Fassade

Objektbasiertes Strukturmuster

C. Restorff & M. Rohlfing

Übersicht

- Motivation
- Anwendbarkeit
- Struktur
- Teilnehmer
- Interaktion
- Konsequenz
- Implementierung
- Beispiel
- Bekannte Verwendung
- Verwandte Muster

Motivation I

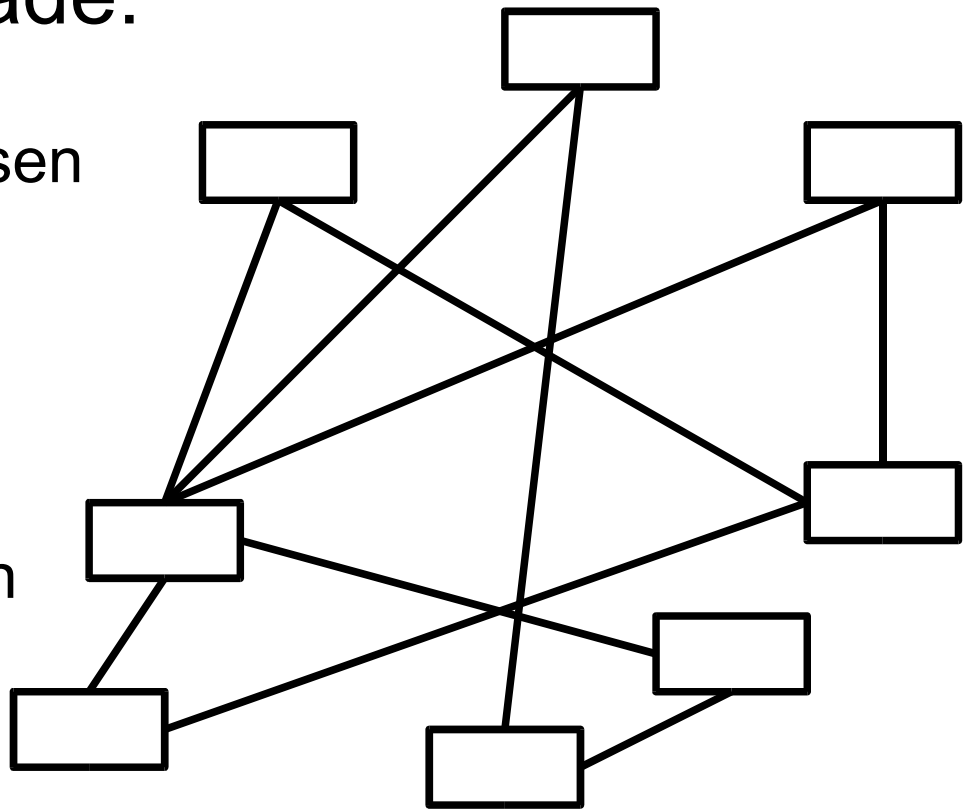
- Motivation:
  - Einheitliche Schnittstelle zu einer Menge von Schnittstellen
  - Verringerung der Komplexität des Systems durch Unterteilung in Subsysteme
  - Vereinfachte Nutzung des Subsystems
  - Minimierung der Abhängigkeiten und der Kommunikation

Motivation II

- Ohne Fassade:

