

## 3.5 Interaktions-Sicht

---

3.5.1 Interaktion

3.5.2 Kollaborationsdiagramme

3.5.3 Sequenzdiagramme

3.5.4 Kollaborationen

3.5.5 Entwurfsmuster und Rahmenwerke

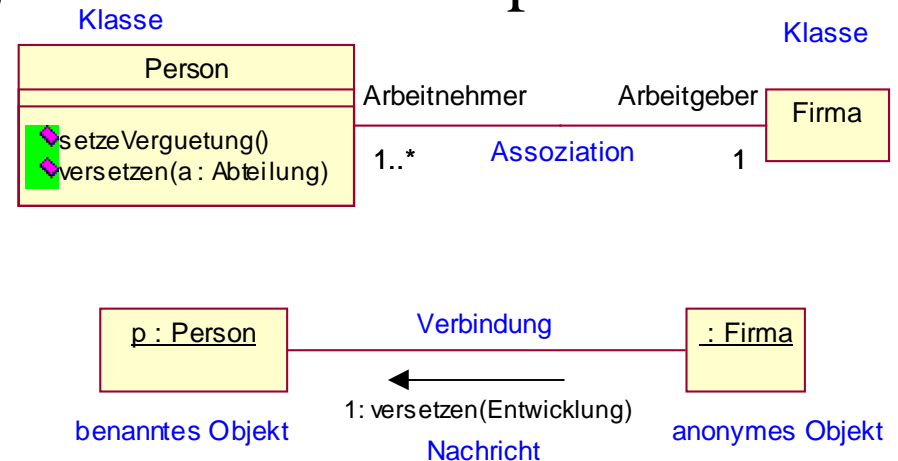
## 3.5.1 Interaktion (Interaction)

---

- *Interaktion* (Interaction) = Verhalten, das sich aus der Zusammenarbeit einer Menge von Objekten durch Austausch von Nachrichten ergibt
- Objekte sind konkret oder Prototypen
- Interaktion findet man
  - bei Realisierung von Anwendungsfällen
  - im Kontext einer Operation/Klasse

# Verbindung (Link)

- *Verbindung* (Link) zweier Objekte repräsentiert Instanz einer Assoziation
  - Verbindungen können Namen, Rollen und Navigationsrichtungen tragen, jedoch *keine* Multiplizität
- Über eine Verbindung kann eine *Nachricht* geschickt werden



# Verbindung (Forts.)

---

- Stereotypen für die Rollen der Verbindung spezifizieren welcher Art die Sichtbarkeit ist
  - `<<parameter>>` Objekt ist Parameter
  - `<<local>>`, `<<global>>`  
Objekt ist lokale/globale Variable
  - `<<self>>` Aufruf einer eigenen Operation
- Solche Verbindungen heissen *vorübergehend* (transient)
  - `<<association>>` ist Standard

# Nachricht (Message)

---

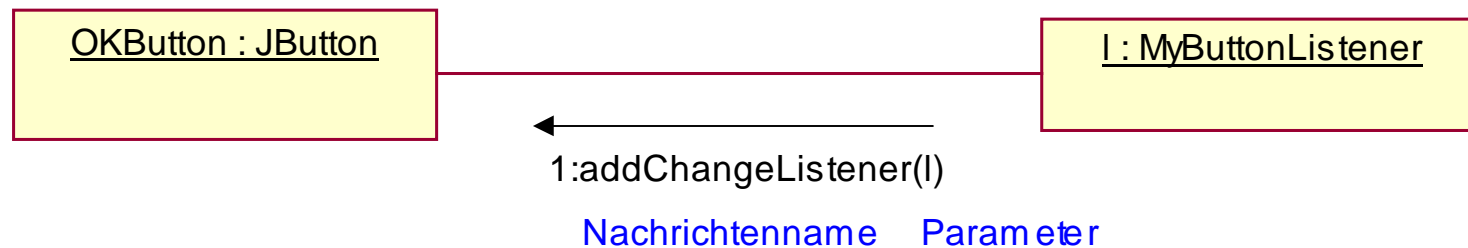
- *Nachricht* (Message) spezifiziert Kommunikation zwischen zwei Objekten
- Bestandteile/Aspekte einer Nachricht:
  - *Aufbau* (Signatur, Sequenz-Ausdruck)
  - *Synchronisationsverhalten*
  - *Stereotypen* für spezielle Nachrichten
  - *Einschränkungen* für Verbindungen und Objekte

# Nachricht - Signatur

---

- *Signatur* = Nachrichtenname und Parameterliste
- **Nachrichtenname** entspricht i.a. Operations- oder Signalname des Zielobjektes
- Operationsaufruf kann **Werte zurückliefern**
- **Syntax:**

Rückgabewerte := Nachrichtenname (Parameter)



# Nachricht - Sequenz-Ausdruck

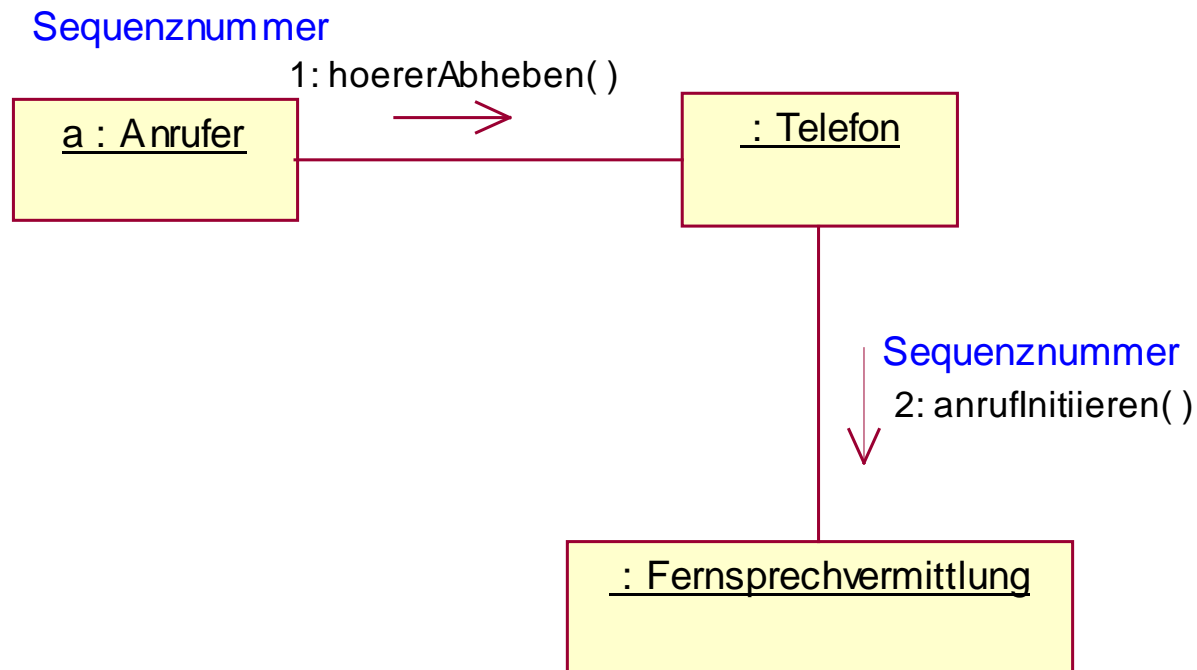
---

- Jede Interaktion besteht aus einer **Sequenz von Nachrichten**
- Man unterscheidet:
  - **flache** Nachrichtensequenzen
  - **verschachtelte** Nachrichtensequenzen

