

POWER ⚡ FLASHER

Design Patterns Workshop

von Nico Zimmermann

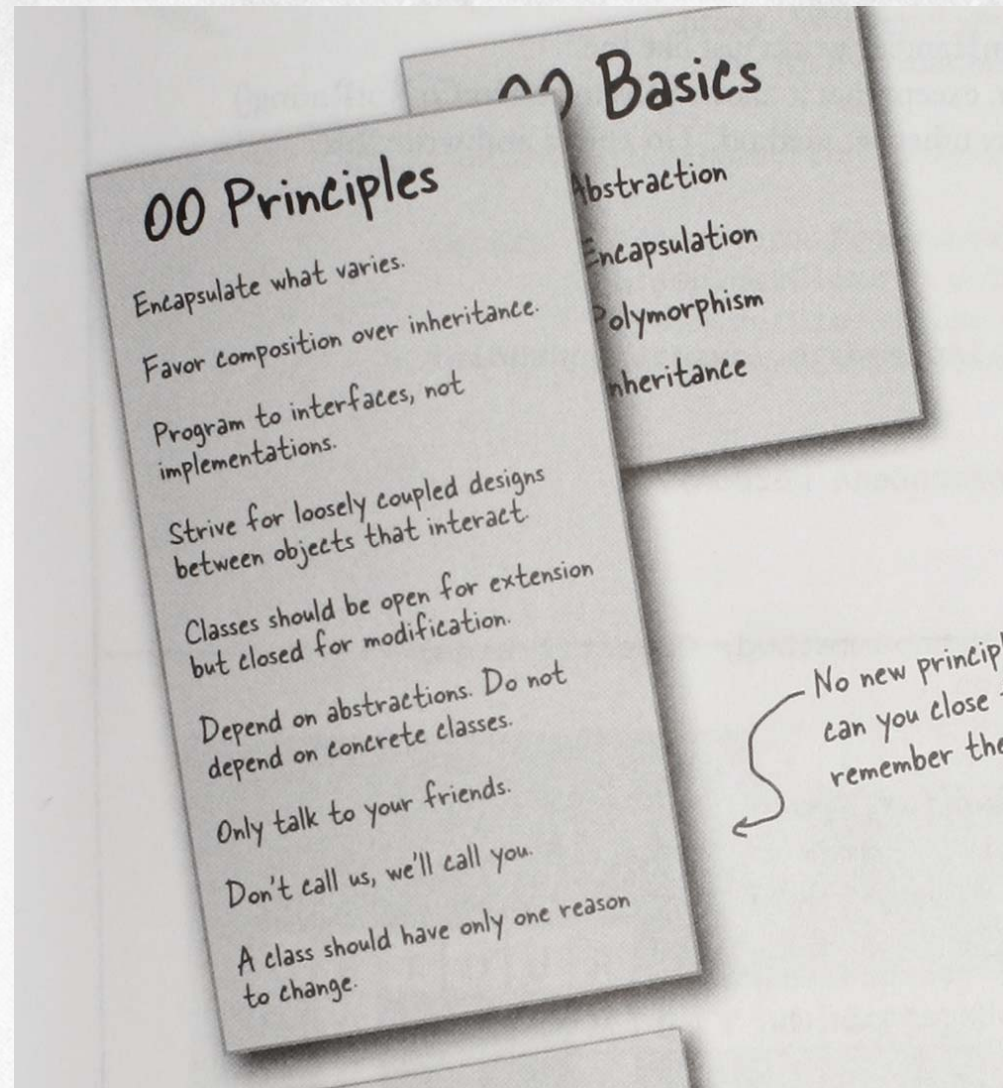
Tools der Software Entwicklung

- Prozessmodelle (Waterfall/ XP/ Scrum)
- Informatische Grundlagen (Algorithmik)
- Unit Testing
- Refactoring
- OOP Basics/ Prinzipien
- Design Patterns

OOP Basics

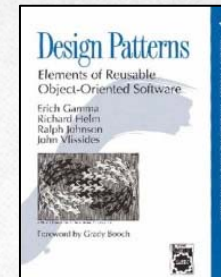
- Abstraction
- Encapsulation
- Polymorphism
- Inheritance

OOP Prinzipien

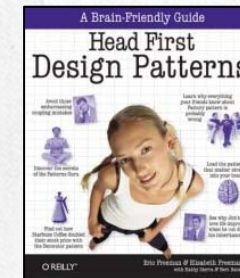


Bekannte Design Patterns Bücher

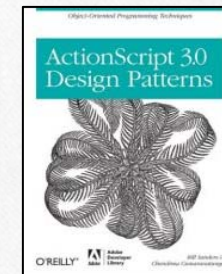
Design Patterns, Elements of Reusable Object-Oriented Software
Erich Gamma (GoF)



Head First Design Patterns
Elisabeth und Eric Freeman



ActionScript 3.0 Design Patterns
William Sanders



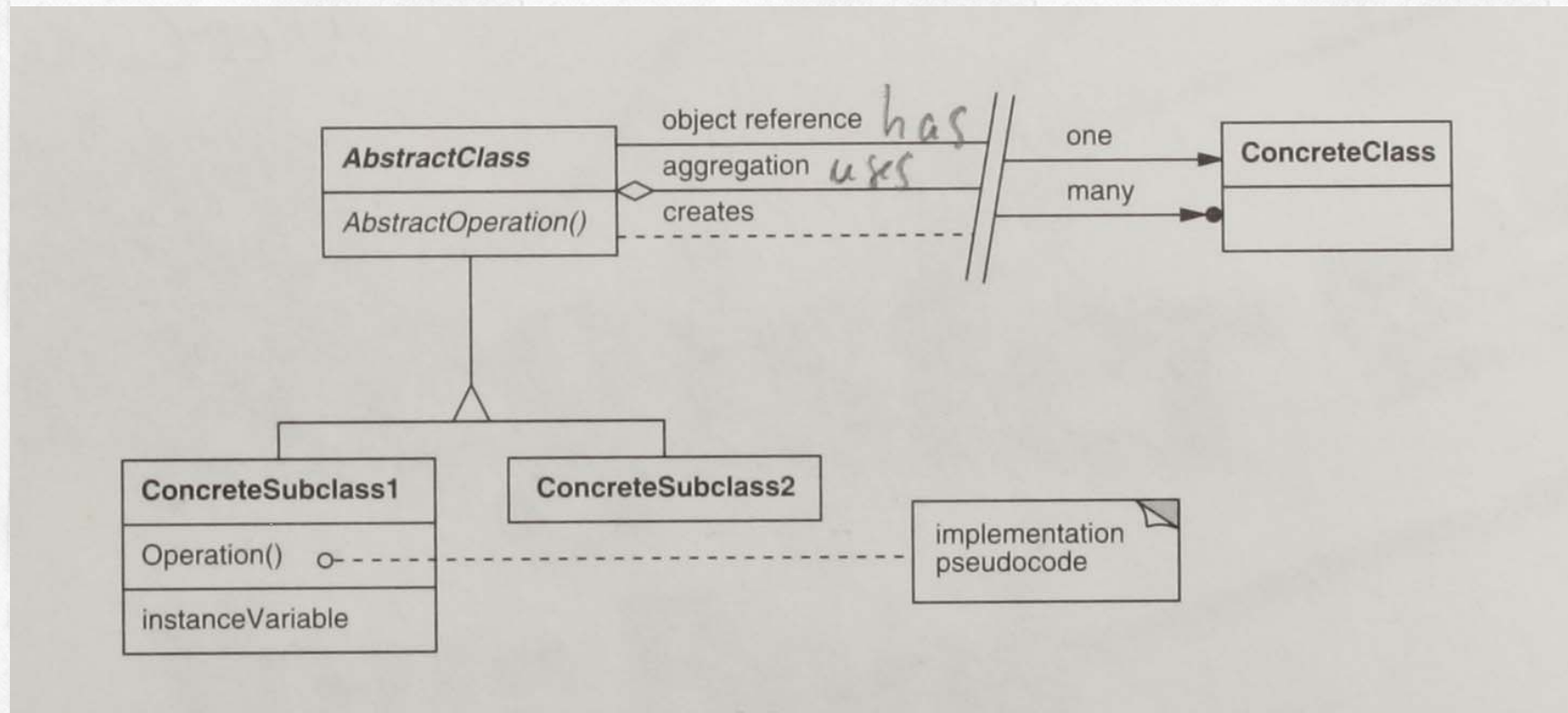
Advanced ActionScript 3 with Design Patterns
Joey Lott, Danny Patterson



Woraus besteht ein Design Pattern?

