

Entwerfen lernen – Entwurfsmuster in der Informatik-Ausbildung

Karlheinz Hug

Folien zum Bewerbungsvortrag für die C3-Professur
„Informatik, Softwaretechnologie und Betriebssysteme“
im Fachbereich Informatik, Hochschule Reutlingen

24. November 2003

Entwerfen lernen

- Software-Entwurfsproblem
- Entwurfsmetaphern & -muster
- Entwurfsmuster in Informatik 2++
 - Testszenarium
 - Behälterklasse Menge
 - Konzeptklassen
 - Model-View-Controller
 - Carrier-Rider-Mapper
 - Message Queuing System
 - Cleo-Übersetzer

Software-Entwurfsproblem – negativ

Manche Software-Systeme sind

- ☹️ schwer zu ändern,
weil Änderungen viele Teile betreffen
- ☹️ instabil nach Änderungen,
weil diese Unerwartetes bewirken
- ☹️ schwer wiederzuverwenden,
weil die Teile zu stark verflochten sind
- ☹️ ⇒ schlecht entworfen

Schwache Entwürfe sind

- ☹️ kaum wiederverwendbar
- ☹️ kaum lehrbar, es sei denn zur Abschreckung

Software-Entwurfsproblem – positiv

Gute Software-Systeme haben

☺ die *[bar]keiten

- Zuverlässigkeit
- Benutzbarkeit
- Wartbarkeit
- Änderbarkeit
- Erweiterbarkeit
- Wiederverwendbarkeit

☺ ein elegantes Design,
das zu lehren & lernen lohnt

Software-Entwurfsproblem

Ziel

Den reichen Erfahrungsschatz
guter Software-Architekten
mit guten Software-Entwürfen
vermitteln, lehren, lernen

Wie erreichen?

Software-Entwurfsproblem

Wichtig, aber nicht ausreichend

- **Algorithmen**
 - ☹ nur Atome in Entwürfen
- **Bibliotheken**
 - ☹ nur Fundamente in Implementationen
- **Frameworks**
 - ☹ nur Gerüste für Anwendungen
- **Methoden**
 - ☹ nur prozess- oder dokumentorientiert

Software-Entwurfsproblem

Die Lücke schließen

- Entwurfsmetaphern
- Entwurfsmuster

Entwurfsmetaphern – Merkmale

- Beschreiben Software-Konzepte & -Komponenten durch Dinge des Alltags
- Beziehen sich verständlich & anschaulich auf allgemeine Erfahrungen
- Strukturieren das Wahrnehmen von Problemen
- Leiten das Vorstellen & Vermitteln von Lösungsentwürfen
- Dienen dem Gestalten von Entwurfsmustern

Entwurfsmuster – Merkmale

- Beschreiben abstrakt & strukturiert oft vorkommende Probleme & ihre Lösungen
- Erfassen Entwurfsentscheidungen, die erfahrungsgemäß gut funktionieren
- Sind Moleküle in Entwürfen = Mikroarchitekturen
- Bilden Makromoleküle = Miniarchitekturen
- Formen Sprache, um Entwürfe zu diskutieren
- Ergänzen Entwurfsmethoden
 - Methode ↔ Prozess & Dokument
 - Entwurfsmuster ↔ Produkt
- Sind wiederverwendbar
- Müssen an konkrete Probleme angepasst werden

Entwurfsmuster – Literatur 1

Viererbande (Gang of Four – GoF):

Gamma/Helm/Johnson/Vlissides:

*Entwurfsmuster - Elemente
wiederverwendbarer objektorientierter
Software (Design Patterns)*

- Seit 1995 viele Auflagen
- Klassiker
- Katalog von 23 Entwurfsmustern
 - 5 Erzeugungsmuster
 - 7 Strukturmuster
 - 11 Verhaltensmuster

Entwurfsmuster

Literatur 2

Jézéquel/Train/Mingins:

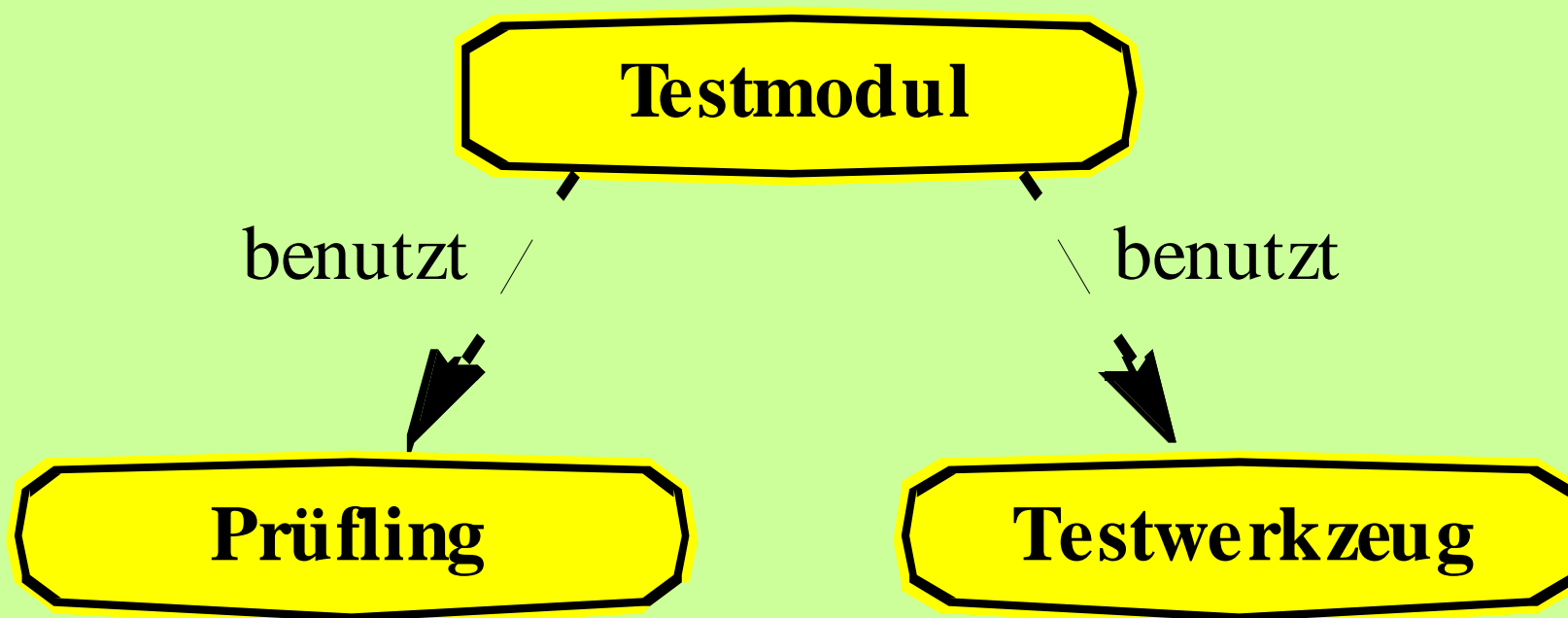
Design Patterns and Contracts (2000)

