φ,Φ) repräsentiert, ist durch ε gekennzeichnet.
- Drei Operationen. Sei $p = [(X, \leqslant_x, \lambda_x)]$, $q = [(Y, \leqslant_y, \lambda_Y)]$ und $X \cap Y = \Phi$
  1. Nebenläufigkeit: **p||q**, ist die Klasse $[(X \cup Y, \leqslant_x \cup \leqslant_y, \lambda_x \cup \lambda_Y)]$
  2. Konkatenation: **p;q**, ist die Klasse $[(X \cup Y, (\leqslant_x \cup \leqslant_y \cup (X \times Y))^*, \lambda_x \cup \lambda_Y)]$
  3. [illegible] -Konaktenation: p ;[illegible] q , ist die Klasse

     $[(X \cup Y, (\leqslant_x \cup \leqslant_y \cup \{(x,y) \in X \times Y \mid \lambda_X(x)$ [illegible] $\lambda_Y(y)\})^*, \lambda_x \cup \lambda_Y)]$

Syntax und Semantik der Interaktionen



Mögliche Abläufe: e1→e2→e3→e4 oder e1→e3→e2→e4

⇓

[({e1,e2,e3,e4},

{(e1,e1),(e1,e2),(e1,e3)(e2,e2),(e2,e4)(e3,e3),(e3,e4)(e4,e4)},

{e1→snd(x,y,a}, e2→rcv(x,y,a), e3→snd(x,y,b), e4→rcv(x,y,b)})]

Die Semantik des positiven Fragments

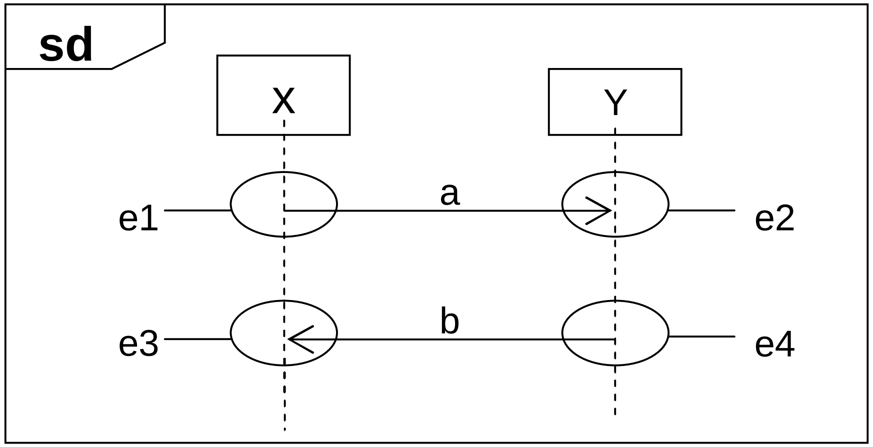
- Interaktionen ohne Auftreten der Negation (neg) und Assertion (assert)
- Die positive Befriedigung: $t \models_p S$

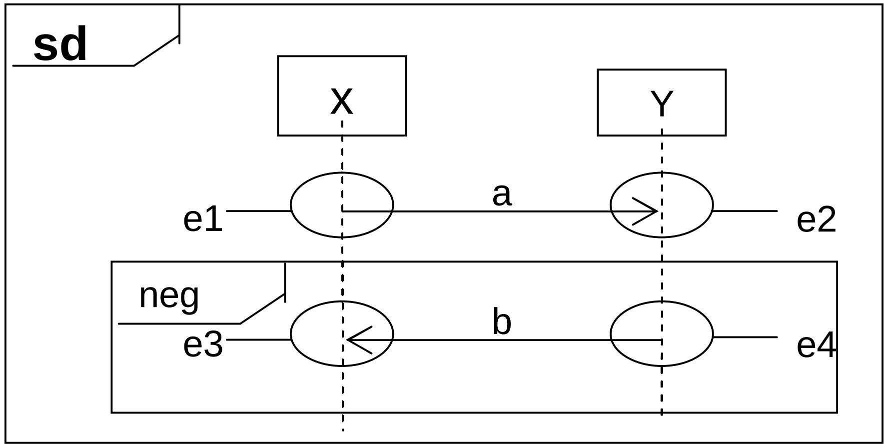
$$t \models_p B \quad \text{if } t \in \mathit{lin}(B)$$
$$t \models_p \mathsf{strict}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t = t_1 \,;\, t_2 \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$$
$$t \models_p \mathsf{seq}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t \in \mathit{lin}(t_1 \,;_{\asymp}\, t_2) \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$$
$$t \models_p \mathsf{par}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t \in \mathit{lin}(t_1 \parallel t_2) \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$$
$$t \models_p \mathsf{loop}(0, 0, S) \quad \text{if } t = \varepsilon$$
$$t \models_p \mathsf{loop}(0, n+1, S) \quad \text{if } t = \varepsilon \vee t \models_p \mathsf{seq}(S, \mathsf{loop}(0, n, S))$$
$$t \models_p \mathsf{loop}(m+1, n+1, S) \quad \text{if } t \models_p \mathsf{seq}(S, \mathsf{loop}(m, n, S))$$
$$t \models_p \mathsf{loop}(m, \infty, S) \quad \text{if } \exists n \geq m \,.\, t \models_p \mathsf{loop}(m, n, S)$$
$$t \models_p \mathsf{ignore}(M, S) \quad \text{if } \exists t_1 \,.\, t_1 \in \mathit{filter}(M)(t) \wedge t_1 \models_p S$$
$$t \models_p \mathsf{alt}(S_1, S_2) \quad \text{if } t \models_p S_1 \vee t \models_p S_2$$