- **Name**
- **Wann nutze ich das Pattern? (Motivation)**
- **Was ist das Design Ergebnis? (Intend)**
- **Wie baue ich das Pattern? (Structure/ Teilnehmer)**
- **Welche Vor- und Nachteile hat das Pattern? (Consequences)**

UML Basics



Strategy Pattern

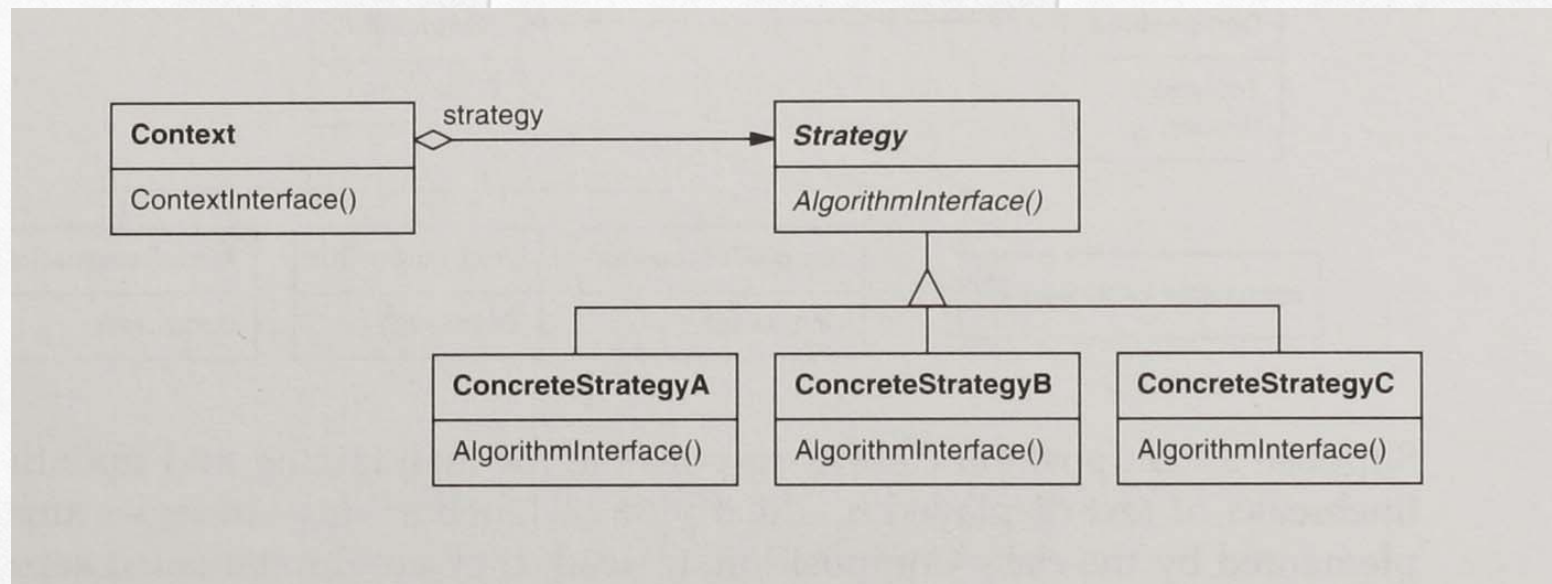
Motivation (Wann nutze ich das Pattern?):

Viele ähnliche Klasse unterscheiden sich nur durch ihr Verhalten.

Intent (Was ist das Design Ergebnis?):

Definiere eine Familie von Algorithmen, kapsle jede einzelne und mache sie austauschbar.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



State Pattern

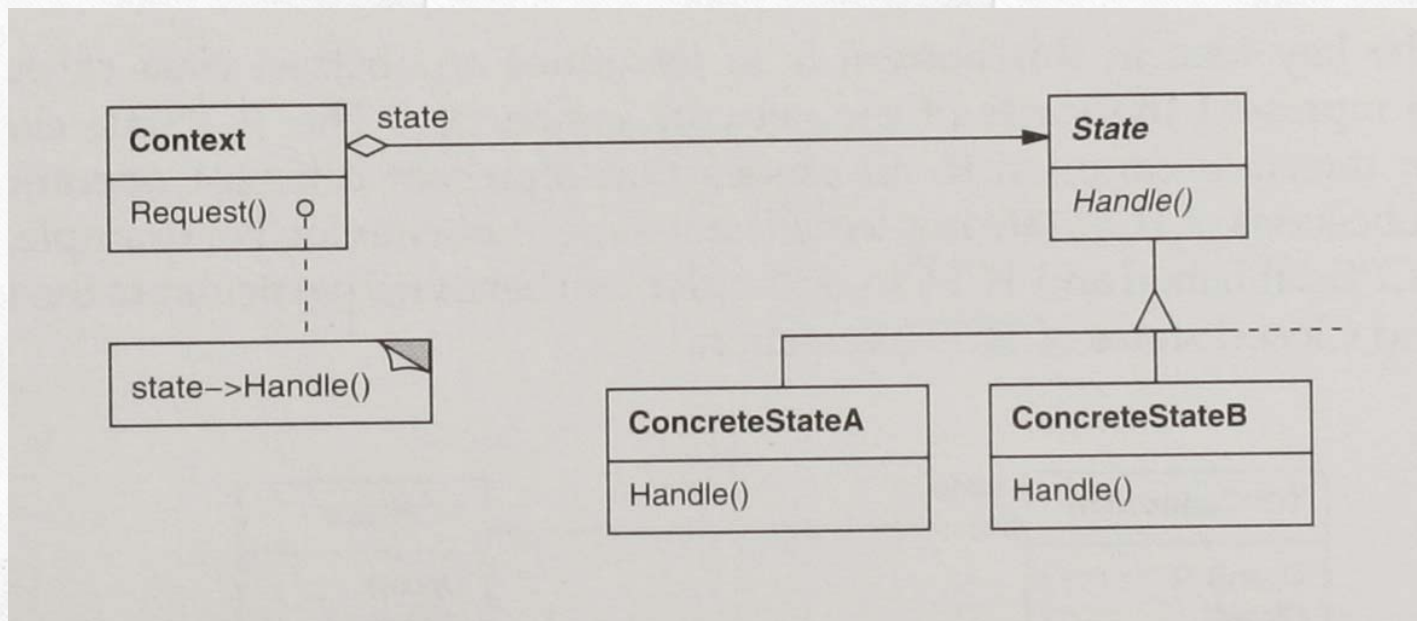
Motivation (Wann nutze ich das Pattern?):

Ein Objekt verhält sich je nach seinem Zustand unterschiedlich. Durch Statusvariablen und viele Bedingungen wird der Code unübersichtlich.

Intent (Was ist das Design Ergebnis?):

Ermögliche einem Objekt seinen Verhalten zu ändern, wenn sein interner Status wechselt.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Memento Pattern

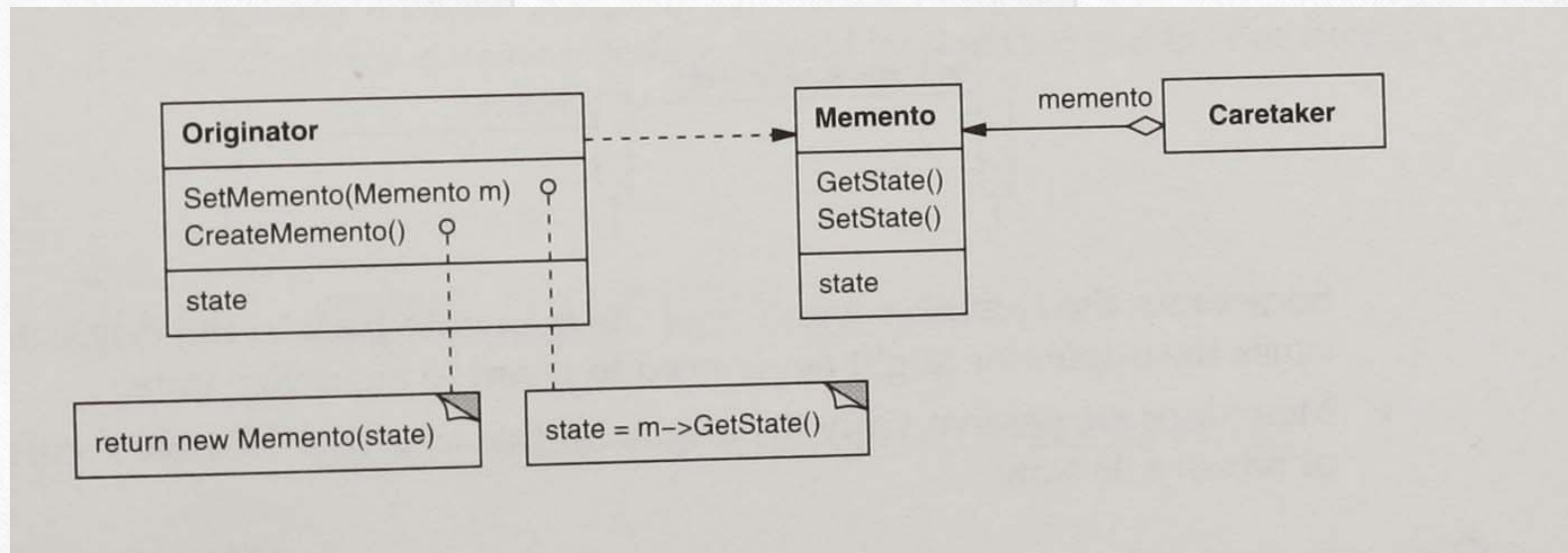
Motivation (Wann nutze ich das Pattern?):

Der Status eines Objektes soll gespeichert werden und das Objekt kann seinen Status daraus wieder herstellen.