# Nachricht - Sequenz-Ausdruck (Forts.)

---

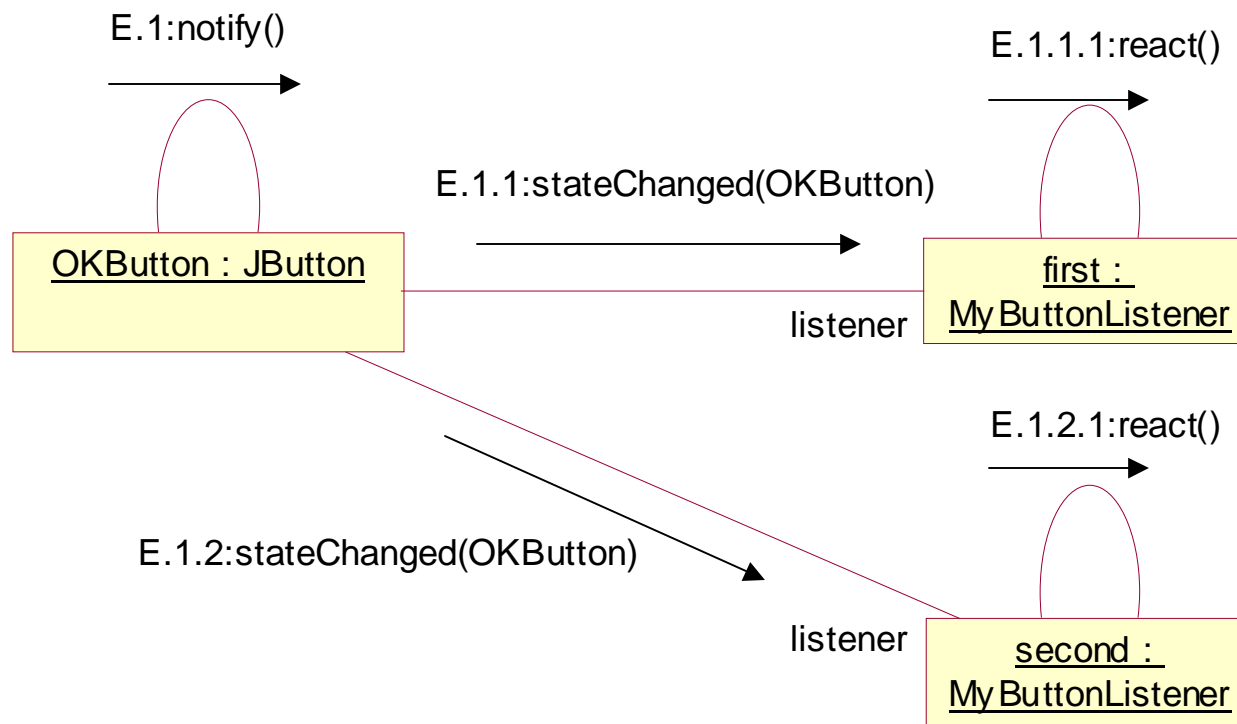
## Beispiel (flache Nachrichtensequenz):





# Nachricht - Sequenz-Ausdruck (Forts.)

## Beispiel (verschachtelte Nachrichtensequenz):



# Nachricht - Sequenz-Ausdruck (Forts.)

---

- **Syntax** für Sequenz-Ausdruck:

`<Sequenz-Term> [ . <Sequenz-Term> ] :`

wobei

`<Sequenz-Term> ::= [Name|int][Wiederholung]`

- **Beispiel:**

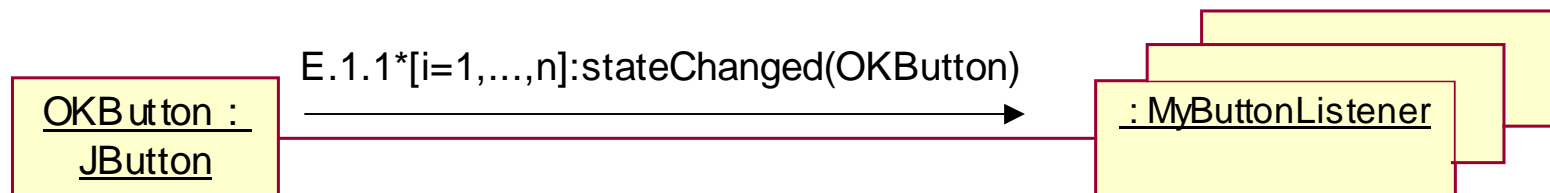
– `E.1:notify()`

– `E.1.1:stateChanged(OKButton)`

# Nachricht - Sequenz-Ausdruck (Forts.)

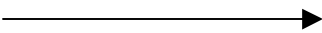
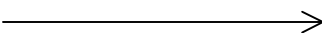
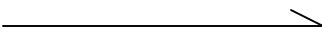
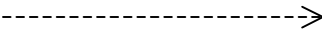
---

- Mit dem Wiederholungskonstrukt lässt sich **wiederholter** oder **bedingter** Aufruf einer Nachricht spezifizieren
- **Syntax:**  
**\*[Iterator-Ausdruck] bzw.**  
**[Bedingung]**



# Nachricht - Synchronisationsverhalten

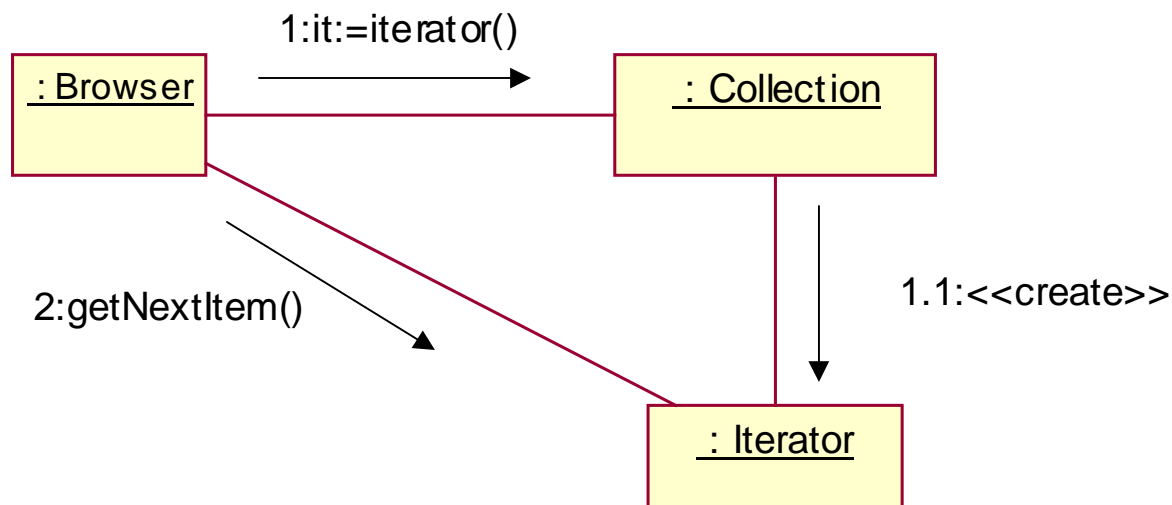
---

- Darstellung über **verschiedene Pfeilformen**
  - verschachtelter Aufruf (synchron) 
  - flacher Kontrollfluss (asynchron) 
  - asynchroner Kontrollfluss, i.a. für Versenden eines Signals 
  - Rückkehr eines Operationsaufrufs (meistens implizit) 