Klientenklassen

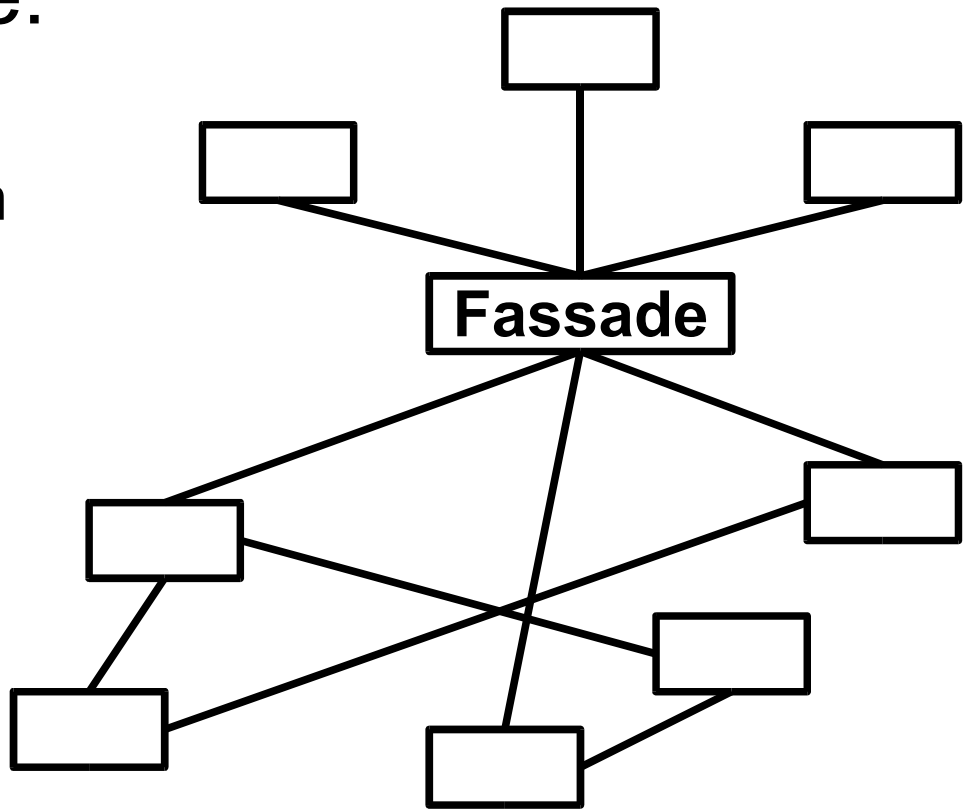
Subsystemklassen



Motivation III

- Mit Fassade:

Klientenklassen



Subsystemklassen

Anwendbarkeit I

- Verwendung der Fassade wenn:
  - Eine einfache Schnittstelle zu komplexen Subsystemen geboten werden soll
    - Bei Anwendung anderer Muster entstehen meist kleinere Klassen
    - → Für Klienten einfacher anzupassen
    - → Aber: schwerer zu verwenden
    - → Erhöht Komplexität
  - → Fassade als voreingestellte Sicht genügt den meisten Klienten

