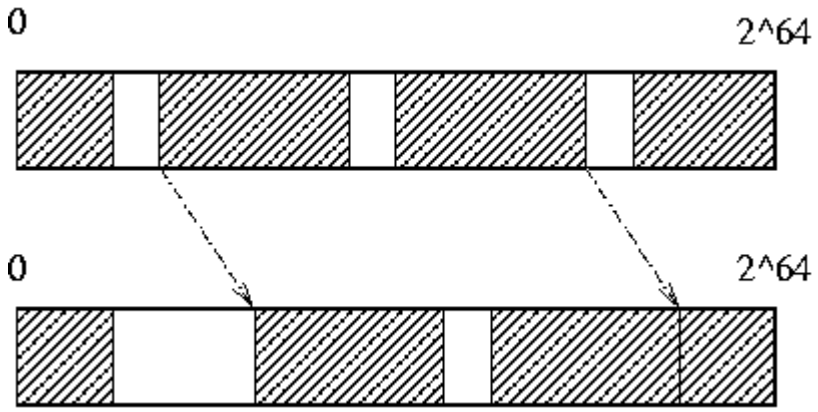
Nester und Bausteine

Eine Betriebssystem-Architektur
mit nur zwei Abstraktionen:
Nester und Bausteine

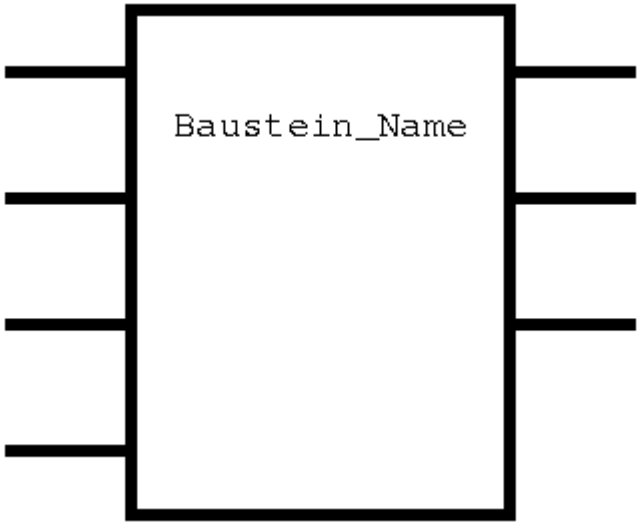
Thomas Schöbel-Theuer
schoebel@informatik.uni-stuttgart.de

Nester

- Adressraum-Abstraktion (Software, auch Hardware)
- Sparse (Löcher)
- Definitionsbereich im ***adjungierten Nest***
- ***Verschiebe-Operation move***