- Kombiniert zwei gute Ideen
- Spezifiziert alle GoF-Entwurfsmuster durch abstrakte Verträge
- Entwurfsmuster sind exakter & verständlicher
- Code-Schablonen sind wiederverwendbar

Entwurfsmuster in der Lehre

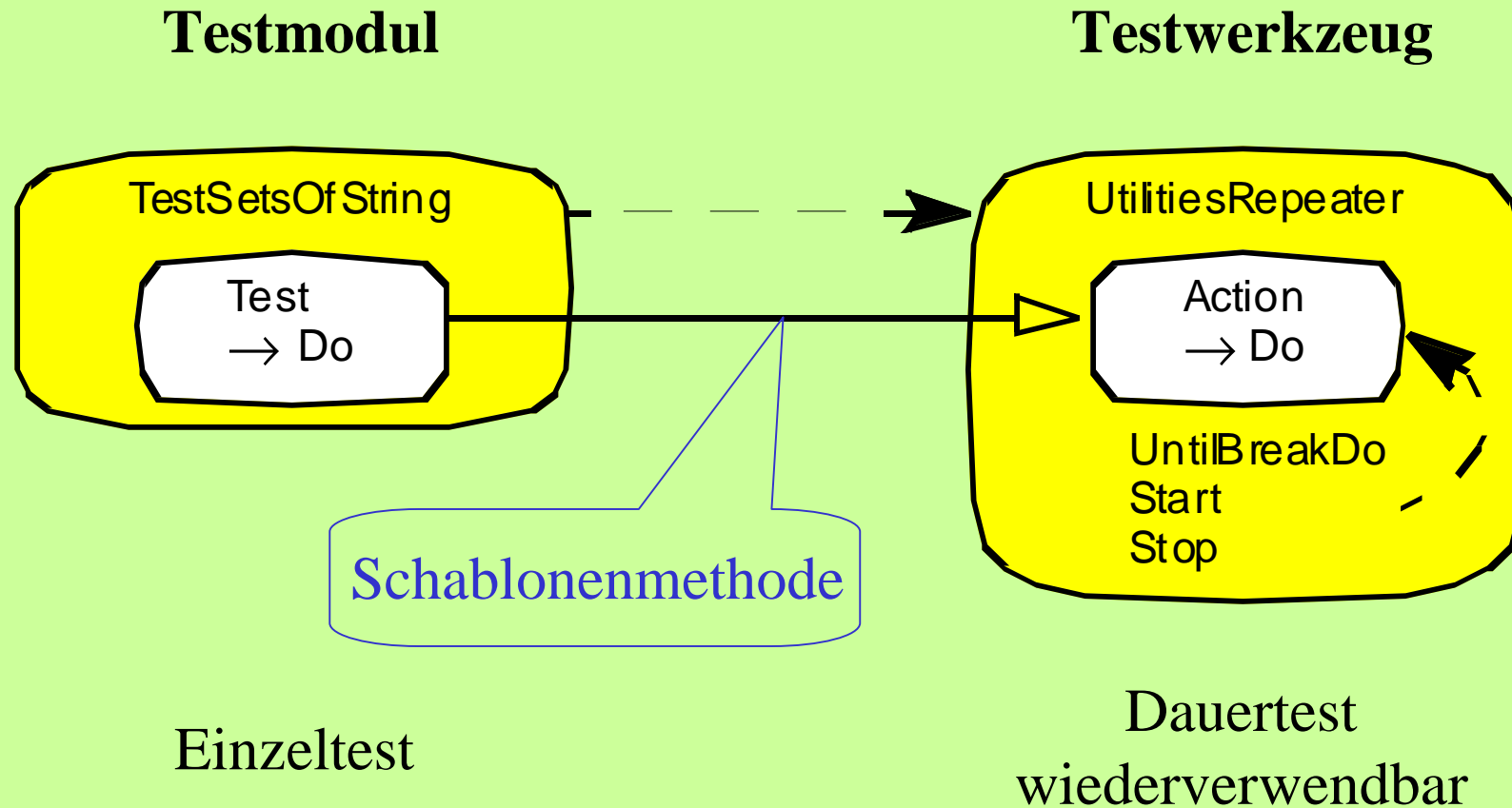
- Gamma: „Teach how to emulate good design decisions from good architects“
- Zielkonflikt:
 - Entwurfsmuster früh in die Ausbildung einbringen
 - Manche GoF-Entwurfsmuster für Anfänger ungeeignet
- Passende Entwurfsmuster suchen
- Passende Beispiele suchen