# Nachricht - Stereotypen für spezielle Nachrichten

---

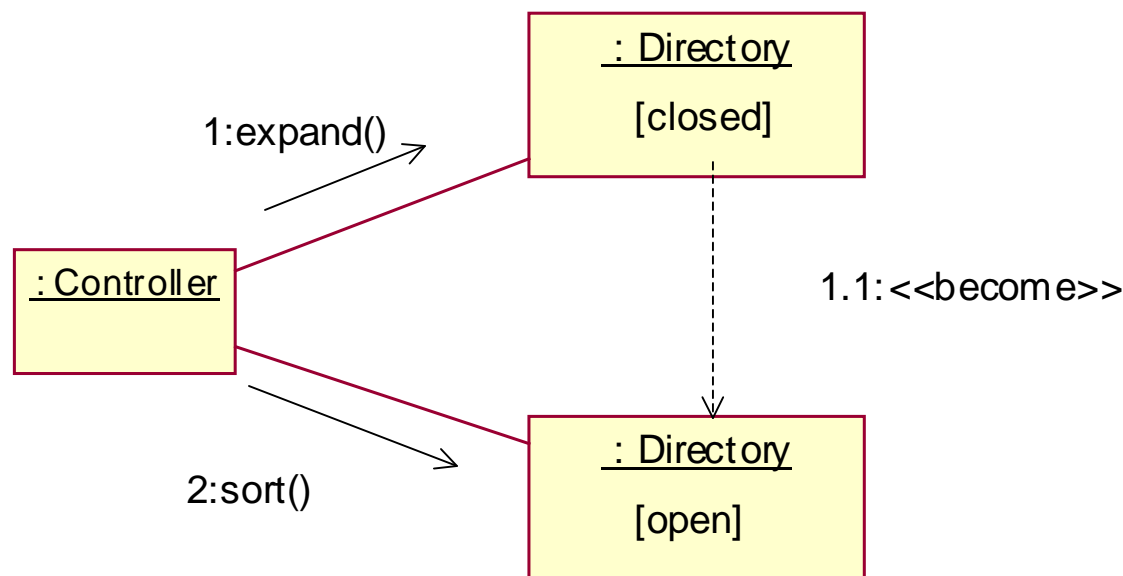
- **Kreieren** und **Zerstören** eines Objektes im Laufe einer Interaktion
- Stereotypen `<<create>>` und `<<destroy>>`



# Nachricht - Stereotypen für spezielle Nachrichten (Forts.)

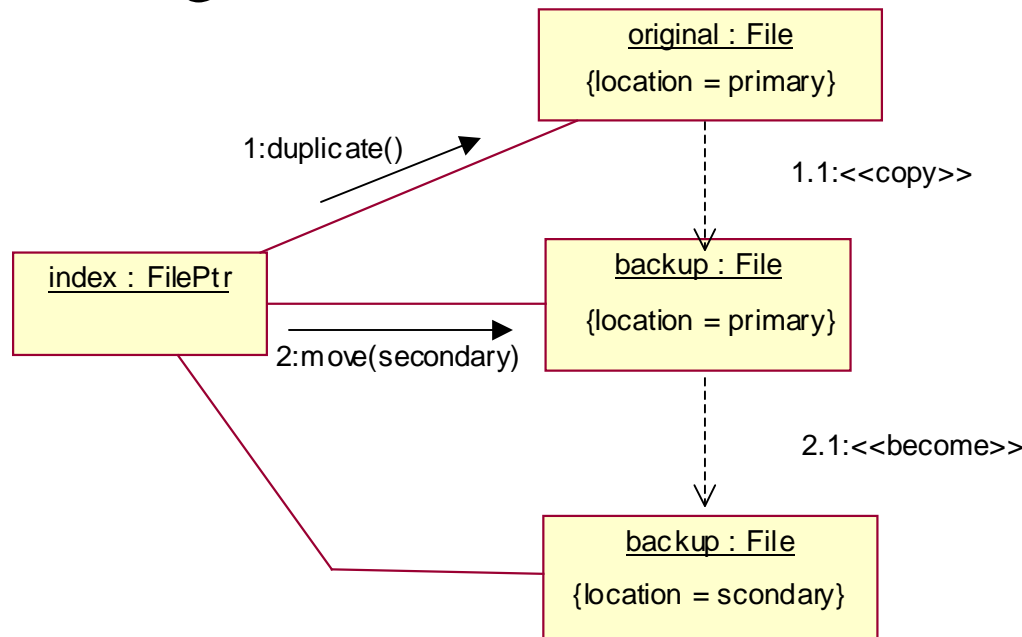
---

- Zustandsänderung eines Objektes
- Stereotyp <<become>>



# Nachricht - Stereotypen für spezielle Nachrichten (Forts.)

- **Kopieren** eines Objektes wird durch Stereotyp `<<copy>>` dargestellt



# Einschränkungen für Verbindungen und Objekte

---

- Genauere Beschreibung für Objekte und Verbindungen der Interaktion
  - {new}  
Objekt/Link wird während der Interaktion kreiert
  - {destroyed}  
Objekt/Link wird während der Interaktion zerstört
- Zusammen
  - {transient}  
Objekt/Link existiert nur während der Interaktion



# Darstellung von Interaktion

---

- Für die Interaktions-Sicht gibt es zwei Diagramm Arten
  - Kollaborationsdiagramm
  - Sequenzdiagramm
- Diese sind semantisch äquivalent

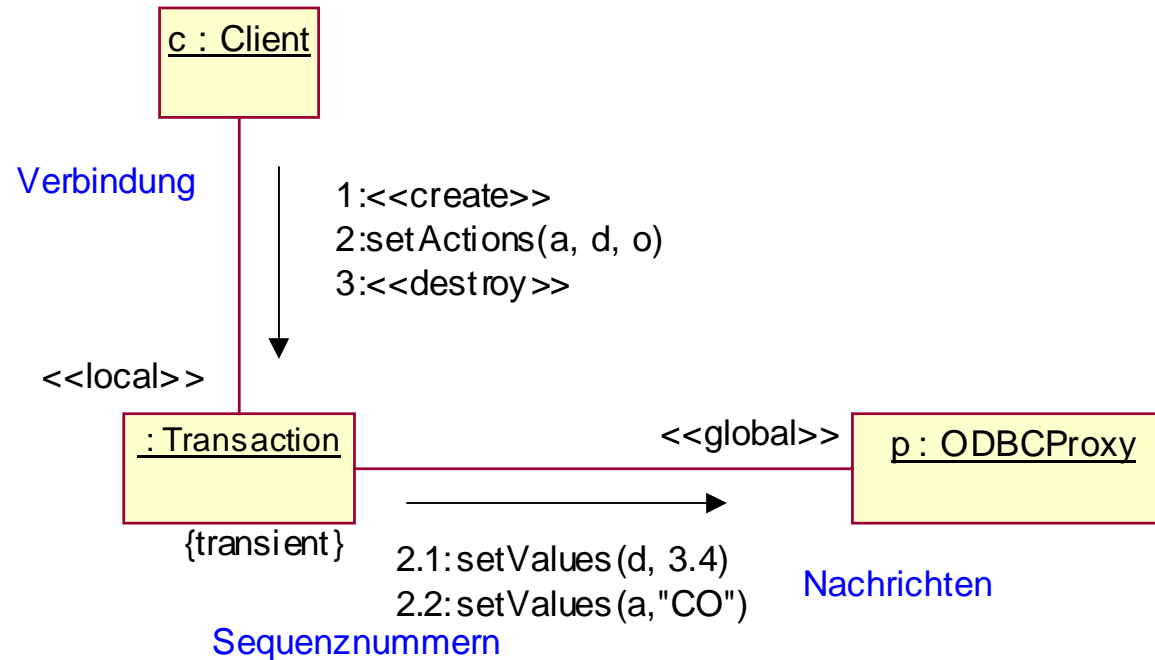
## 3.5.2 Kollaborationsdiagramm

---

- **Betonung** der **strukturellen Organisation** der Objekte, die an einer Interaktion beteiligt sind
- **Graphische Darstellung:**
  - Anordnung der beteiligten Objekte einer Interaktion gemäss ihrer Beziehungen
  - Eintragung der Verbindungen
  - Darstellung der Nachrichten mit Sequenznummern entlang der Verbindungen