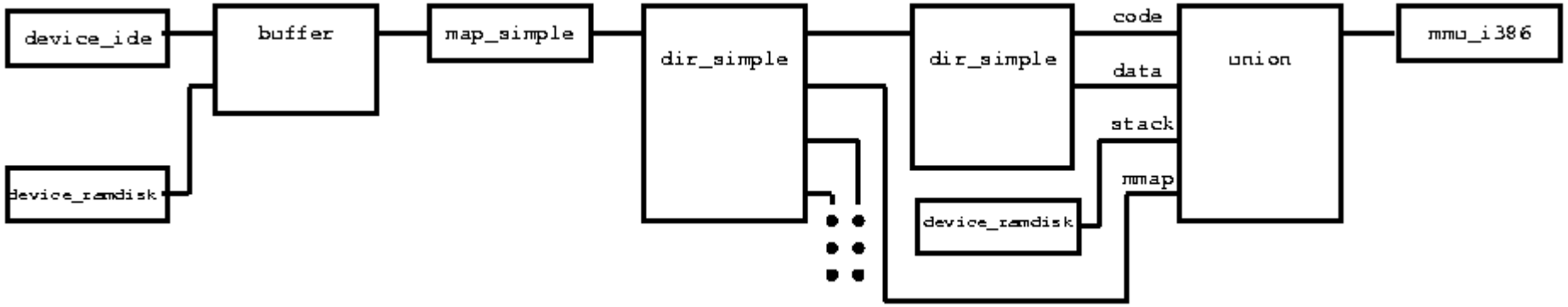
Bausteine



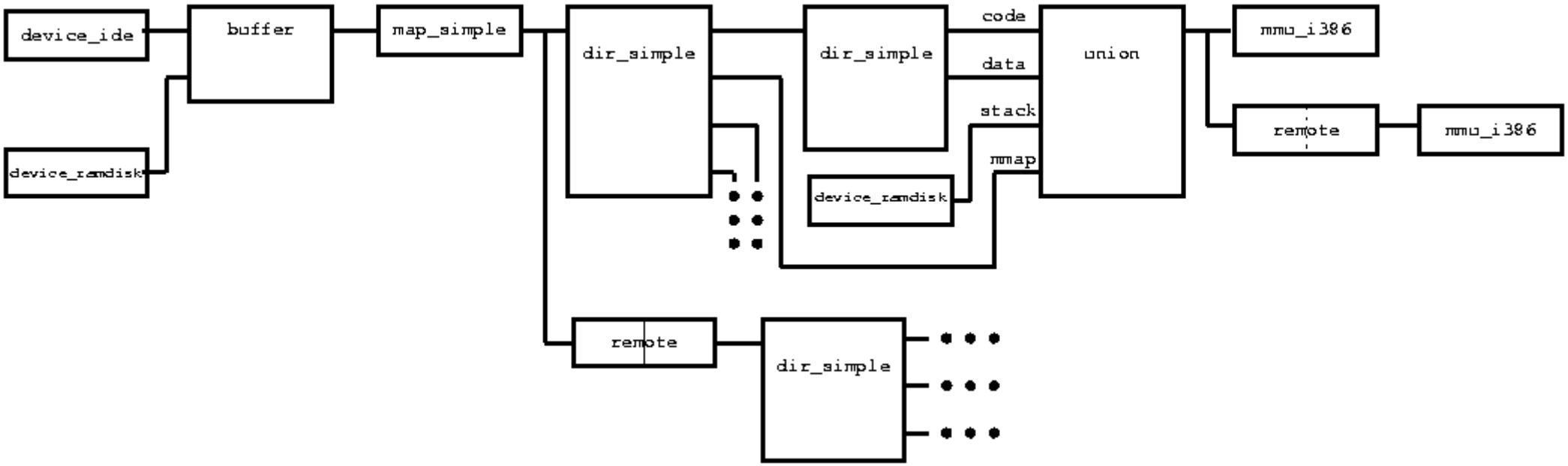
Transformator: Eingangs-Nester --> Ausgangs-Nester

Verdrahtungs-Leitung = Nest-Instanz

Beispiel 1


device_ide
buffer
map_simple
dir_simple
dir_simple
code
data
stack
mmap
union
mmu_i386
device_ramdisk
device_ramdisk