Testszenarium Moduldiagramm



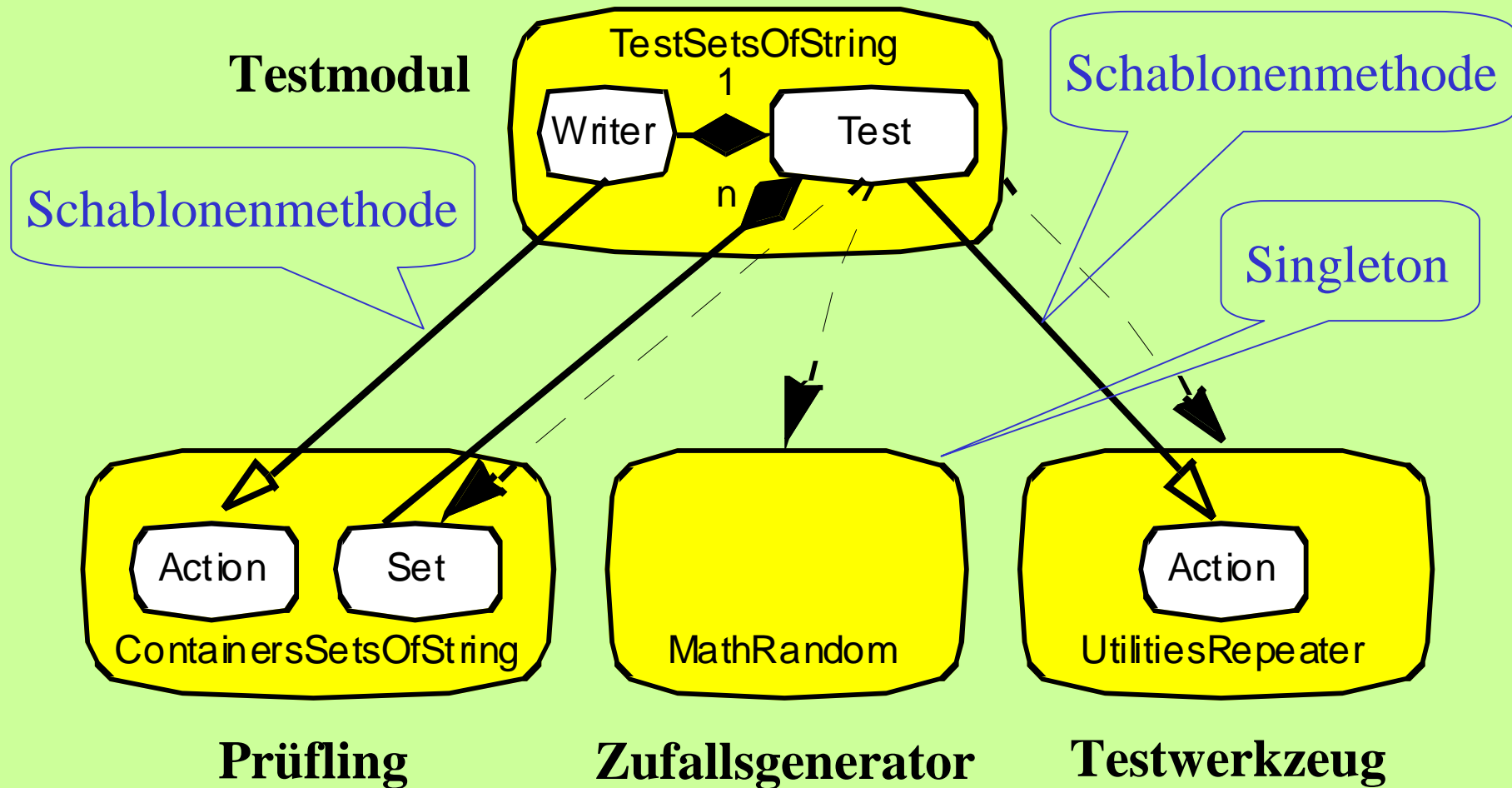
Testszenarium

Entwurfsmodell für Testmodul & -werkzeug



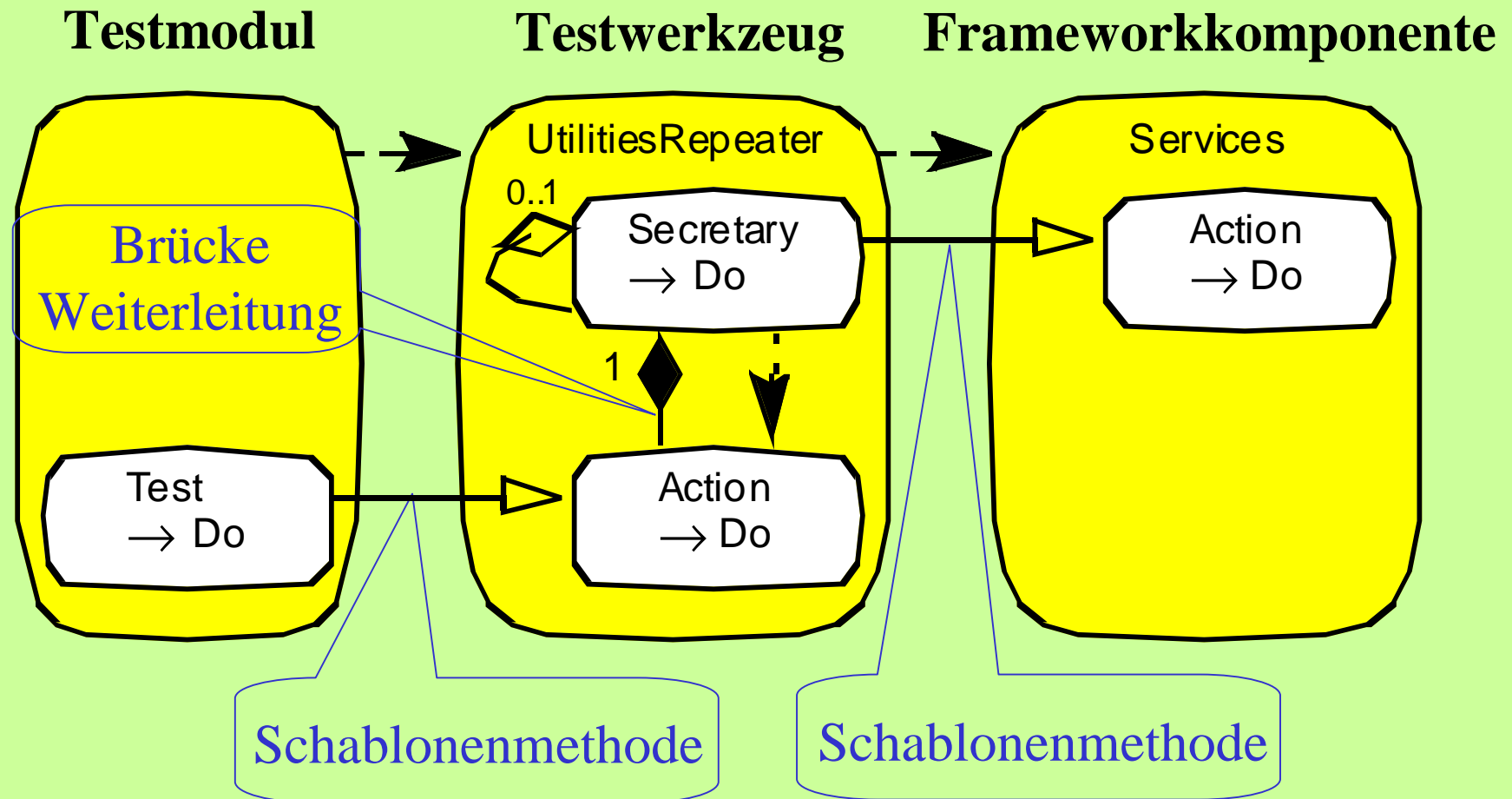