Syntax und Semantik der Interaktionen

- Definitionen der Negation
  - Störrle's Definition für Negation:
    1. not the [valid] traces of s
    2. Anything but the [valid] traces of s
    3. The negative traces of S to be the positive traces for neg(s)
  - Haugen und Stølen's Definition für Negation
    1. the valid traces of neg(S) consists just of the empty trace
- Spuren(Eng. Traces)
  - Drei artige Spuren:
    1. positive (Eng. valid) Spuren
    2. Negative (Eng. invalid) Spuren
    3. ergebnislos (Eng. inconclusive) Spuren

Syntax und Semantik der Interaktionen





Positive Spuren:	e1, e2, e4, e3	e1, e2
Negative Spuren:	NULL	e1,e2,e4,e3
Inconclusive Spuren:	alle andere Spuren	alle andere Spuren

Syntax und Semantik der Interaktionen

F Die Semantik des negativen Fragments

$$t \models_{\mathrm{p}} \mathsf{neg}(S) \quad \text{if } t = \varepsilon$$
$$t \models_{\mathrm{p}} \mathsf{assert}(S) \quad \text{if } t \models_{\mathrm{p}} S$$

$$t \models_{\mathrm{n}} \mathsf{strict}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t = t_1 \,;\, t_2 \wedge (t_1 \models_{\mathrm{n}} S_1 \vee (t_1 \models_{\mathrm{p}} S_1 \wedge t_2 \models_{\mathrm{n}} S_2))$$
$$t \models_{\mathrm{n}} \mathsf{seq}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t \in \mathit{lin}(t_1 \,;_{\asymp}\, t_2) \wedge (t_1 \models_{\mathrm{n}} S_1 \vee (t_1 \models_{\mathrm{p}} S_1 \wedge t_2 \models_{\mathrm{n}} S_2))$$
$$t \models_{\mathrm{n}} \mathsf{par}(S_1, S_2) \quad \text{if } \exists t_1, t_2 \,.\, t \in \mathit{lin}(t_1 \parallel t_2) \wedge ((t_1 \models_{\mathrm{n}} S_1 \wedge t_2 \models_{\mathrm{n}} S_2) \vee (t_1 \models_{\mathrm{n}} S_1 \wedge t_2 \models_{\mathrm{p}} S_2) \vee (t_1 \models_{\mathrm{p}} S_1 \wedge t_2 \models_{\mathrm{n}} S_2))$$
$$t \models_{\mathrm{n}} \mathsf{loop}(0, n+1, S) \quad \text{if } t \models_{\mathrm{n}} \mathsf{seq}(S, \mathsf{loop}(0, n, S))$$
$$t \models_{\mathrm{n}} \mathsf{loop}(m+1, n+1, S) \quad \text{if } t \models_{\mathrm{n}} \mathsf{seq}(S, \mathsf{loop}(m, n, S))$$
$$t \models_{\mathrm{n}} \mathsf{loop}(m, \infty, S) \quad \text{if } \exists n \geq m \,.\, t \models_{\mathrm{n}} \mathsf{loop}(m, n, S)$$
$$t \models_{\mathrm{n}} \mathsf{ignore}(M, S) \quad \text{if } \exists t_1 \,.\, t_1 \in \mathit{filter}(M)(t) \wedge t_1 \models_{\mathrm{n}} S$$
$$t \models_{\mathrm{n}} \mathsf{alt}(S_1, S_2) \quad \text{if } t \models_{\mathrm{n}} S_1 \wedge t \models_{\mathrm{n}} S_2$$
$$t \models_{\mathrm{n}} \mathsf{neg}(S) \quad \text{if } t \neq \varepsilon \wedge t \models_{\mathrm{p}} S$$
$$t \models_{\mathrm{n}} \mathsf{assert}(S) \quad \text{if } t \not\models_{\mathrm{p}} S$$