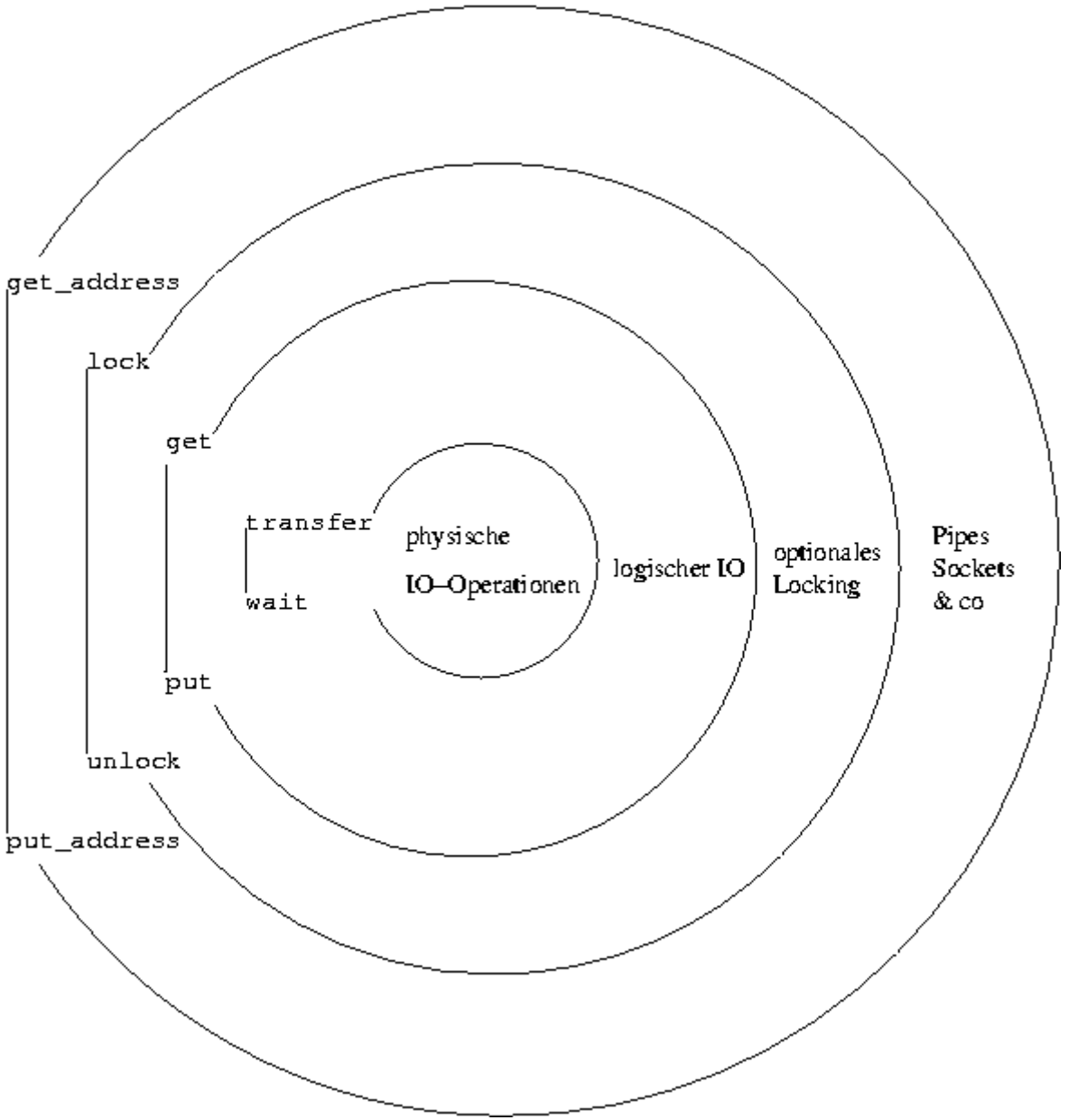
Beispiel 2


device_ide
buffer
map_simple
dir_simple
dir_simple
code
data
stack
mmap
union
mmu_i386
remote
mmu_i386
device_ramdisk
device_ramdisk
remote
dir_simple

Entwurfsprinzipien

- Universelle Generizität
- Kompositorische Generizität
- Erweiterungs-Generizität

Nest-Elementaroperationen

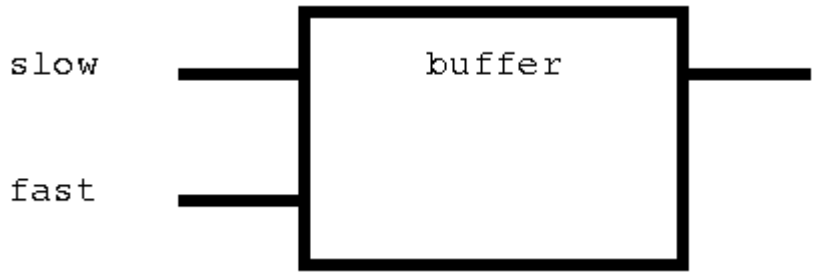


Aufruf-Mechanismen

- Unabhängige / späte Bindung an Aufruf-Mechanismus
  - RPC
  - LRPC
  - Indirekter Prozeduraufruf
  - Direkter Prozeduraufruf
  - Makro / Inline-Expansion