Testszenarium

Entwurfmodell für Testmodul



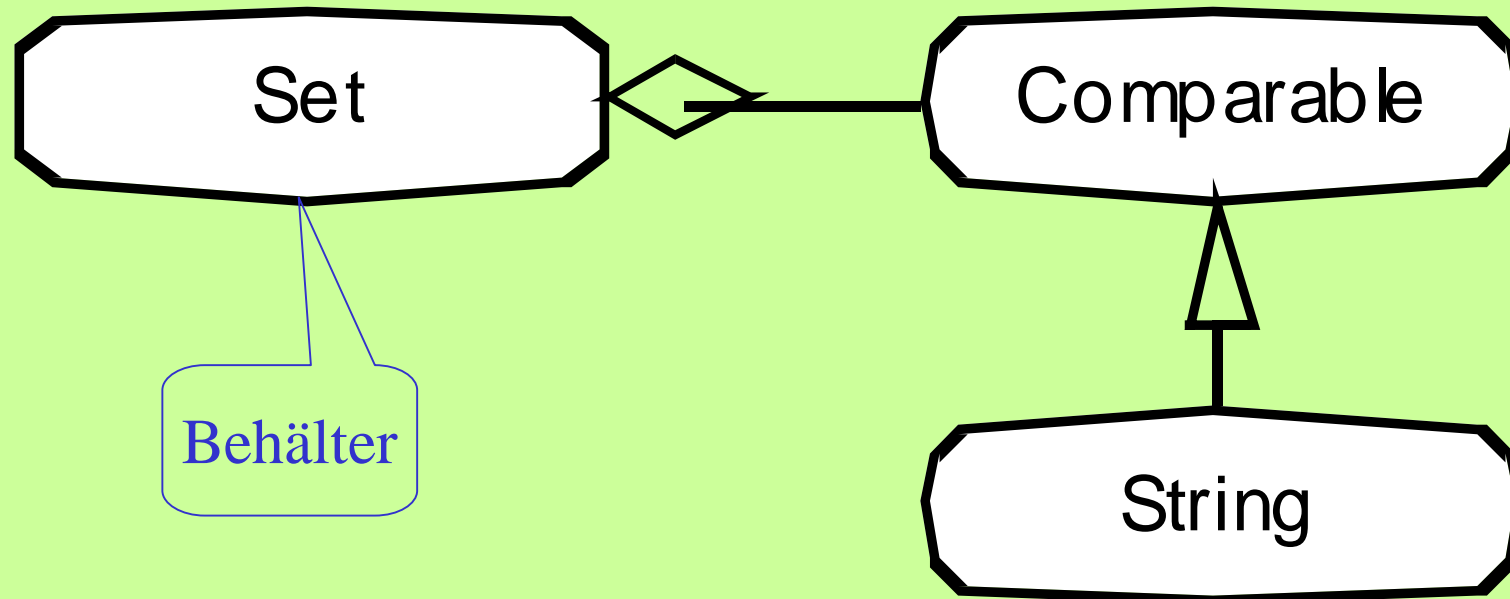
Testszenarium

Entwurfsmodell für Testwerkzeug