Intent (Was ist das Design Ergebnis?):

Speichere und stelle den Status eines Objektes wieder her, ohne die Privatsphäre des Objektes zu verletzen.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Builder Pattern

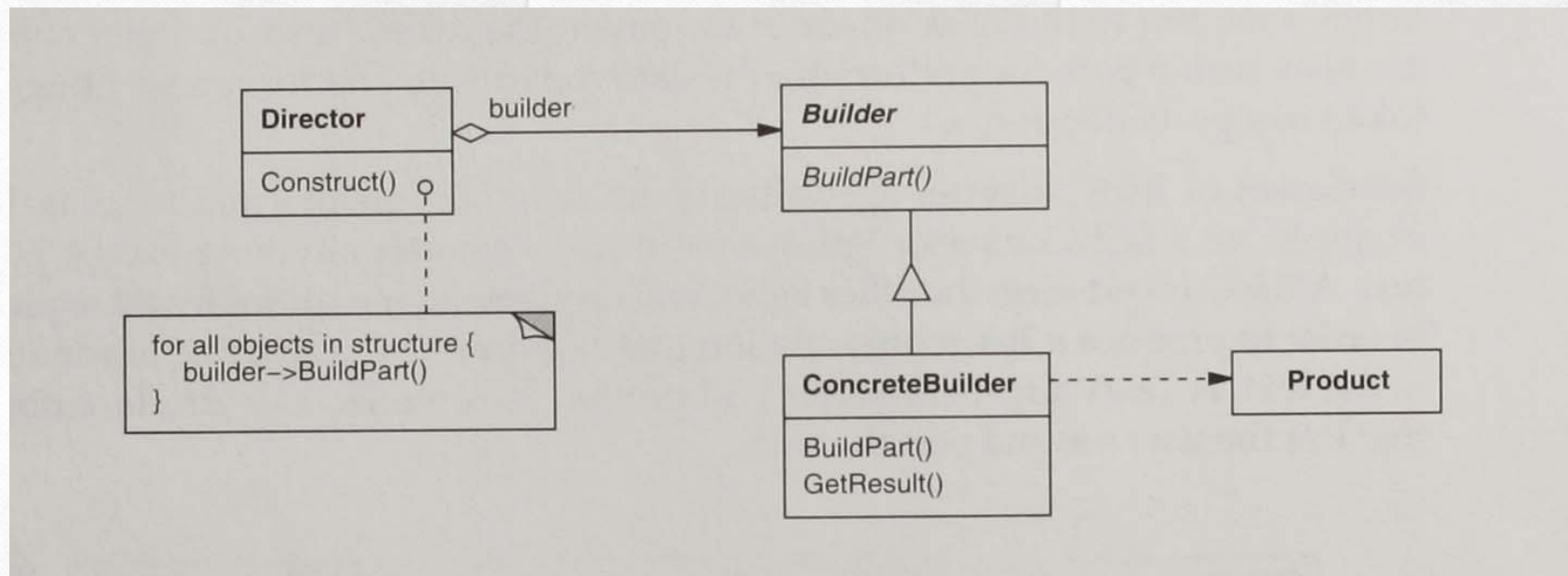
Motivation (Wann nutze ich das Pattern?):

Die Konfiguration oder die Konstruktion eines Objektes soll vereinfacht oder generalisiert werden.

Intent (Was ist das Design Ergebnis?):

Trenne die die Konstruktion eines komplexen Objektes von deren Aussehen, so dass die selbe Konstruktion unterschiedliche Aussehen haben kann.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Flyweight Pattern

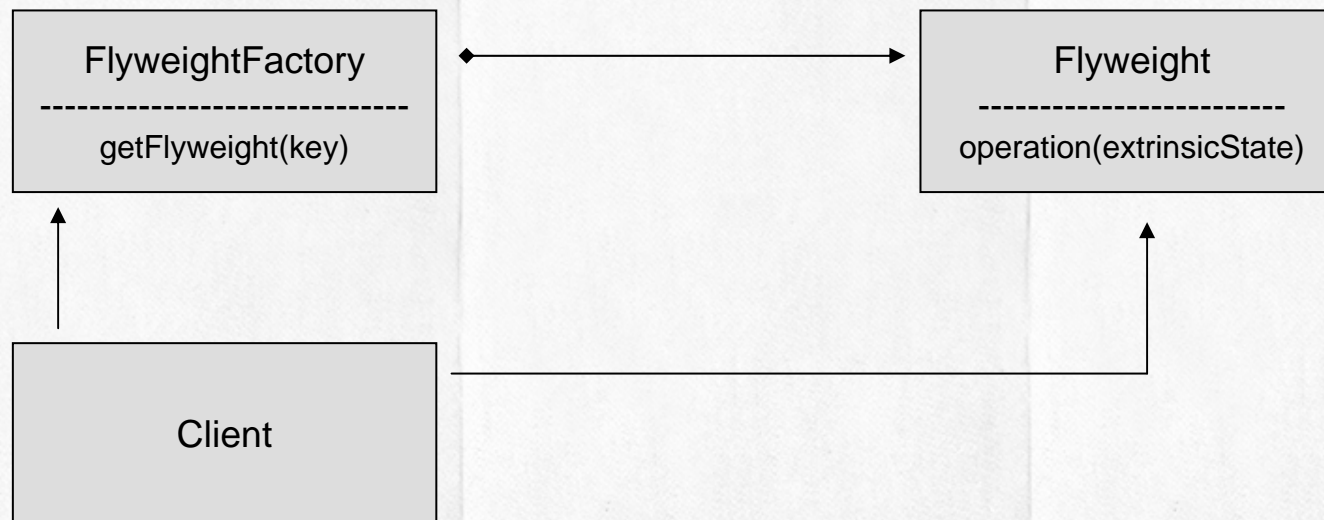
Motivation (Wann nutze ich das Pattern?):

Es gibt eine große Anzahl gleicher Objekte, die auf ein Objekt reduziert werden könnten.

Intent (Was ist das Design Ergebnis?):

Ermögliche durch das Teilen von Objekten eine effizientere Nutzung.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Abstact Factory Pattern

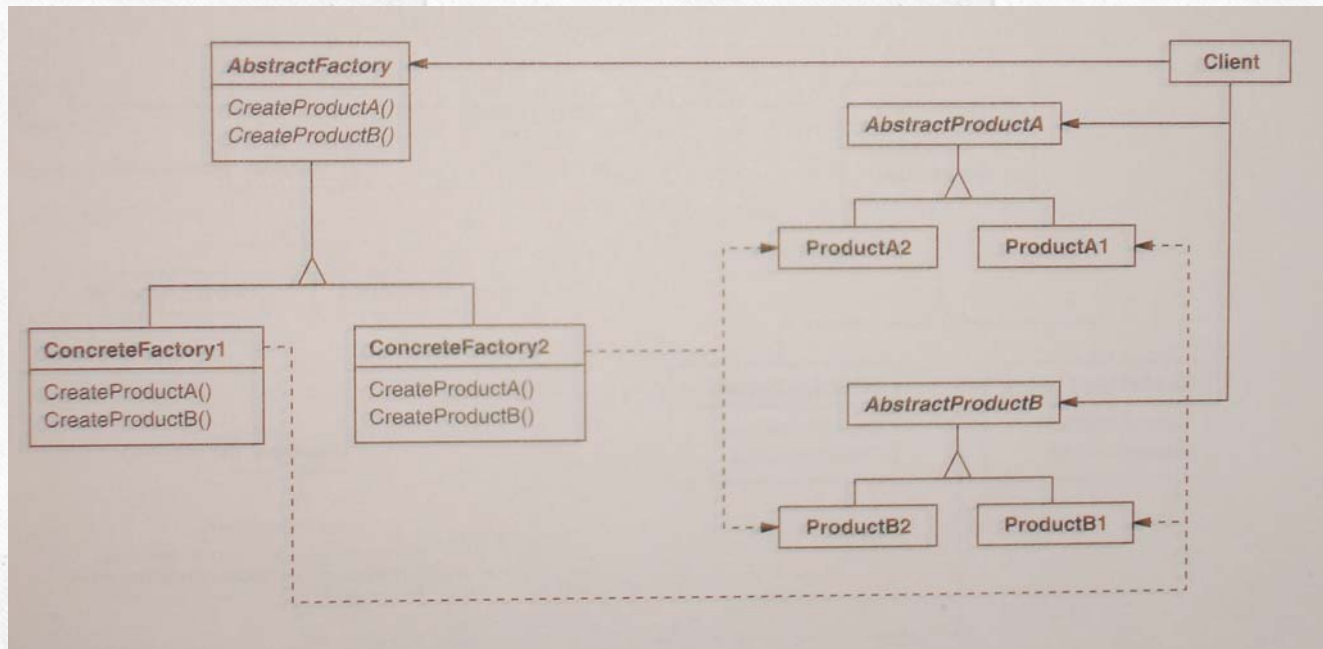
Motivation (Wann nutze ich das Pattern?):