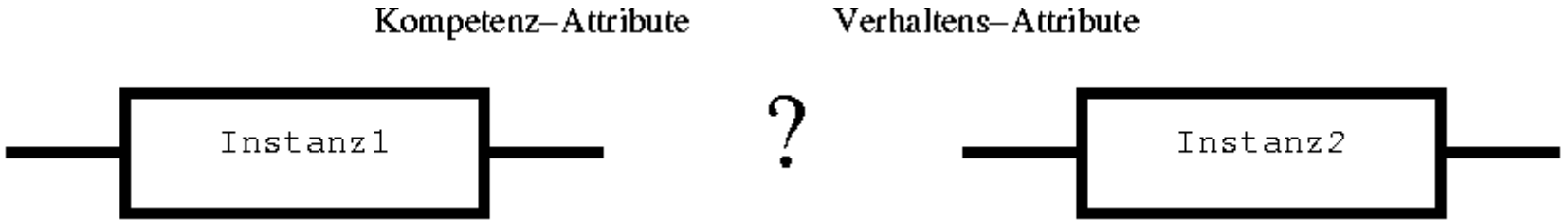
Statuslosigkeit

Re-Instantiierung ohne beobachtbare Änderung



Modelle

- Einfache versus komplizierte Ausbaustufen
- Darstellung: Attribute
- Funktional oder nichtfunktional
- Statische vs. dynamische Attribute
- Beispiel: **singleuser multiuser multiversion**



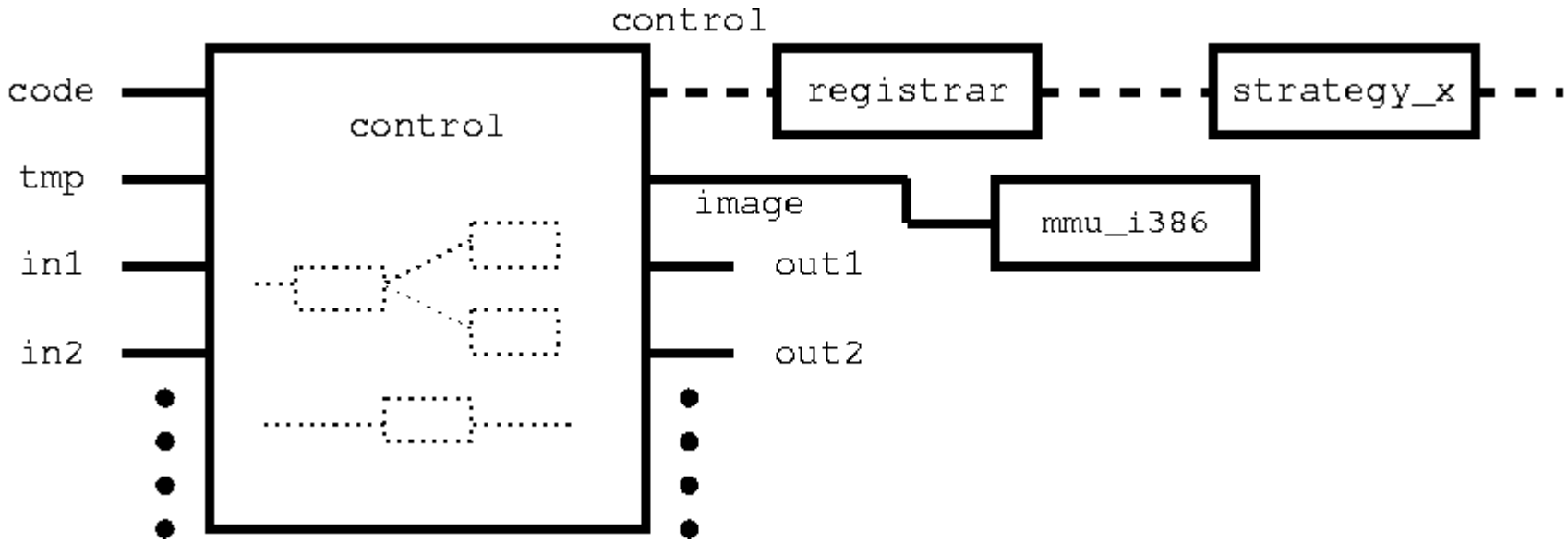
Fazit

- Theoretische Mächtigkeit: simuliert OO, FP und vermutlich auch AOP
- Extrem flexibel an dynamische Anforderungen anpassbar
- Transaktionen und Datenbank-Funktionalität integrierbar
- Aushandeln nichtfunktionaler Anforderungen wird erleichtert