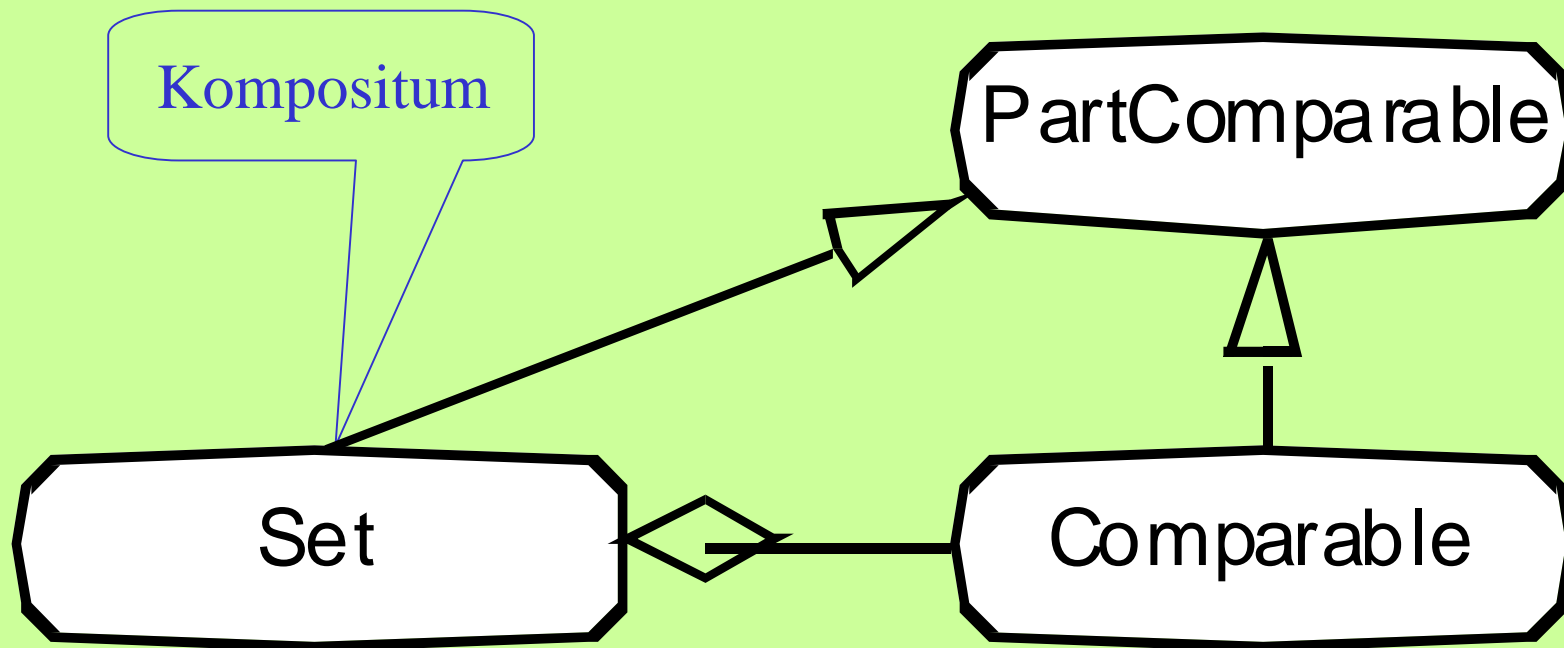
Behälterklasse Menge

Polymorphe linear geordnete Elemente

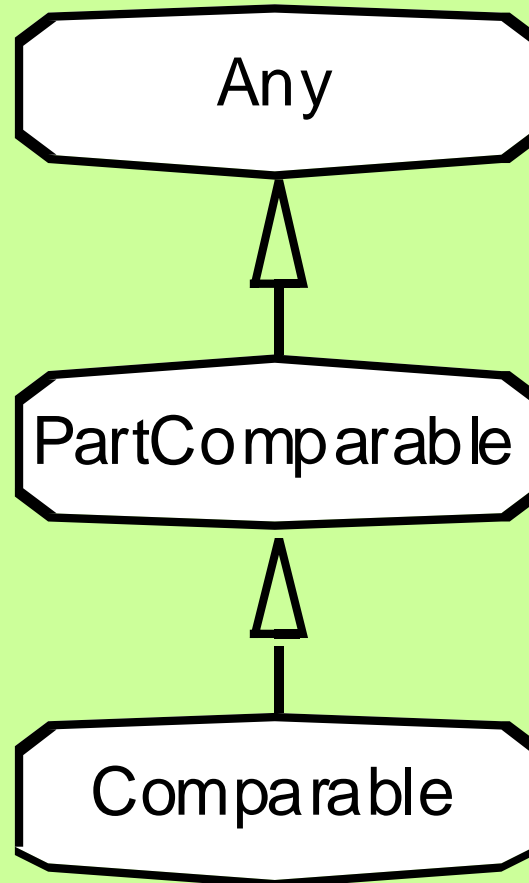


Behälterklasse Menge