Eine Klasse soll Instanzierungen vornehmen, spezifiziert das zu erzeugende Objekt aber nur abstrakt.

Intent (Was ist das Design Ergebnis?):

Stelle ein Interface zur Erzeugung von Familien von Klassen zur Verfügung ohne die konkreten Klassen zu kennen.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Singleton Pattern

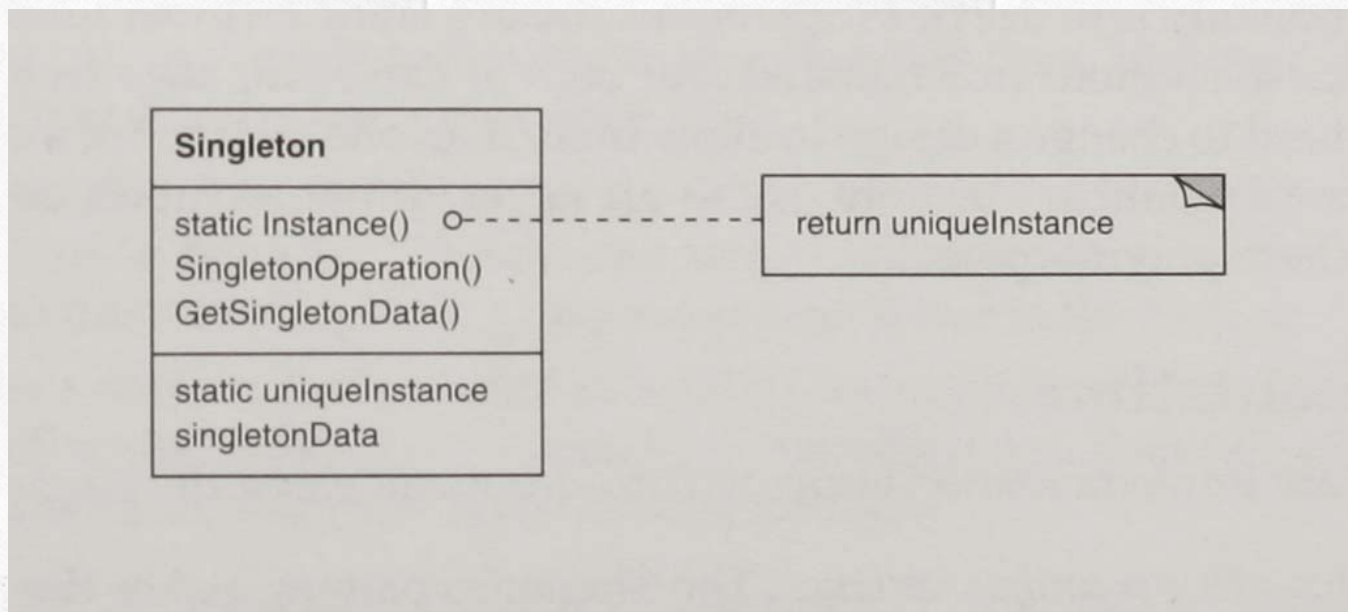
Motivation (Wann nutze ich das Pattern?):

Für manche Klassen ist es wichtig, dass sie nur eine Instanz haben.

Intent (Was ist das Design Ergebnis?):

Stelle sicher, dass eine Klasse nur eine Instanz hat und ermögliche einen globalen Zugriff auf sie.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Mediator Pattern

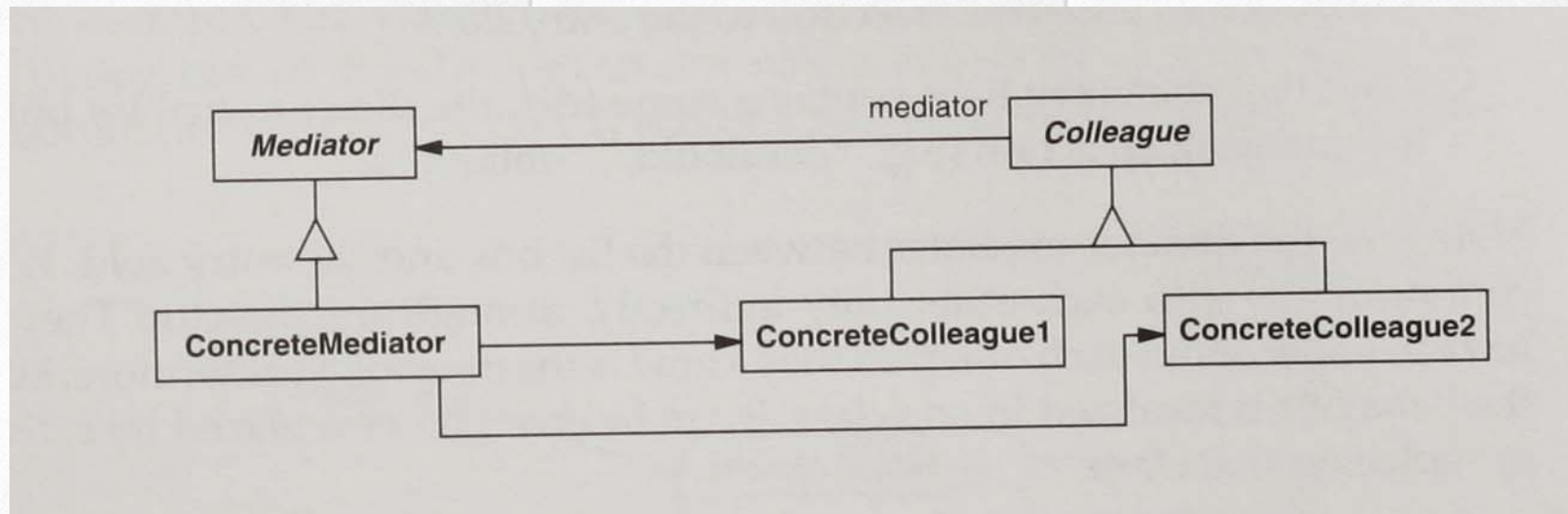
Motivation (Wann nutze ich das Pattern?):

Eine Gruppe von Objekten, die miteinander kommunizieren sollen kennen sich nicht.

Intent (Was ist das Design Ergebnis?):

Definiere ein Objekt welches die Interaktion zwischen anderen Objekten kapselt.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Iterator Pattern