Ausblick

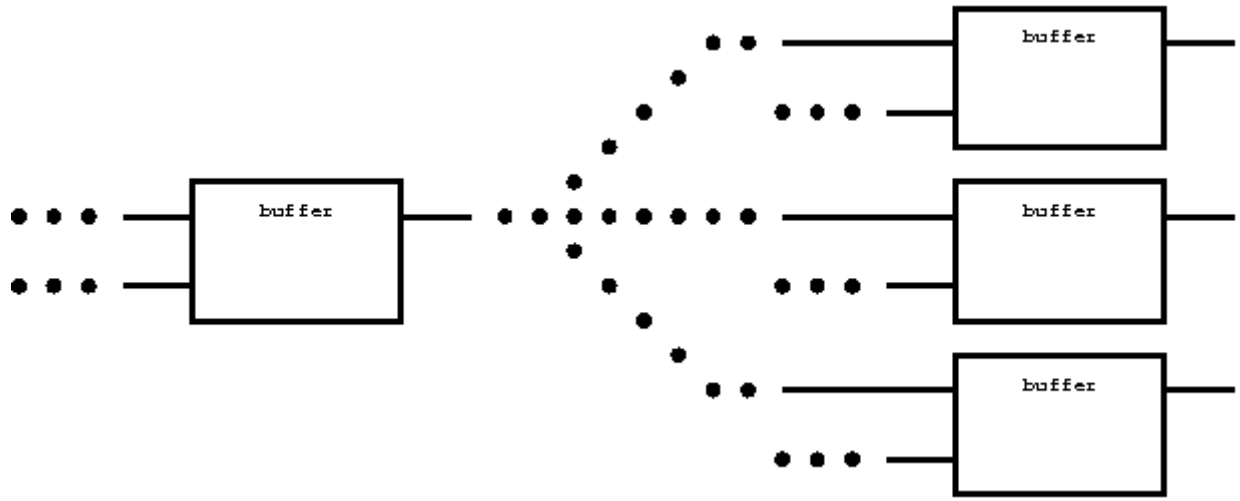
- Implementierung --> Evaluation möglich
- Detailprobleme bedürfen weiterer Forschung, z.B. Sicherheit bei dynamischer Rekonfiguration
- Kompatibilität zu existierenden BS
  - Generische Operationen
  - Nest != File (wegen Definitionsbereich)

Instantiierung



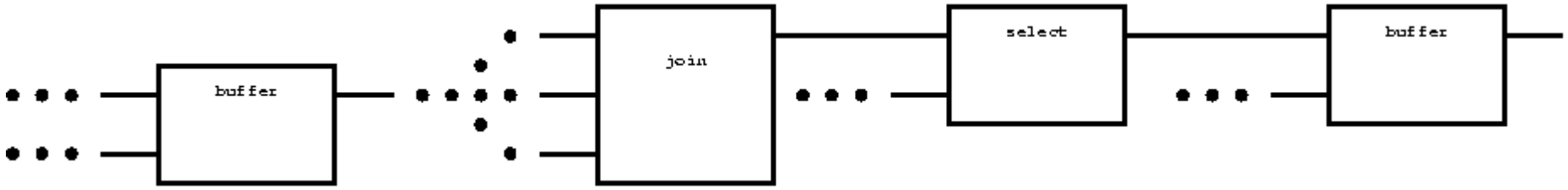
Cache-Kohärenz

- Rückgabe von Datenblöcken: notify_data
- Rückgabe von Locks: notify_lock
- --> spekulatives Locking

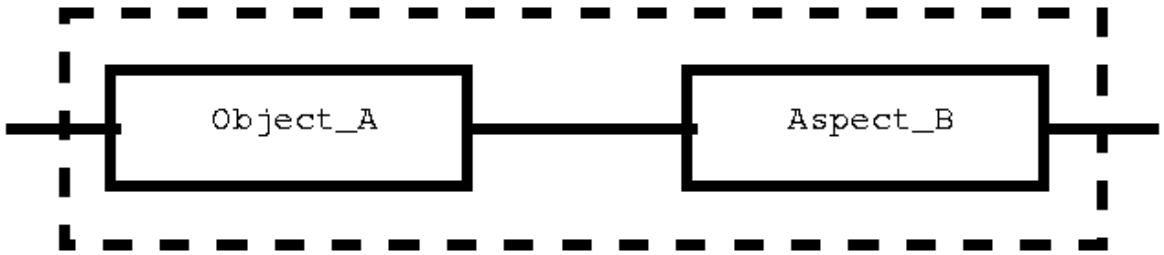


Ungewöhnliche Anwendung

Mehrfache Caches



Simulation von AOP (Idee)



Interpretativ (evtl. Dyn. Codegenerierung)