Verbesserung der Negation

F Die Definition für die σ Funktion

$\sigma(B) = B$

$\sigma(\text{strict}(S1, S2)) = \text{strict}(\sigma(S1), \sigma(S2))$

$\sigma(\text{par}(S1, S2)) = \text{par}(\sigma(S1), \sigma(S2))$

$\sigma(\text{loop}(m, \bar{n}, S)) = \text{loop}(m, \bar{n}, \sigma(S))$

$\sigma(\text{ignore}(M, S)) = \text{ignore}(M, \sigma(S))$

$\sigma(\text{alt}(S1, S2)) = \text{alt}(\sigma(S1), \sigma(S2))$

$\sigma(\text{neg}(S)) = \text{skip}$

$\sigma(\text{assert}(S)) = \sigma(S)$

$t \models_p B \quad \text{if } t \in lin(B)$

$t \models_p \text{strict}(S_1, S_2) \quad \text{if } \exists t_1, t_2 . t = t_1 ; t_2 \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$

$t \models_p \text{seq}(S_1, S_2) \quad \text{if } \exists t_1, t_2 . t \in lin(t_1 ;_{\lessgtr} t_2) \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$

$t \models_p \text{par}(S_1, S_2) \quad \text{if } \exists t_1, t_2 . t \in lin(t_1 \parallel t_2) \wedge t_1 \models_p S_1 \wedge t_2 \models_p S_2$

$t \models_p \text{loop}(0, 0, S) \quad \text{if } t = \varepsilon$

$t \models_p \text{loop}(0, n+1, S) \quad \text{if } t = \varepsilon \vee t \models_p \text{seq}(S, \text{loop}(0, n, S))$

$t \models_p \text{loop}(m+1, n+1, S) \quad \text{if } t \models_p \text{seq}(S, \text{loop}(m, n, S))$

$t \models_p \text{loop}(m, \infty, S) \quad \text{if } \exists n \geq m . t \models_p \text{loop}(m, n, S)$

$t \models_p \text{ignore}(M, S) \quad \text{if } \exists t_1 . t_1 \in filter(M)(t) \wedge t_1 \models_p S$

$t \models_p \text{alt}(S_1, S_2) \quad \text{if } t \models_p S_1 \vee t \models_p S_2$

$t \models p\ S$ falls $t \models p\ \sigma(S)$

Verbesserung der Negation

- Interaktion in den klassischen Sinnen

  Die Operatoren neg(-) und assert(-) werden durch not(-) ersetzt.

  $t \models_{\mathrm{p}} \mathsf{not}(S) \quad if\, t \not\models_{\mathrm{p}} S$

  Any = ignore(M, skip)
  None = not(Any)
  and(S1, S2) = not(all(not(S1),not(S2)))

- Die Definition für die ν Funktion

$$
\begin{aligned}
\nu(B) &= \mathsf{None} \\
\nu(\mathsf{strict}(S_1, S_2)) &= \mathsf{alt}(\mathsf{strict}(\nu(S_1), \mathsf{Any}), \mathsf{strict}(\sigma(S_1), \nu(S_2))) \\
\nu(\mathsf{seq}(S_1, S_2)) &= \mathsf{alt}(\mathsf{seq}(\nu(S_1), \mathsf{Any}), \mathsf{seq}(\sigma(S_1), \nu(S_2))) \\
\nu(\mathsf{par}(S_1, S_2)) &= \mathsf{alt}(\mathsf{par}(\nu(S_1), \nu(S_2)), \mathsf{par}(\nu(S_1), \sigma(S_2)), \mathsf{par}(\sigma(S_1), \nu(S_2))) \\
\nu(\mathsf{loop}(m, \overline{n}, S)) &= \mathsf{and}(\mathsf{loop}(m, \overline{n}, \nu(S)), \mathsf{not}(\mathsf{skip})) \\
\nu(\mathsf{ignore}(M, S)) &= \mathsf{ignore}(M, \nu(S)) \\
\nu(\mathsf{alt}(S_1, S_2)) &= \mathsf{and}(\nu(S_1), \nu(S_2)) \\
\nu(\mathsf{neg}(S)) &= \mathsf{and}(\sigma(S), \mathsf{not}(\mathsf{skip})) \\
\nu(\mathsf{assert}(S)) &= \mathsf{not}(\sigma(S))
\end{aligned}
$$

Verbesserung der Negation

F Zusammengefasst haben wir zwei Resultaten:

- $t \models_p S$ falls $t \models_p \sigma(S)$
- $t \models_n S$ falls $t \models_p \nu(S)$

Zusammenfassung

- F Im Dokument UML 2.0 Spezifikation ist die Features der Interaktionssprache nicht deutlich spezifiziert, z.B. Negation.
- F Wir haben zuerst die Syntax und Semantik der Interaktion formal beschrieben.
- F Besonders haben wir die positiven, negativen und inconclusiven Abläufe für Interaktionen klassifiziert.
- F Schließlich versuchen wir durch die Einführung von zwei Funktionen, σ und ע, das Testen für die negative Befriedigung zu vermeiden.

Vielen Dank

für Ihre Aufmerksamkeit!