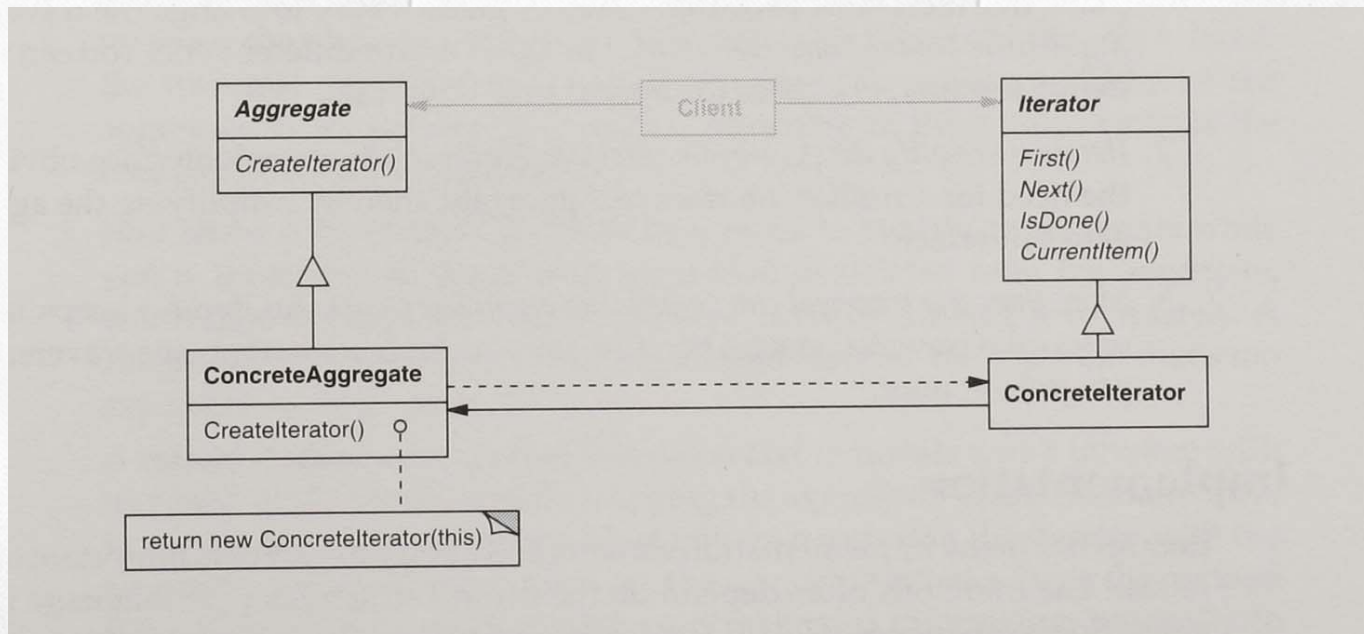
Motivation (Wann nutze ich das Pattern?):

Ich möchte alle Elemente einer Liste „durchgehen“, kenne aber den Typ der List nicht oder möchte ihn nicht spezifizieren.

Intent (Was ist das Design Ergebnis?):

Stelle einen Weg zur Verfügung auf alle Elemente einer Liste zuzugreifen, ohne die darunter liegende Listenstruktur zu kennen.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Factory Method Pattern

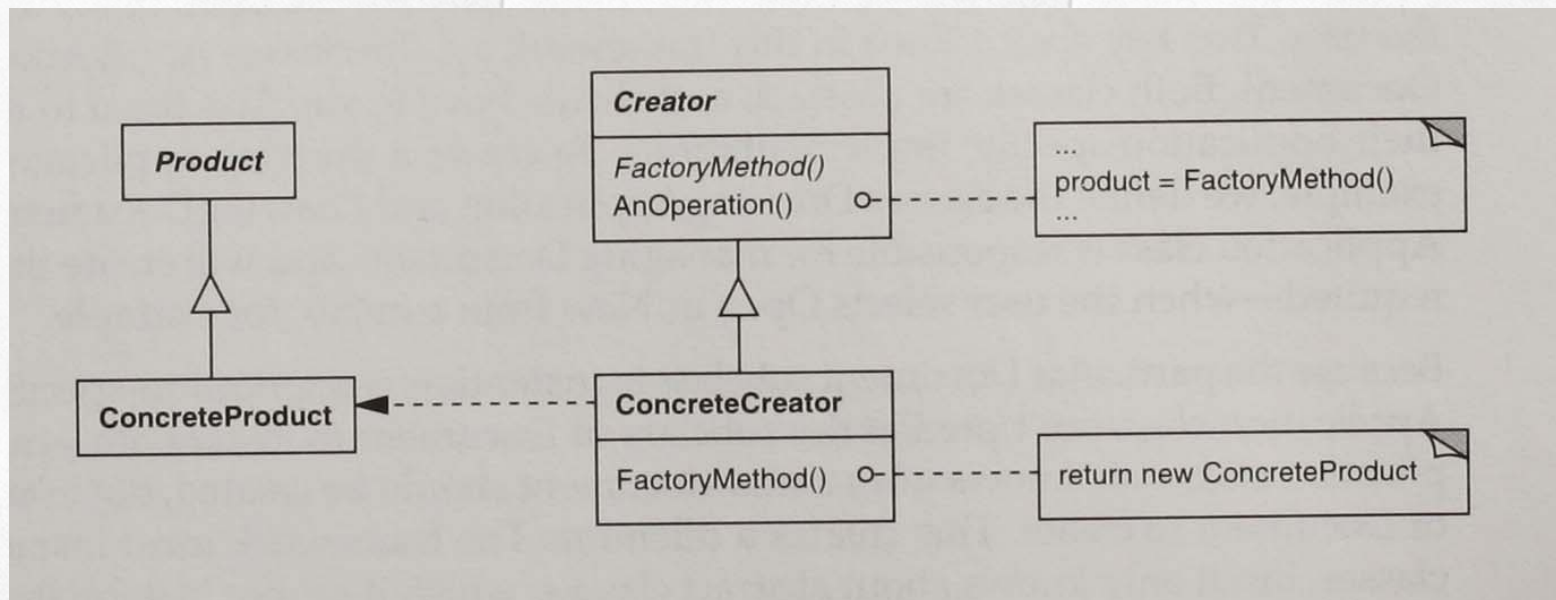
Motivation (Wann nutze ich das Pattern?):

Erst die Objekte von Subklassen wissen was sie instanzieren müssen. Dennoch liegt die Verarbeitung der instanziierten Objekte in der Superklasse.

Intent (Was ist das Design Ergebnis?):

Definiere ein Interface für die Objekterzeugung aber lasse die Subklassen entscheiden, welche Klassen sie instanzieren.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Decorator Pattern

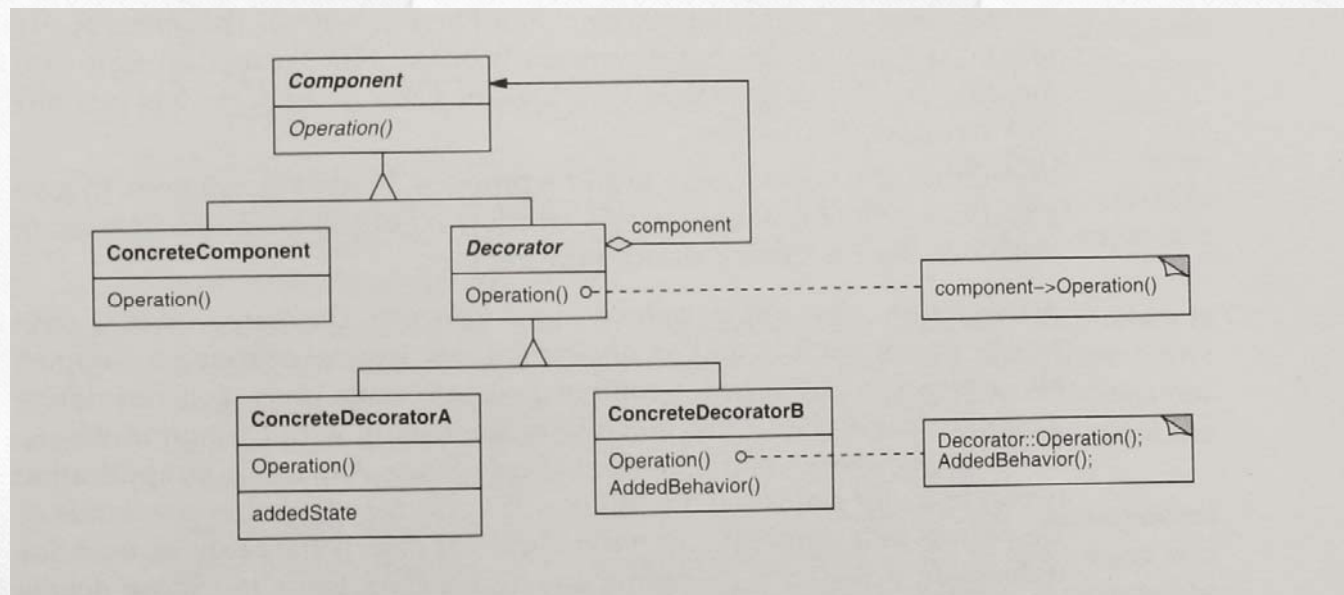
Motivation (Wann nutze ich das Pattern?):

Eine Objekt soll mit zusätzliche Verhaltensweisen zur Laufzeit ausgestattet werden. Z.B. Window mit Border mit Tilebar.