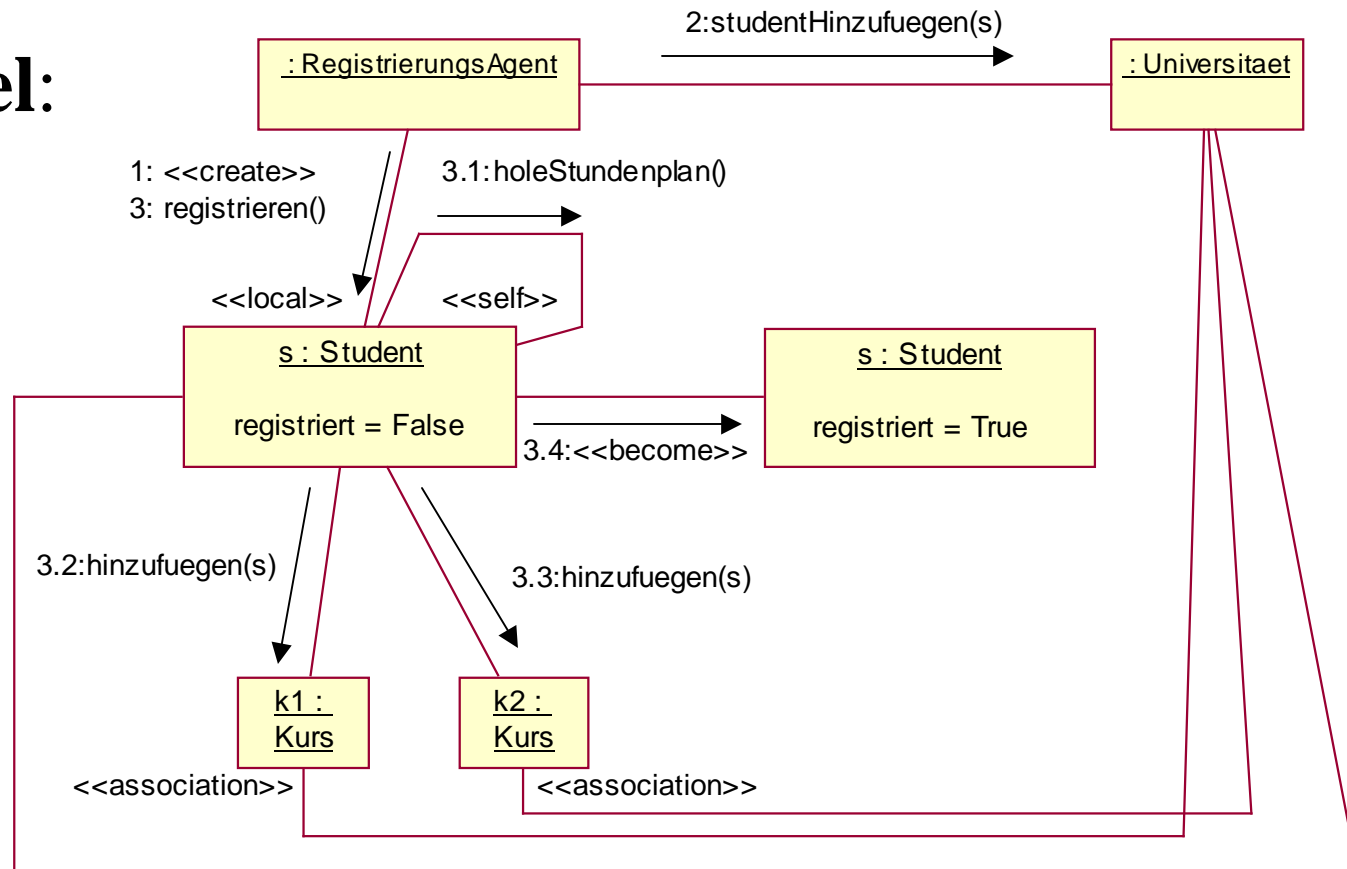
# Kollaborationsdiagramm (Forts.)

## Beispiel:



# Kollaborationsdiagramm (Forts.)

## Beispiel:



## 3.5.3 Sequenzdiagramm

---

- **Betonung** der **zeitlichen Abfolge** der Nachrichten
- **Graphische Darstellung:**
  - Anordnung der beteiligten Objekte einer Interaktion horizontal auf einer Linie  
(geordnet nach ihrem Auftreten)
  - Von jedem Objekt geht vertikal eine Linie - die *Lebenslinie* (Lifeline) - aus
  - Nachrichten als Pfeile zwischen den Lebenslinien der beteiligten Objekte

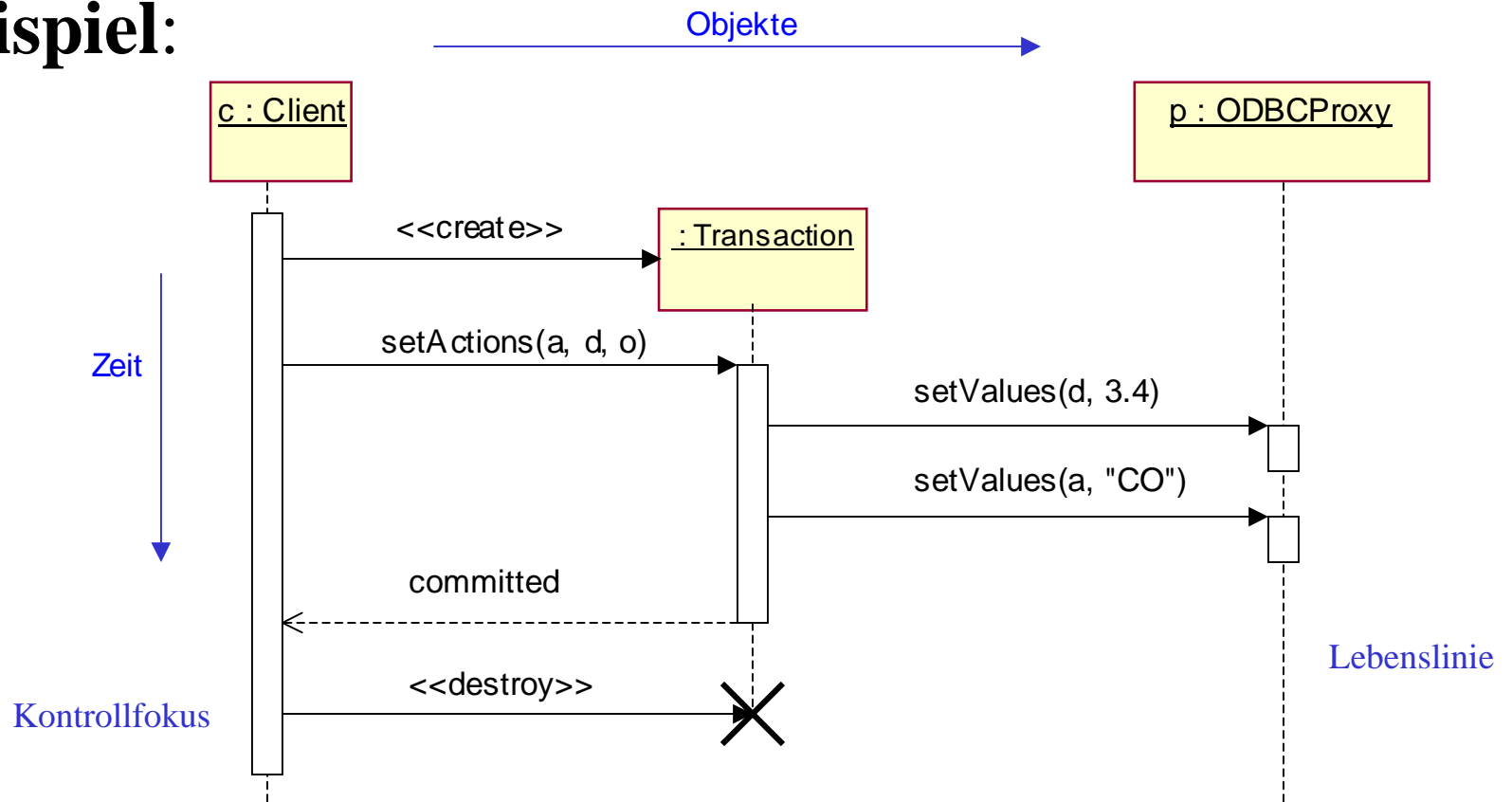
# Sequenzdiagramm (Forts.)