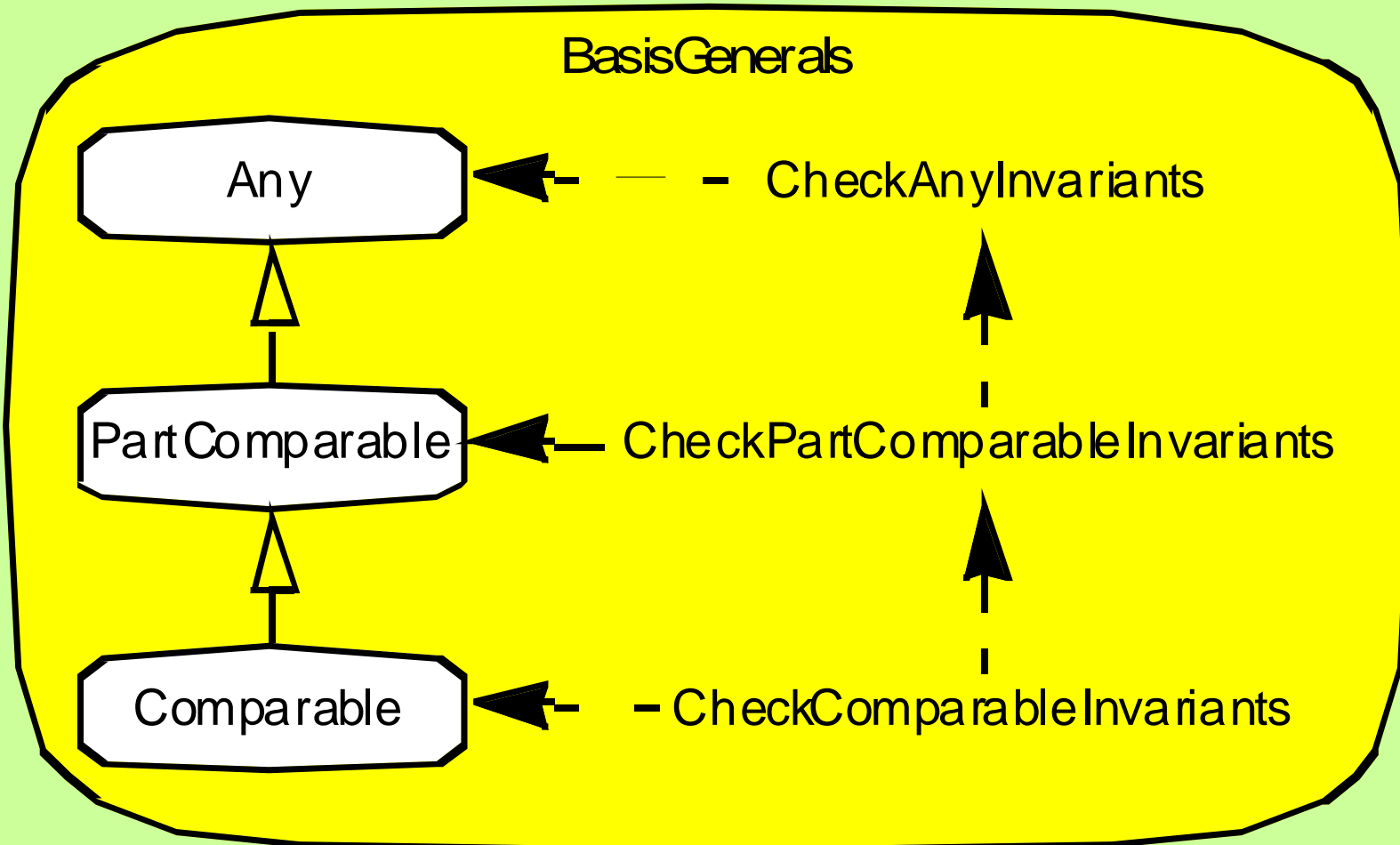
Abstraktion der partiellen Ordnung



Konzeptklassen – Hierarchie

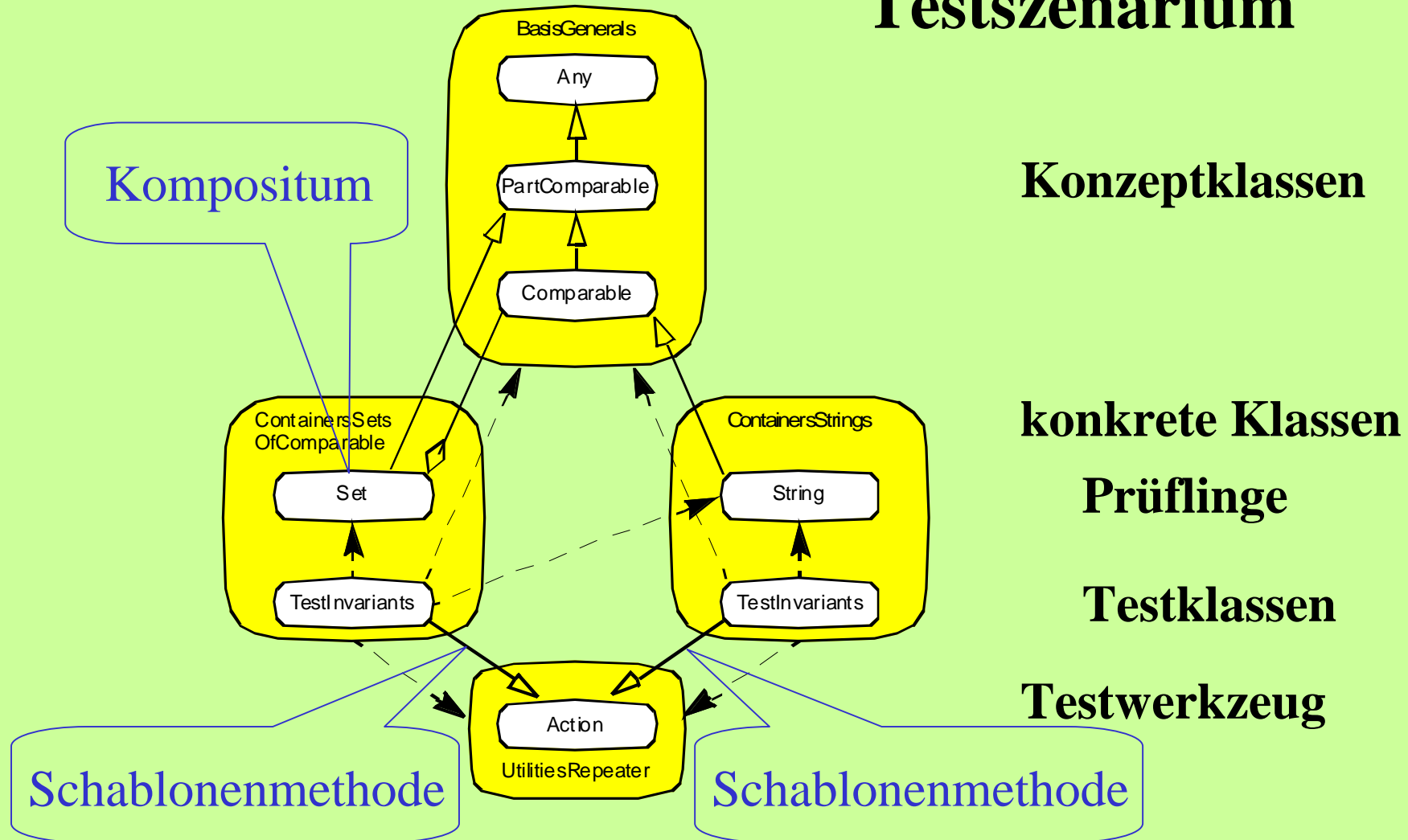