Intent (Was ist das Design Ergebnis?):

Füge einem Objekt zusätzliche Funktionalität dynamisch hinzu ohne die Klasse zu extenden zu müssen.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Observer Pattern

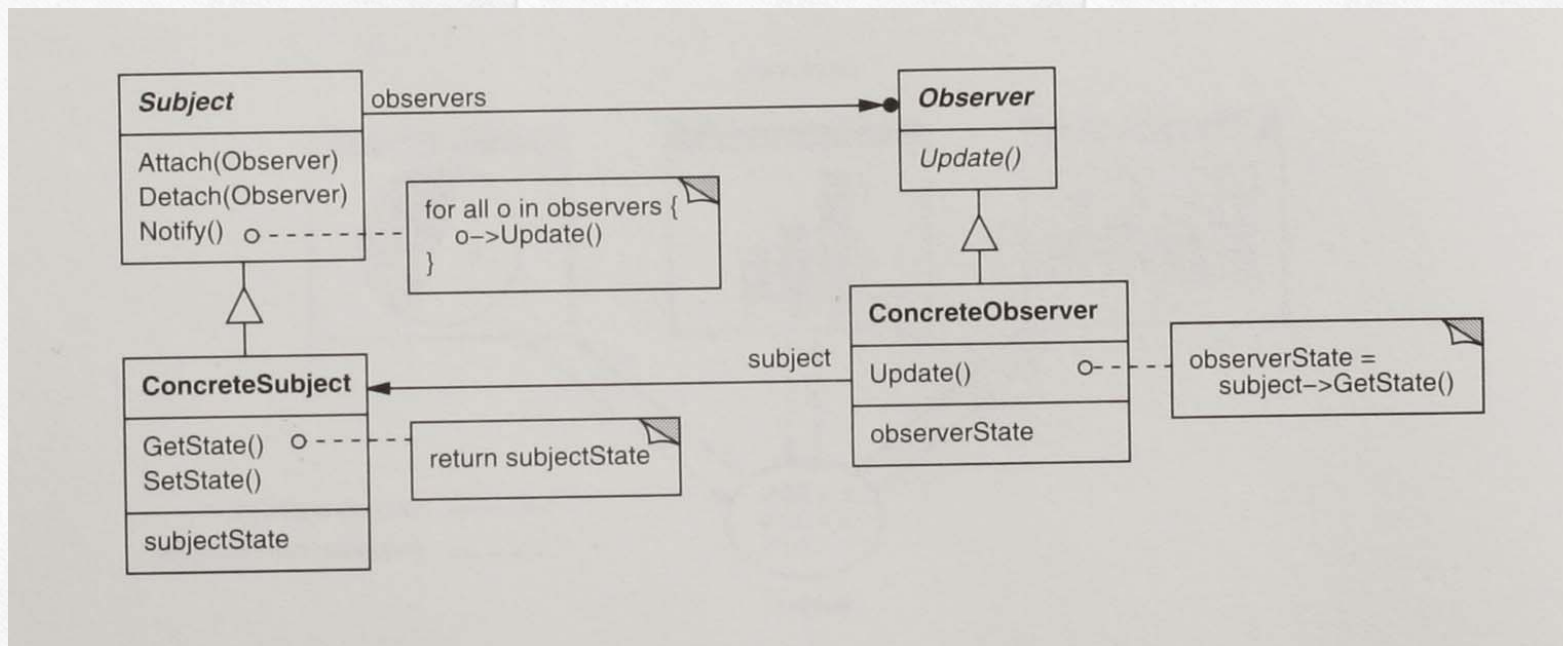
Motivation (Wann nutze ich das Pattern?):

Es solle eine Information gesendet werden, wobei es viele Empfänger geben kann und diese Empfänger unbekannt sind.

Intent (Was ist das Design Ergebnis?):

Definiere eine Eins-Zu-Viele Abhängigkeit zwischen Objekten so, dass wenn sich das eine Objekt ändert die anderen darüber informiert werden.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Adapter Pattern

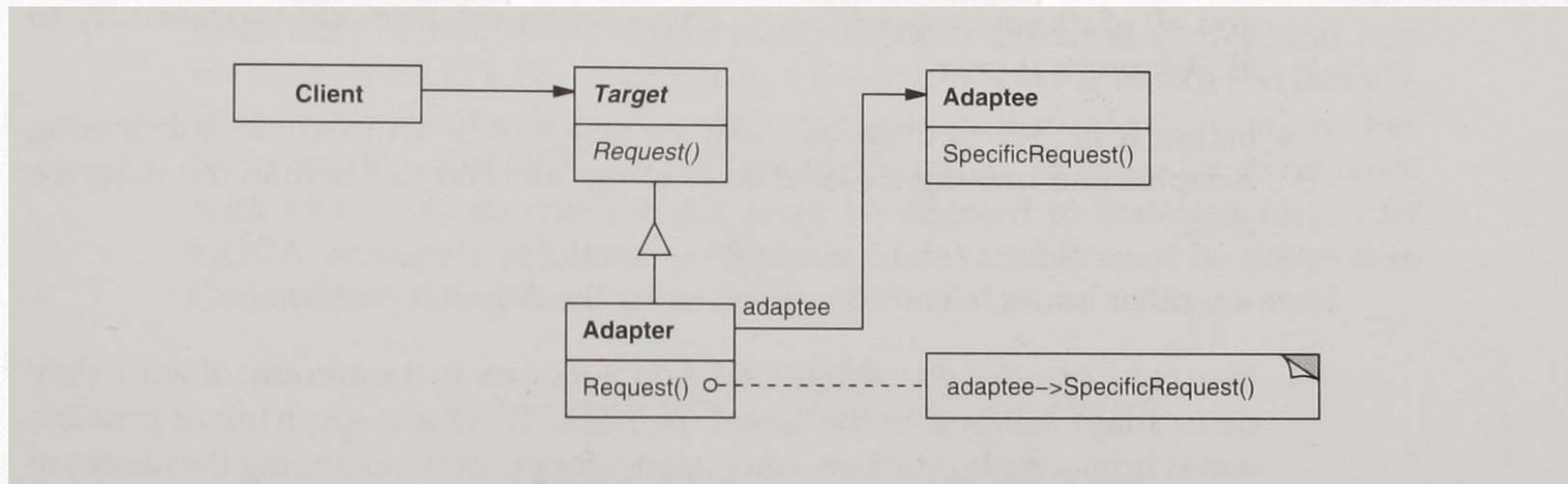
Motivation (Wann nutze ich das Pattern?):

Ein Teil der Software ist nicht Schnittstellen kompatibel zu einem anderen Teil.

Intent (Was ist das Design Ergebnis?):

Wandele das Interface einer Klasse um in eines das der Nutzer erwartet.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Command Pattern

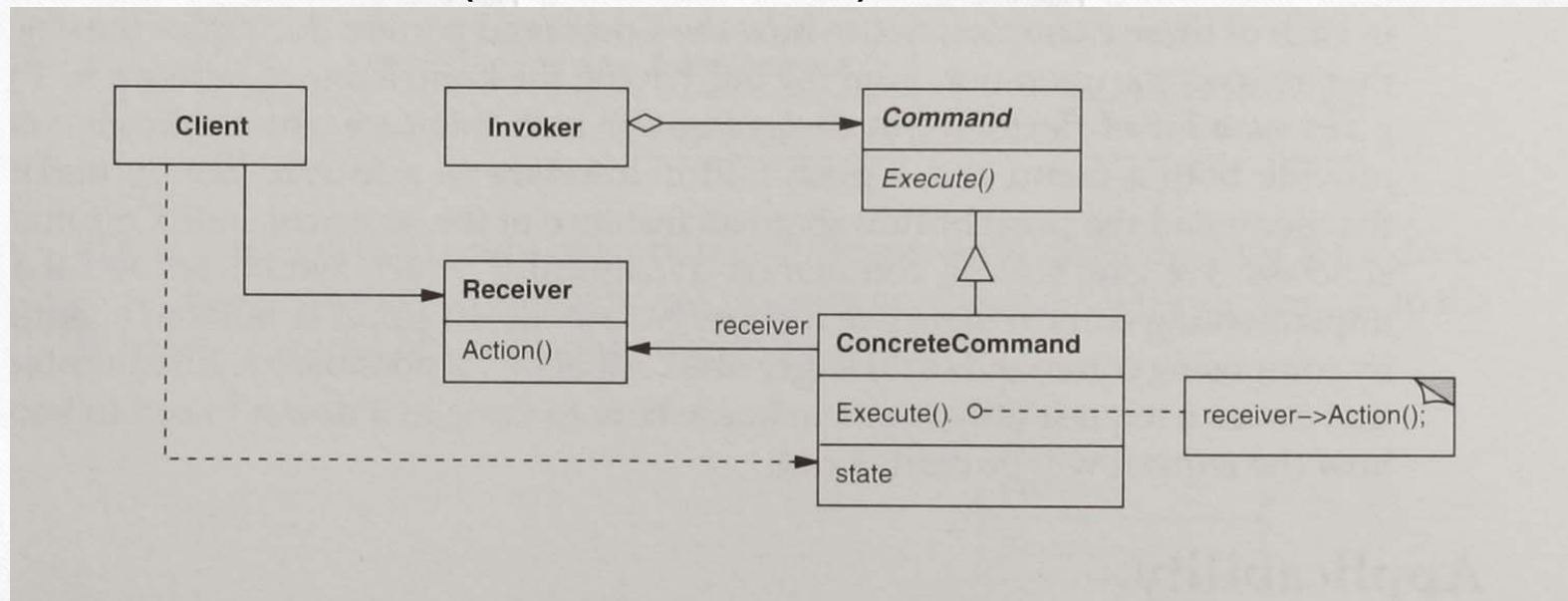
Motivation (Wann nutze ich das Pattern?):

Der Aufruf einer Methode soll „gespeichert“ werden, so dass der Aufruf später an beliebigen Objekten erfolgen kann oder der Nutzer entscheiden kann was ausgeführt wird.