---

- Anordnung der Nachrichten von oben nach unten gemäss ihres zeitlichen Auftretens
- Sequenznummern werden i.a. nicht dargestellt, da zeitliche Abfolge durch Anordnung erkennbar

# Sequenzdiagramm (Forts.)

## Beispiel:



# Sequenzdiagramm - Lebenslinie

---

- **Darstellung/Eigenschaften (Lebenslinie):**
  - Gestrichelte Linie, welche die Existenz eines Objekts während eines Zeitraums darstellt
  - Objekte, die im Verlauf der Interaktion kreiert werden, werden erst auf Höhe der `<<create>>` Nachricht mit ihrer Lebenslinie eingetragen
  - Lebenslinien von Objekten, die im Verlauf der Interaktion zerstört werden, enden auf der Höhe der `<<destroy>>` Nachricht; Kennzeichnung durch **X**



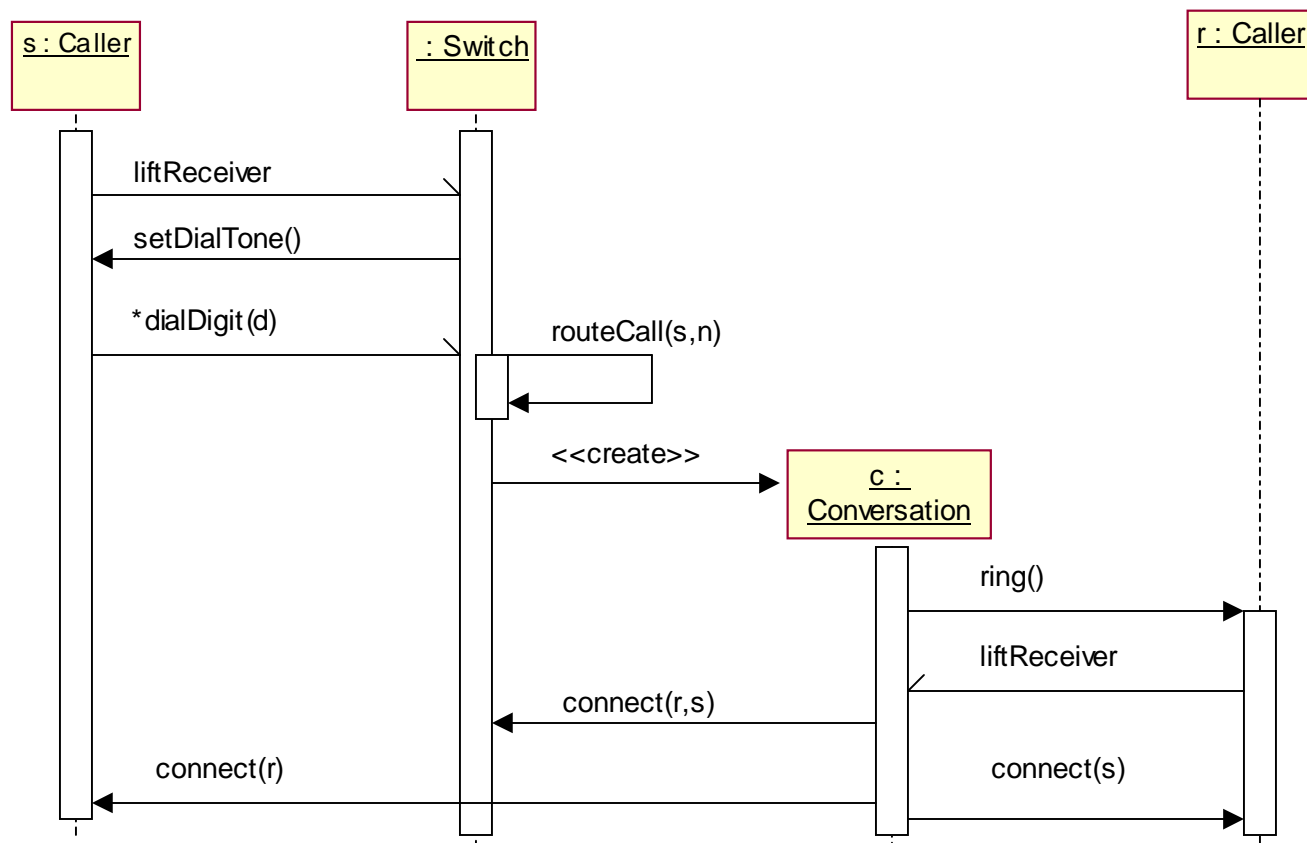
# Sequenzdiagramm - Kontrollfokus

---

- **Darstellung/Eigenschaften (Kontrollfokus):**
  - Darstellung der Zeiträume, in denen ein Objekt aktiv ist, durch schmales Rechteck, welches die Lebenslinie bedeckt (Kontrollfokus)
  - Selbstaufrufe (Rekursion) werden durch weiteren verschobenen Kontrollfokus dargestellt

# Sequenzdiagramm - Kontrollfokus (Forts.)

**Beispiel:**



## 3.5.4 Kollaborationen

---

- *Kollaboration* (Collaboration) = Sammlung von Klassen und Schnittstellen (konzeptuell), die zusammenarbeiten, um ein gewisses Verhalten zur Verfügung zu stellen
- **Graphische Darstellung:**