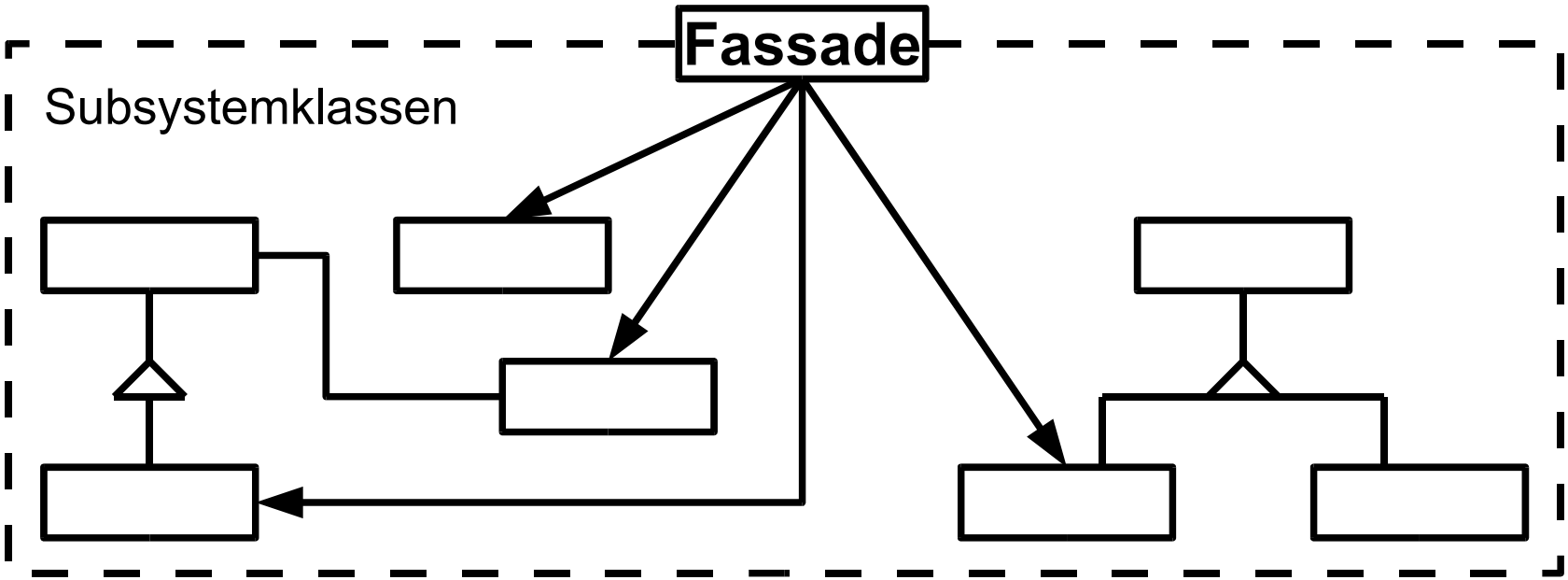
Anwendbarkeit II

- Verwendung der Fassade wenn:
  - Viele Abhängigkeiten zwischen Klienten und Implementierungsklassen bestehen
    - Fassade entkoppelt Subsysteme vom Klienten und anderen Subsystemen
  - Unabhängigkeit und Portabilität des Subsystems wird gefördert

Anwendbarkeit III

- Verwendung der Fassade wenn:
  - Die Subsysteme in Schichten aufgeteilt werden sollen
    - Subsysteme können voneinander abhängen
  - → Vereinfachung der Abhängigkeiten durch Verwendung der Fassade als Kommunikationsschnittstelle zwischen einzelnen Subssystemen / -Schichten

Struktur



Teilnehmer I

- Fassade
  - Weiß, welche Subsystemklasse für eine Anfrage zuständig ist
  - Delegiert Anfragen der Klienten an das entsprechende Objekt des Systems

Teilnehmer II

- Subsystemklassen
  - Implementiert die Funktionalität
  - Führen die Aufgaben aus die ihnen von der Fassade zugewiesen wurden
  - Haben keine Referenz auf Fassade, wissen also nichts von ihr