Konzeptklassen Spezifikation & Test



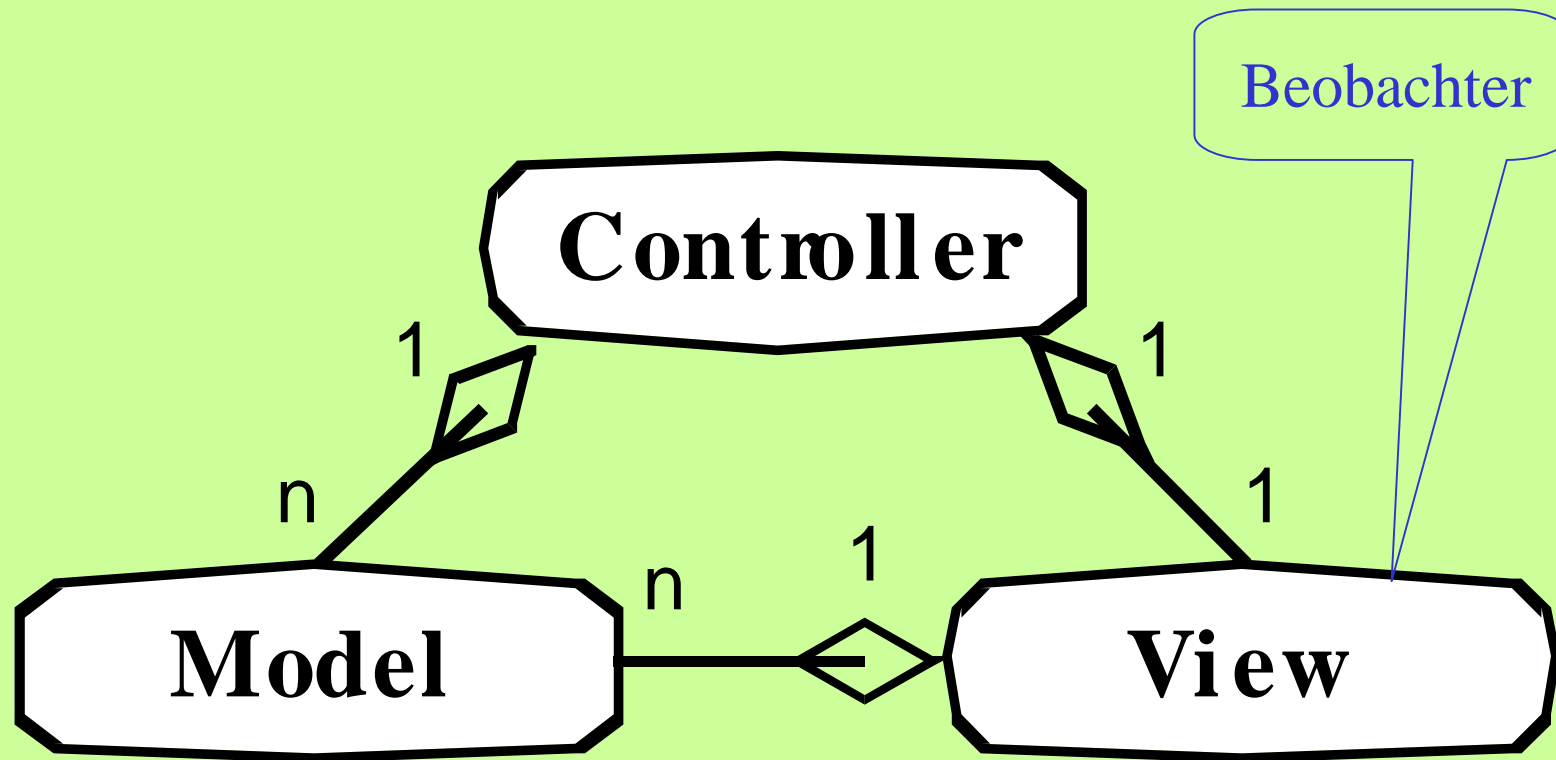
Konzept- & konkrete Klassen

Testszenarium



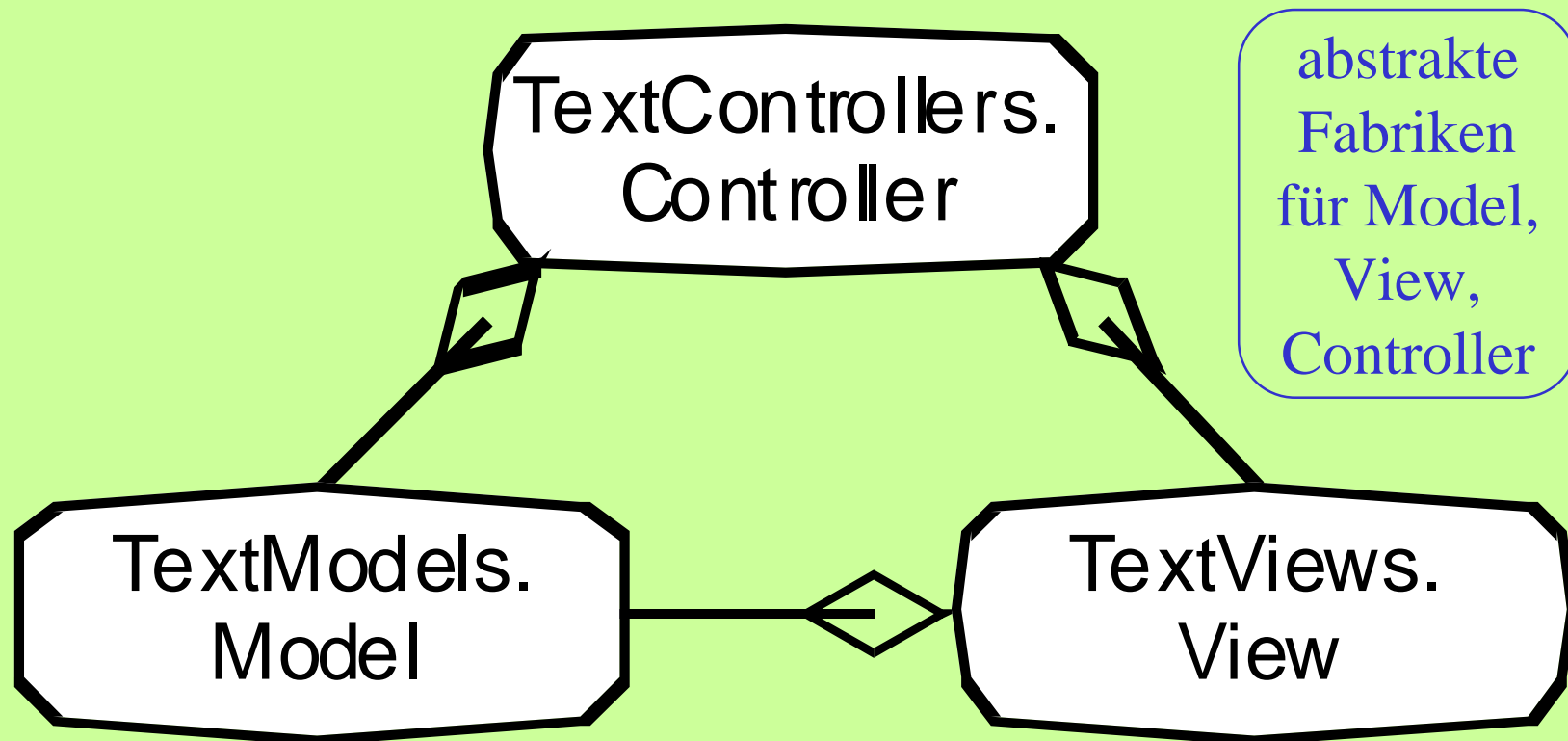