Inter-Node Messaging

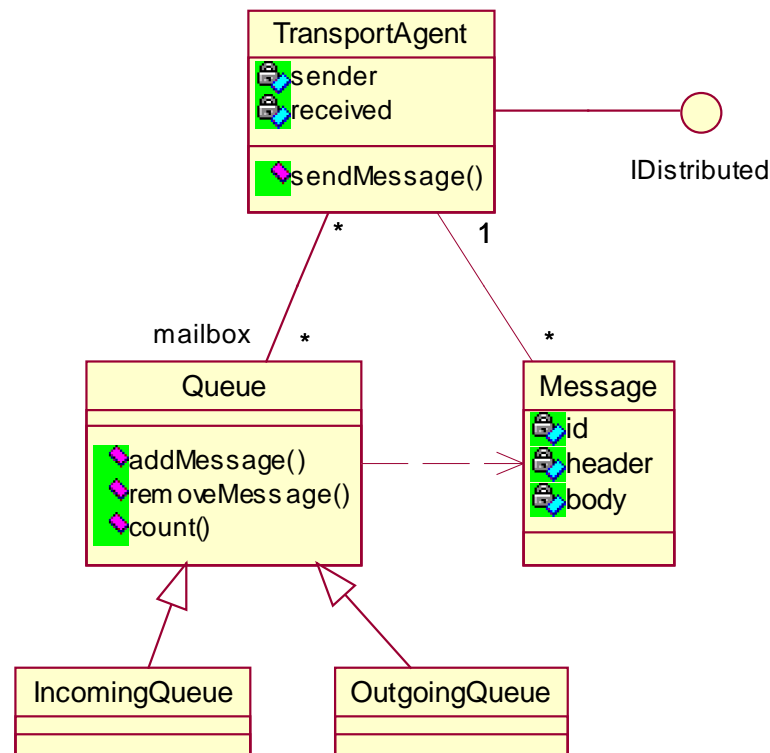
# Kollaborationen (Forts.)

---

- Kollaborationen beinhalten
  - Strukturellen Aspekt  
(dargestellt durch Klassendiagramme)
  - Verhalten  
(dargestellt durch Interaktionsdiagramme)

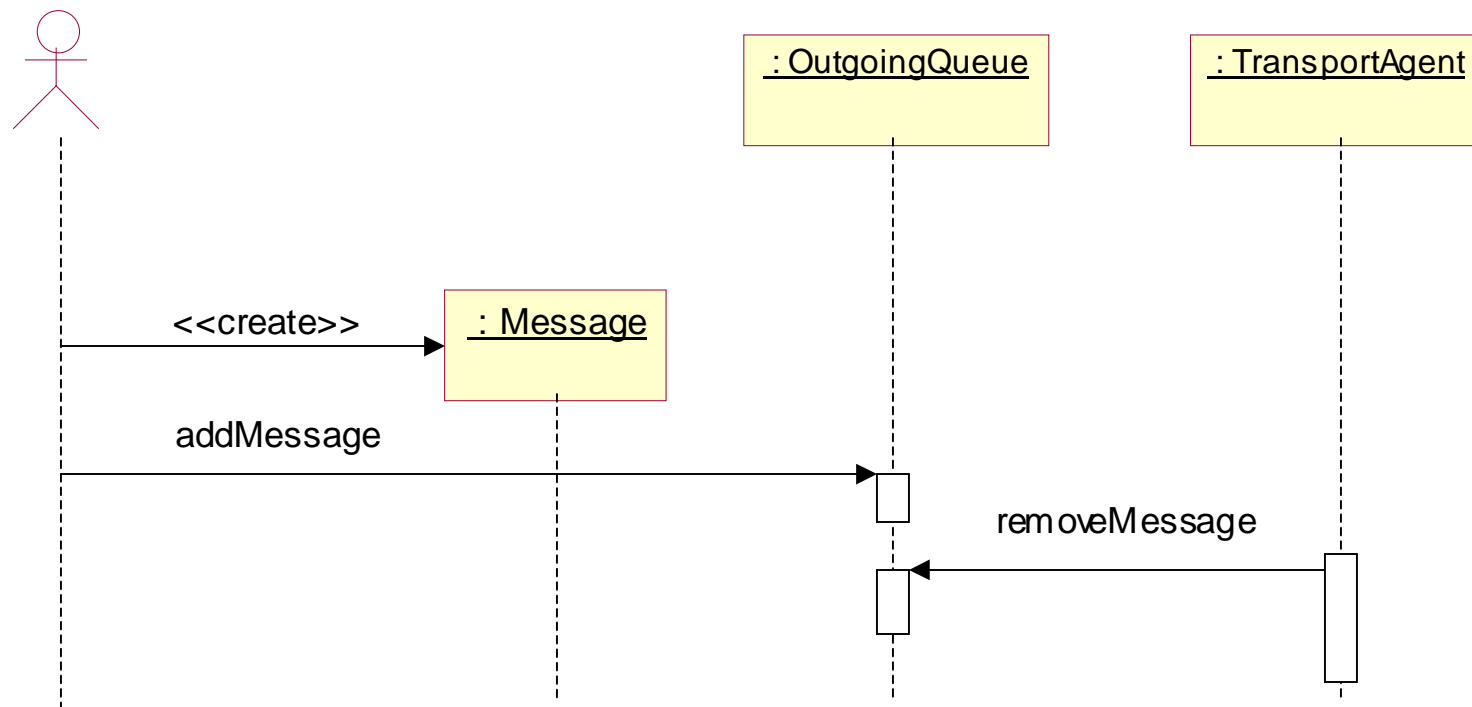
# Kollaborationen (Forts.)

## Beispiel (Struktur von Inter-Node Messaging):



# Kollaborationen (Forts.)

## Beispiel (**Verhalten** von Inter-Node Messaging):



# Kollaboration - Beziehungen

---

- „realisiert“-Beziehung zu Anwendungsfällen oder Operationen
- <<refine>> Abhängigkeit
  - Quelle hat niedrigeren Abstraktionsgrad als Ziel, also detaillierter spezifiziert

## 3.5.5 Entwurfsmuster und Rahmenwerke

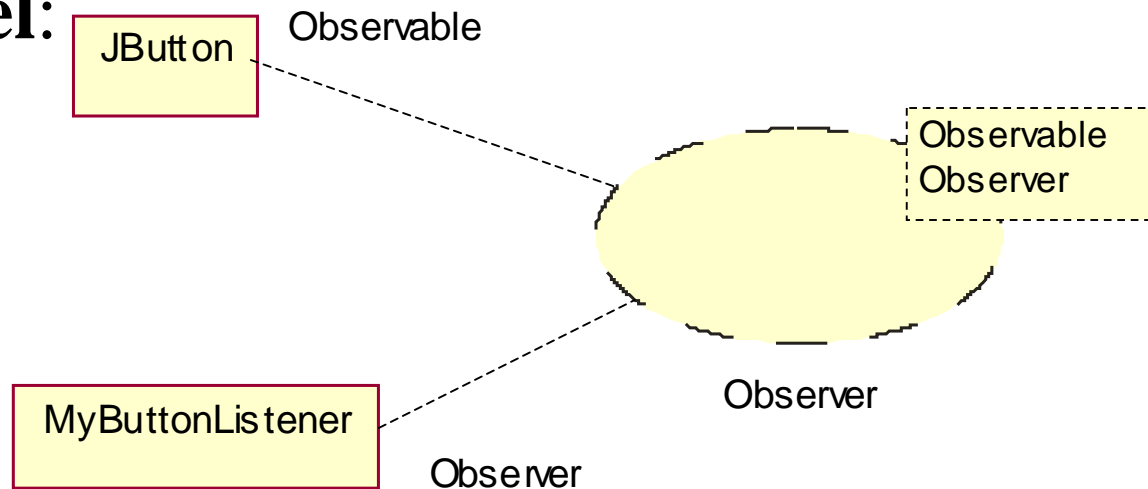
---

- *Entwurfsmuster* (Design Pattern) = allgemeine Lösung für ein allgemeines Problem, anwendbar auf beliebige Klassen  
(In UML Literatur findet man auch den Begriff **Mechanismus**)
- *Rahmenwerk* (Framework) = Architekturmuster, welches eine abstrakte, erweiterbare Struktur für Anwendungen eines Bereichs darstellt