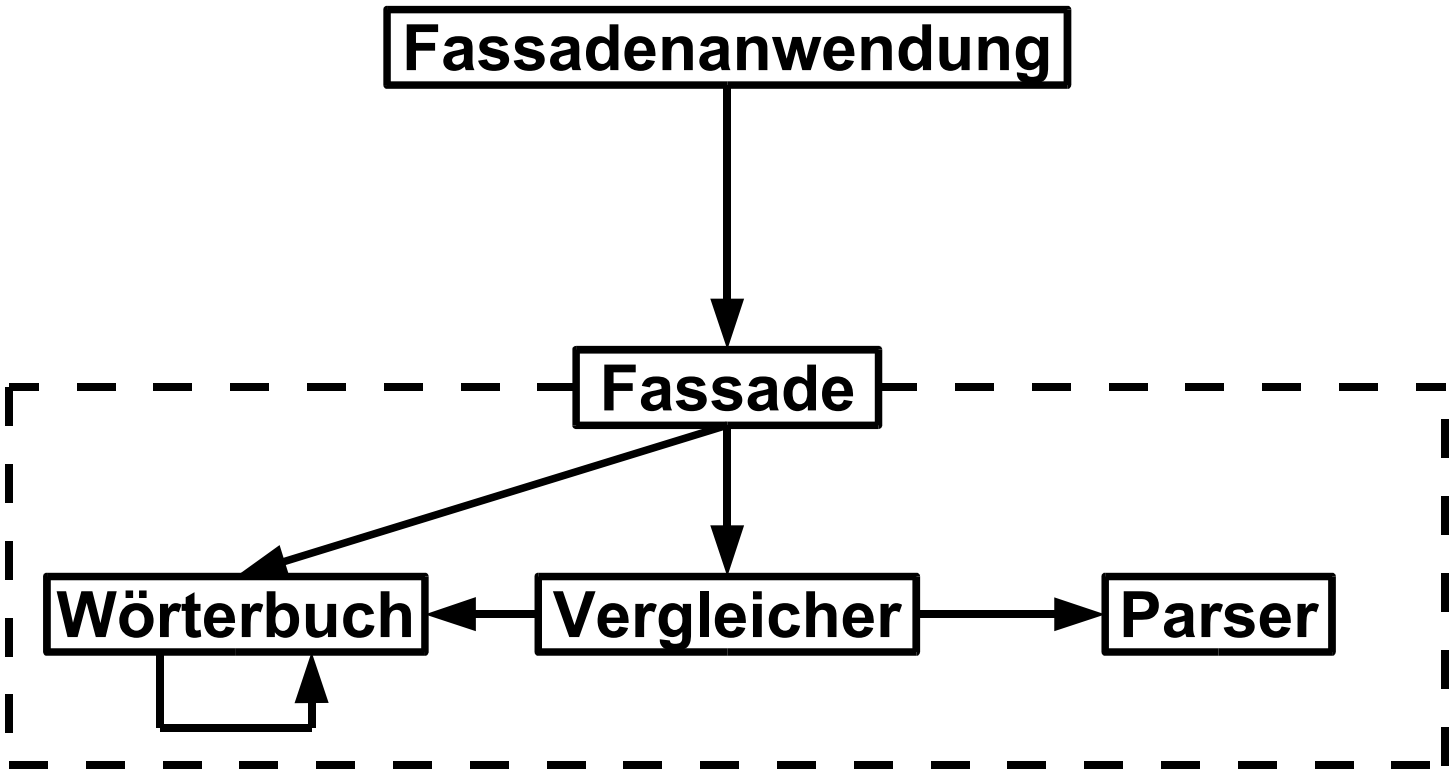
Interaktion I

- Klienten schicken Anfragen an die Fassade
- Fassade leitet Anfragen an Subsystemobjekte weiter
- Fassade muß unter Umständen selber Arbeit verrichten um ihre Schnittstelle auf Schnittstellen der Subsystemobjekte abzubilden

Interaktion II

- Klienten müssen durch Benutzung der Fassade ihre Subsystemobjekte nicht selber benutzen
- Ausnahme: Blick hinter die Fassade
  - Klient kann auch auf das Subsystemobjekt direkt zugreifen, wenn dies erwünscht oder erforderlich ist

Beispiel


Fassadenanwendung
Fassade
Wörterbuch
Vergleicher
Parser

Konsequenzen I

- Klienten werden von Subsystemklassen abgeschirmt
  - Reduktion der Anzahl der Komponenten, die vom Klienten gehandhabt werden müssen
  - Subsystem wird einfacher in der Benutzung

Konsequenzen II

- Fördert lose Kopplung zwischen Subsystem und seinen Klienten
  - Komponenten des Subsystems können ausgetauscht werden ohne Klienten in Mitleidenschaft zu ziehen
  - Durch Reduktion von Übersetzungsabhängigkeiten entsteht Zeitersparnis
  - Portierbarkeit wird verbessert

Konsequenzen III

- Verhindert nicht, das Anwendungen Subsystemklassen verwenden, wenn sie das müssen (Blick hinter die Fassade)
  - Entscheidung zwischen Einfachheit der Verwendung und allgemeiner Verwendbarkeit

Implementierung I

- Folgende Überlegungen müssen in Betracht gezogen werden
  - Die Kopplung zwischen Klienten und Subsystemklassen kann noch weiter reduziert werden indem die Fassade abstrakt definiert wird und konkrete Fassaden abgeleitet werden

Implementierung II

- Folgende Überlegungen müssen in Betracht gezogen werden
  - Wie öffentlich oder privat sollen Subsystemklassen gehalten werden
  - Auf einige Klassen kann von außen zugegriffen werden
  - Sinnvoll: Zugang beschränken

Bekannte Verwendungen

- ODBC/ JDBC
- ObjectWorks/ Smalltalk Übersetzer

Verwandte Muster I

- Abstrakte Fabrik
  - Kann als Alternative zur Fassade verwendet werden
  - Kann mit Fassade zusammen zur Erzeugung von Subsystemobjekten in vom Subsystem unabhängiger Weise verwendet werden

Verwandte Muster II

- Vermittler
  - Abstrahiert wie Fassade von Funktionalität existierender Klassen
  - Dient allerdings der Kommunikation von miteinander in Beziehung stehenden Objekten, die den Vermittler auch kennen
- Singleton
  - Da im allgemeinen nur eine Fassade benötigt wird ist sie meist ein Singleton