Intent (Was ist das Design Ergebnis?):

Kapsle den Aufruf als ein Objekt und lass den Nutzer den Aufruf parametrisieren. Stapeln von Aufrufen und Undo-Operationen sind möglich.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Composite Pattern

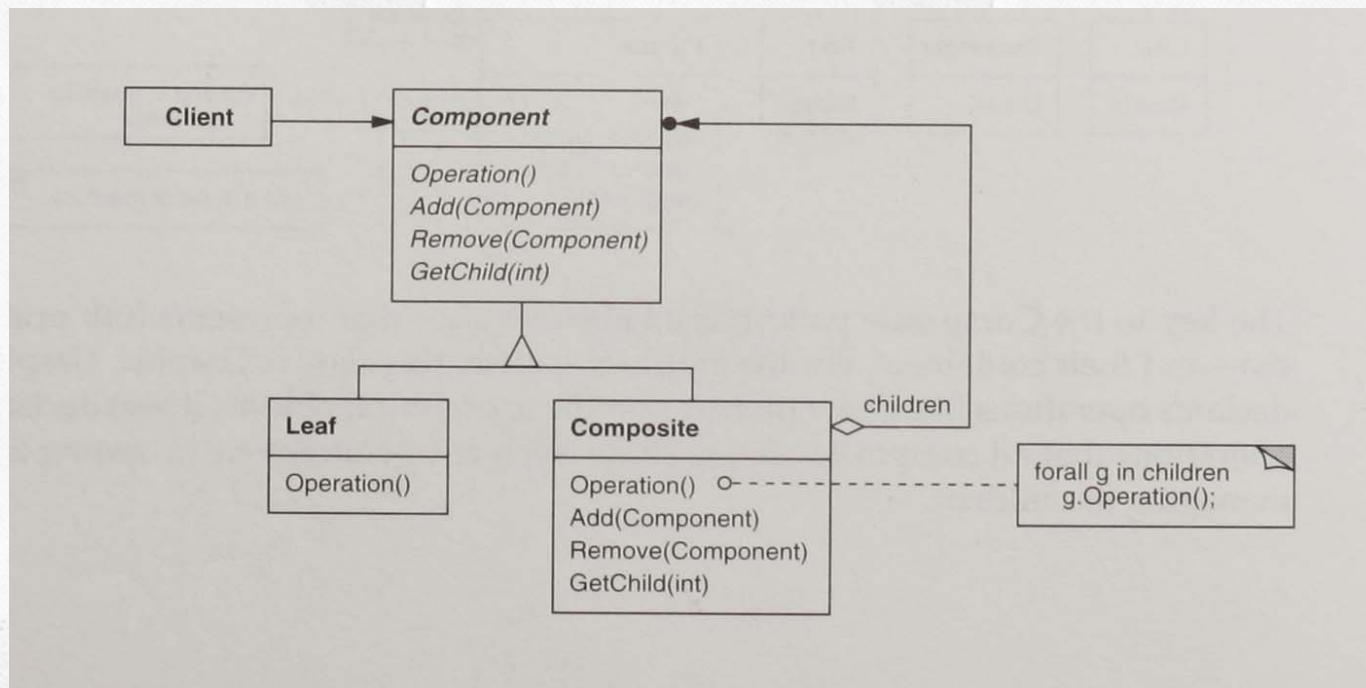
Motivation (Wann nutze ich das Pattern?):

Es soll Kommunikation durch einen Baum stattfinden.

Intent (Was ist das Design Ergebnis?):

Verbinde Objekte zu einer Baumstruktur. Behandle Knoten wie Blätter.

Wie baue ich das Pattern? (Structure/ Teilnehmer):



Spicefactory – www.spicefactory.org

- Founder and Lead Developer Jens Halm
- Open Source

The screenshot displays the Spicefactory website with a central gear icon. The page is organized into several sections:

- PARSLEY**: AS3 Application Framework
Configuration / Dependency Injection • Localization • MVC Framework
Current Release: Version 1.0.0
- CINNAMON**: AS3 - Java Remoting
• For Flex and Pure AS3 Clients
• Deployable in every Servlet Container
Current Release: Version 0.2.0
- SPICELIB**: AS3 Library
Task Framework • Reflection, Logging •
Current AS3 Version: 1.0.1
Current Java Version: 1.0.0
- SERVICES**: Consulting, Training
Icons for UK and Germany
- Upcoming projects:**
 - PIMENTO**: Data and Media Management for Flex and AS3 Clients
Initial preview release scheduled for Q3 2008
- Older projects:**
 - OREGANO**: Java Multiuser Server for AS1/AS2 Clients
Final Release: Version 1.1.0

NEWS

2008-04-30: FFK-08 Conference in Cologne, May 21-24
2008-04-26: Spicelib AS3 and Java Maintenance Releases
2008-03-23: Cinnamon 0.2.0 adds AS3 Source Generator
2008-03-11: Parsley 1.0.0 released

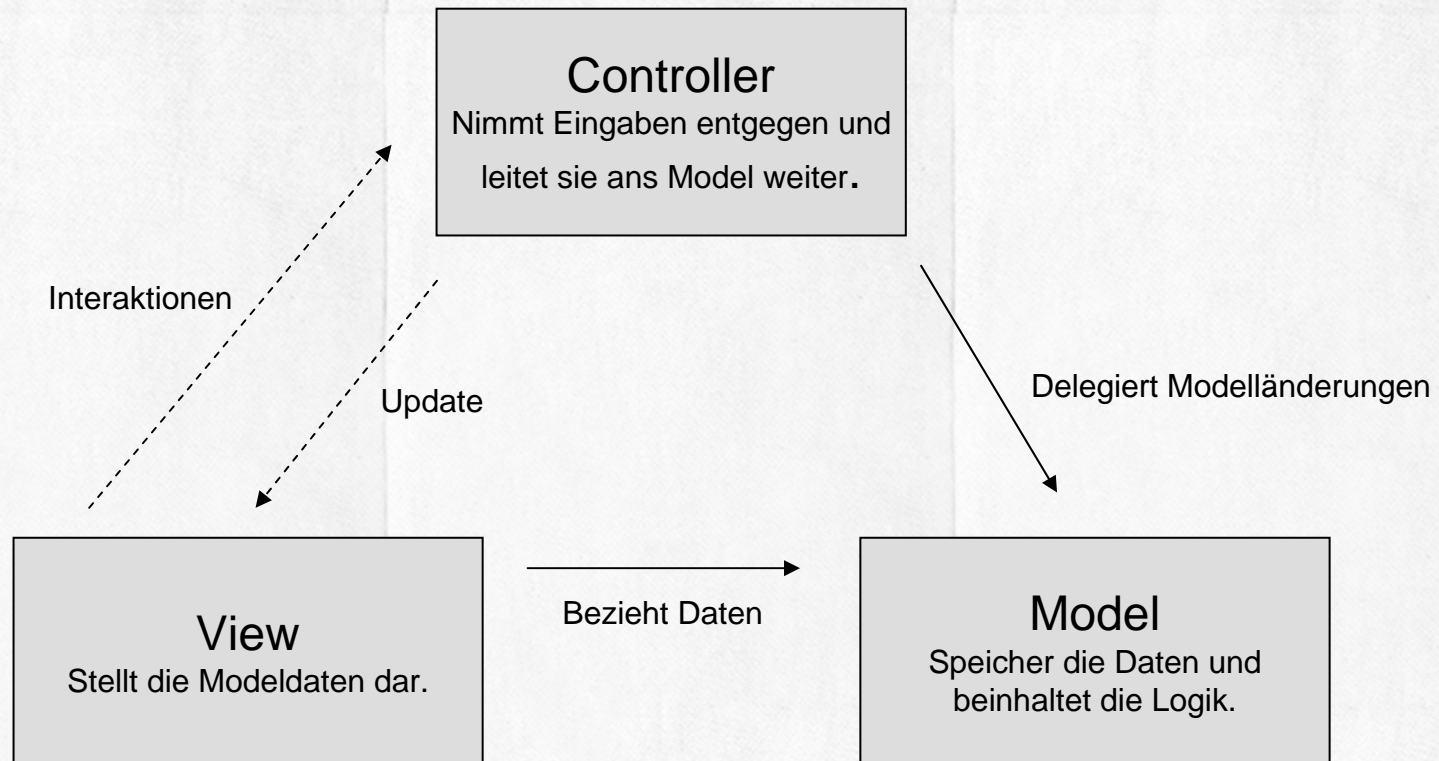
[More News...](#)