# Entwurfsmuster

- Modellierung als **parametrisierte Kollaboration**, welche zur Benutzung an konkrete Klassen gebunden wird
- **Beispiel:**



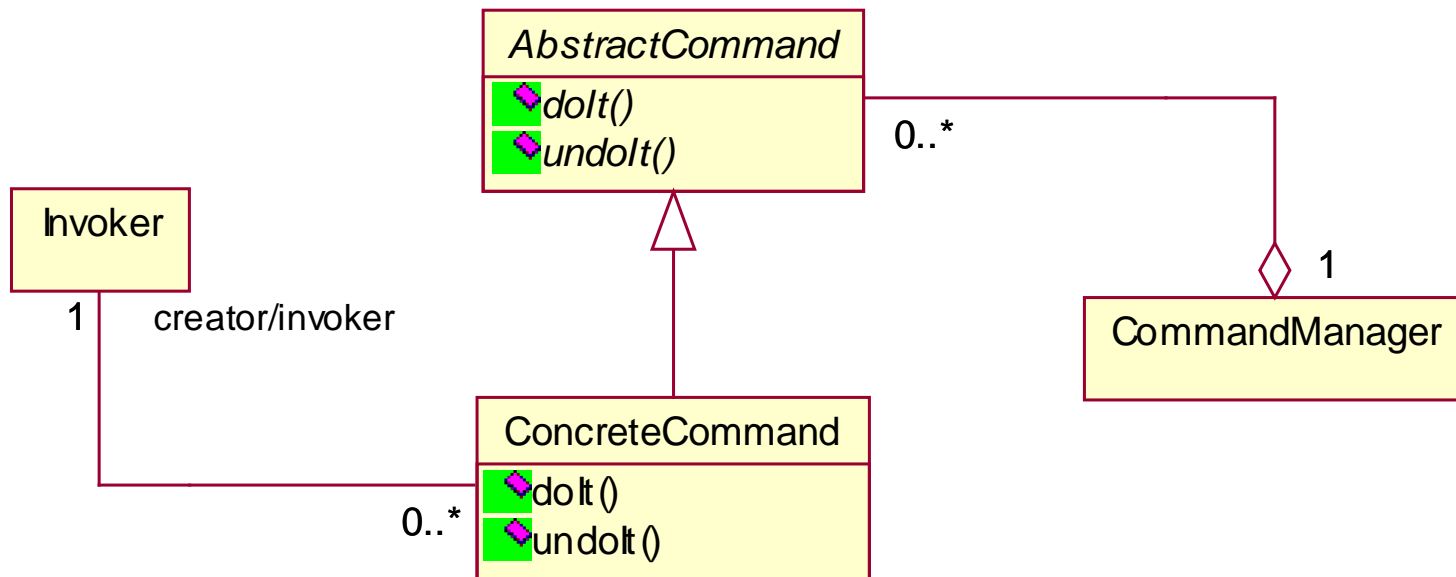
# Entwurfsmuster - Command

---

- *Command* Entwurfsmuster kapselt das Ausführen einer Operation in einem Objekt
- *Anwendung:*
  - Verwaltung und Protokollierung von Befehlssequenzen
  - Erstellen von Makros
  - Do/Undo Funktionalität

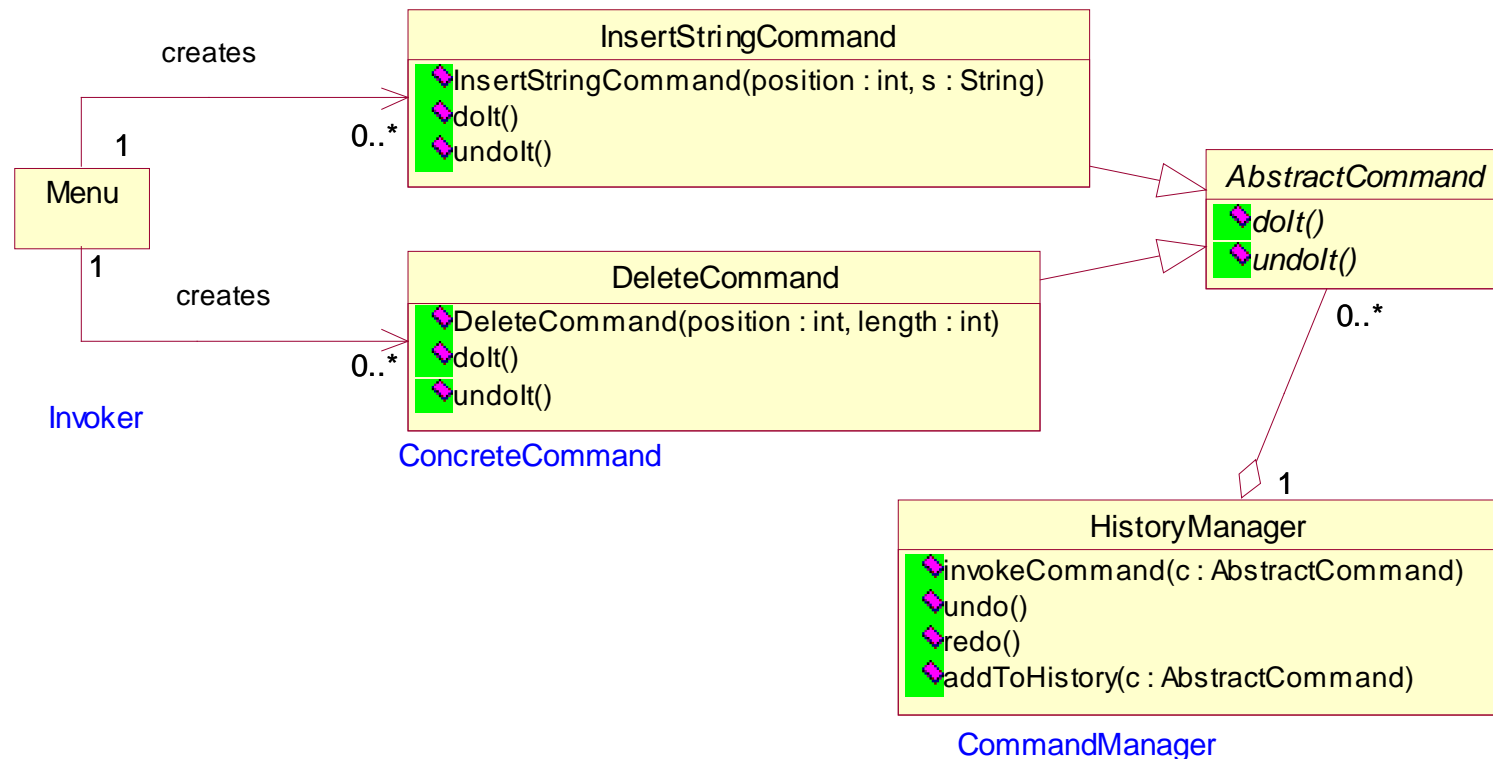
# Entwurfsmuster - Command (Forts.)

## Abstrakte Modellierung:



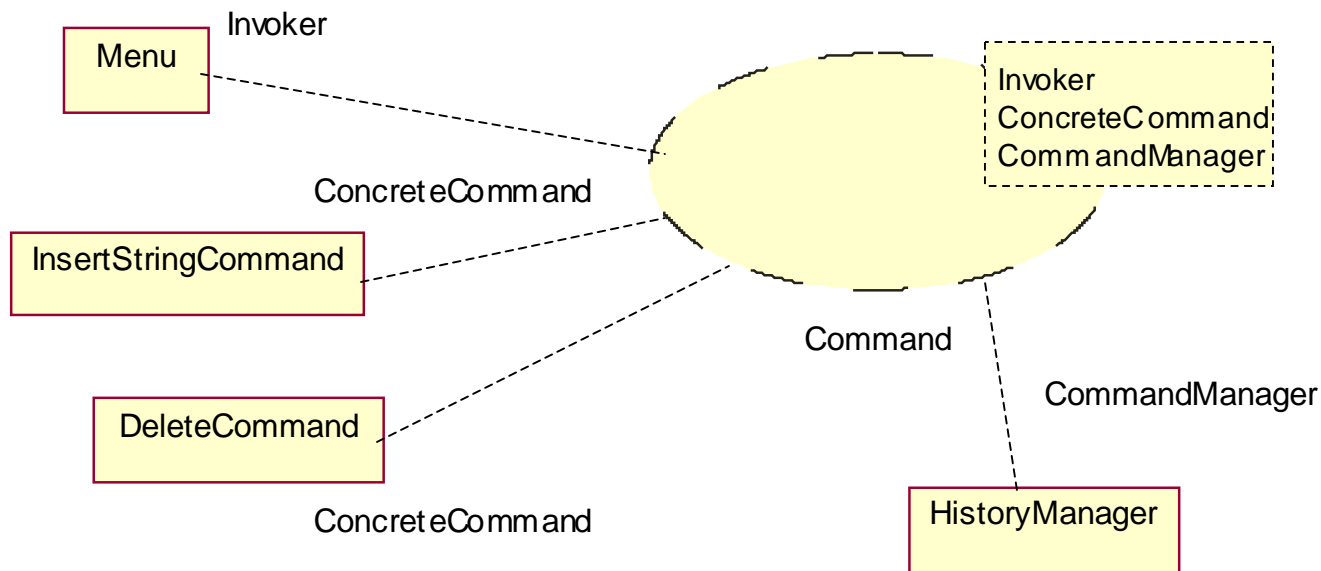
# Entwurfsmuster - Command (Forts.)

## Beispiel (Textverarbeitung):



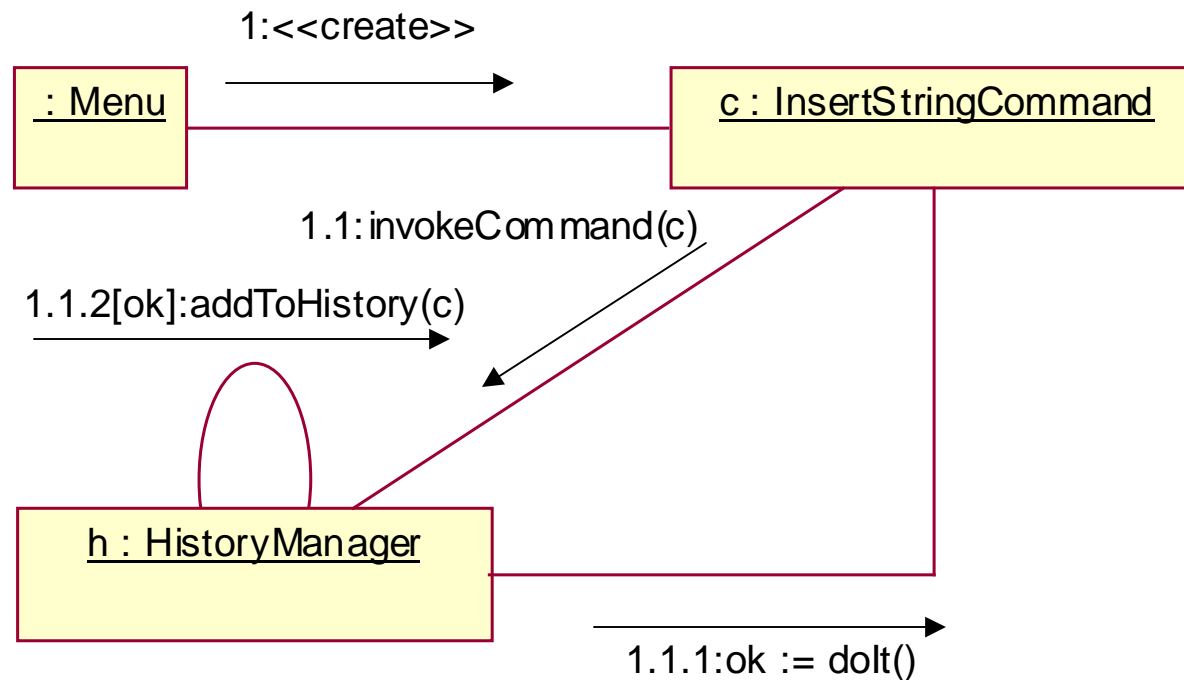
# Entwurfsmuster - Command (Forts.)

Modellierung als parametrisierte Kollaboration:



# Entwurfsmuster - Command (Forts.)

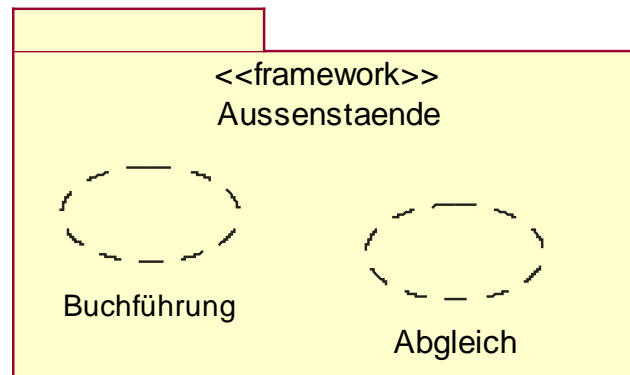
## Verhalten (Kreierung eines Commands):



# Rahmenwerk (Framework)

---

- **Vorgabe** einer **Architektur**, auf der durch **Bindung von Klassen** sowie **Erweiterungen** aufgebaut wird
- Modellierung als stereotypisiertes Paket, welches Entwurfsmuster und Kollaborationen zusammenfasst
- **Graphische Darstellung:**



# Entwurfsmuster und Rahmenwerke - Entwicklung

---

- Bottom-up Ansatz
- Abstraktion eines bewährten, oft benutzten Mechanismus zu einem Entwurfsmuster
- Extraktion wesentlicher, teilweise zusammenarbeitender Mechanismen zu einem Rahmenwerk



# Interaktions-Sicht - Zusammenfassung

---

- *Interaktionsdiagramme* zur Darstellung der Zusammenarbeit von Objekten
- Zusammenarbeit von Objekten
  - zur Realisierung von Anwendungsfällen
  - zur Realisierung von Operationen
- *Sequenzdiagramm* betont zeitliche Abfolge der Nachrichten zwischen beteiligten Objekten

# Interaktions-Sicht - Zusammenfassung (Forts.)

---

- *Kollaborationsdiagramm* betont strukturellen Zusammenhang der beteiligten Objekte
- Zusammenfassung von **struktureller** Information (Klassendiagramme) und Verhaltensinformation (Interaktionsdiagramme) in *Kollaborationen*
- Falls Zusammenarbeit in einer Kollaboration abstrahierbar  $\Rightarrow$  *Entwurfsmuster*

# Interaktions-Sicht - Zusammenfassung (Forts.)

---

- Zusammenfassung von bewährten Mechanismen und Kollaborationen einer Architektur zu *Rahmenwerken*