SPICEFACTORY

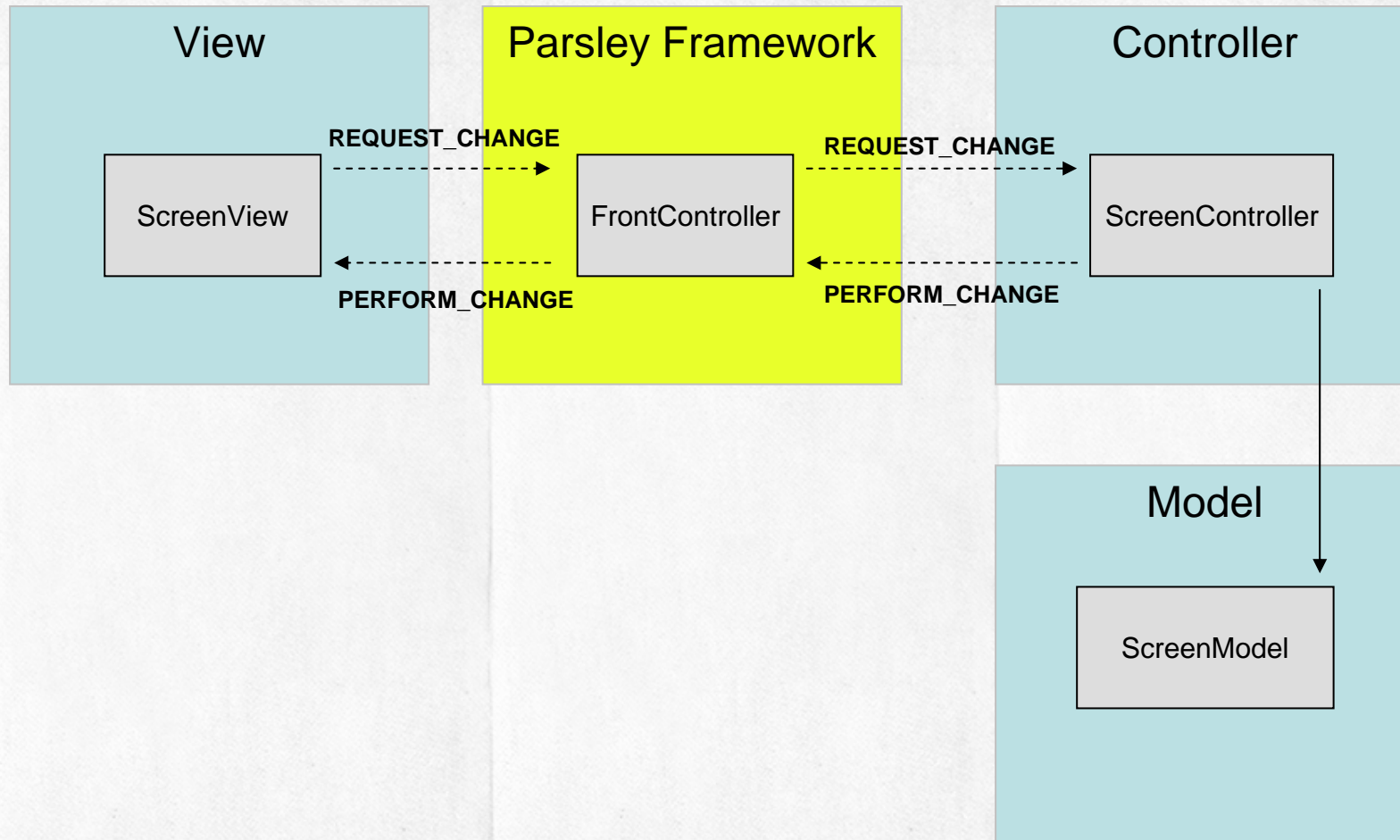
Open Source Software for Rich Internet Applications
built with Flex/Flash/AIR Clients and Java EE Servers

FORUM **CONTACT**

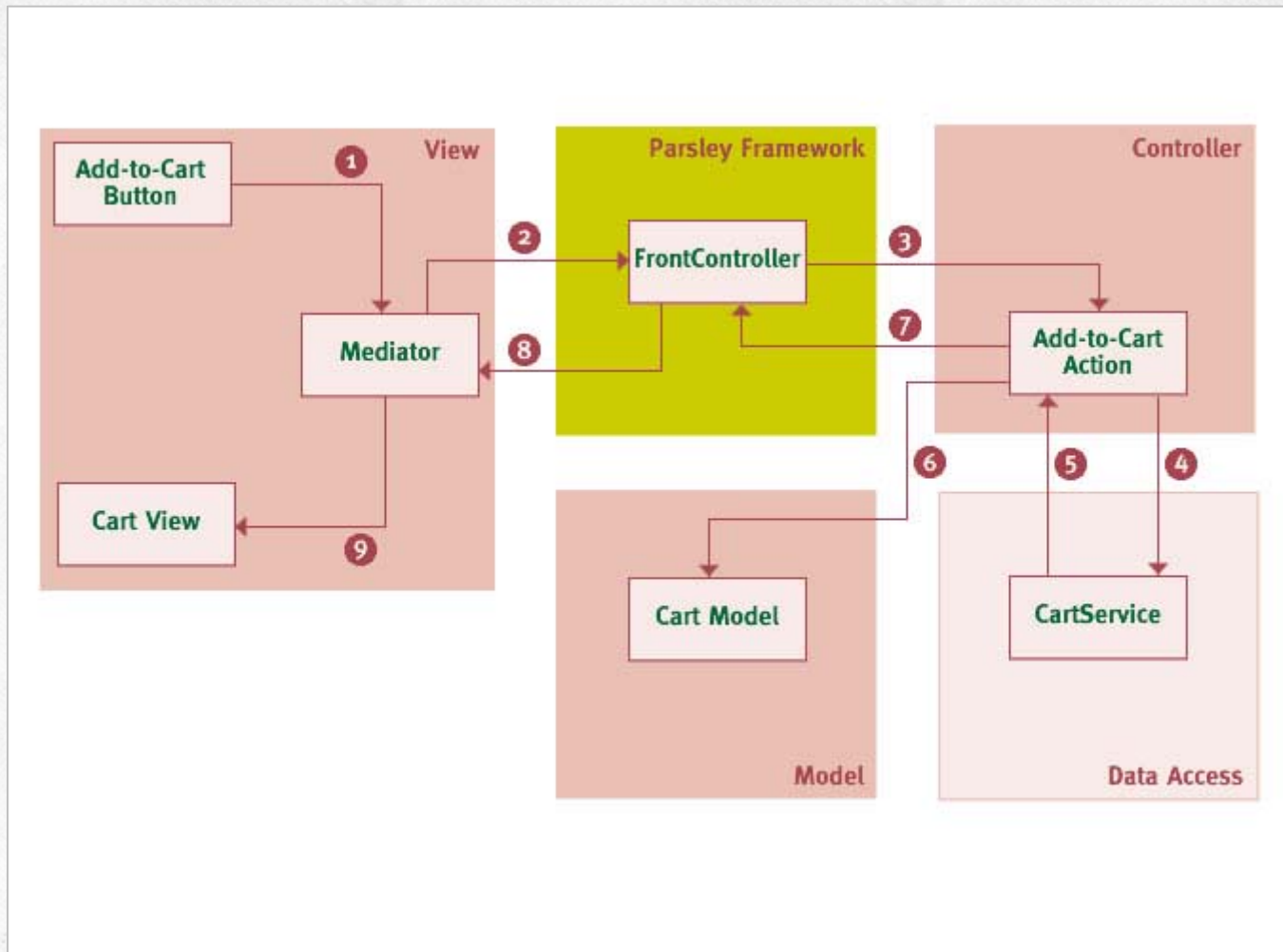
MVC mit Parsley



MVC mit Parsley und Frontcontroller



MVC mit Parsley



Vielen Dank für Eure Aufmerksamkeit!

Nico Zimmermann – Powerflasher

nz@powerflasher.de

Belvedereallee 5
52070 Aachen