Model-View-Controller

Allgemein

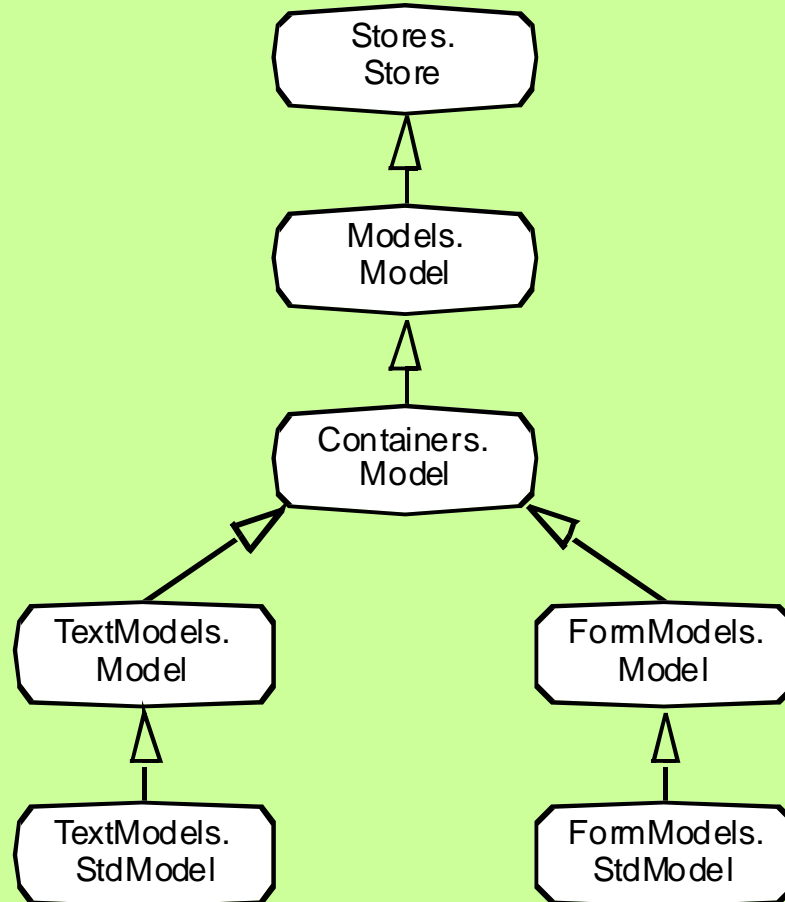


Model-View-Controller

Text in BlackBox



Model-View-Controller Modell-Klassenhierarchie



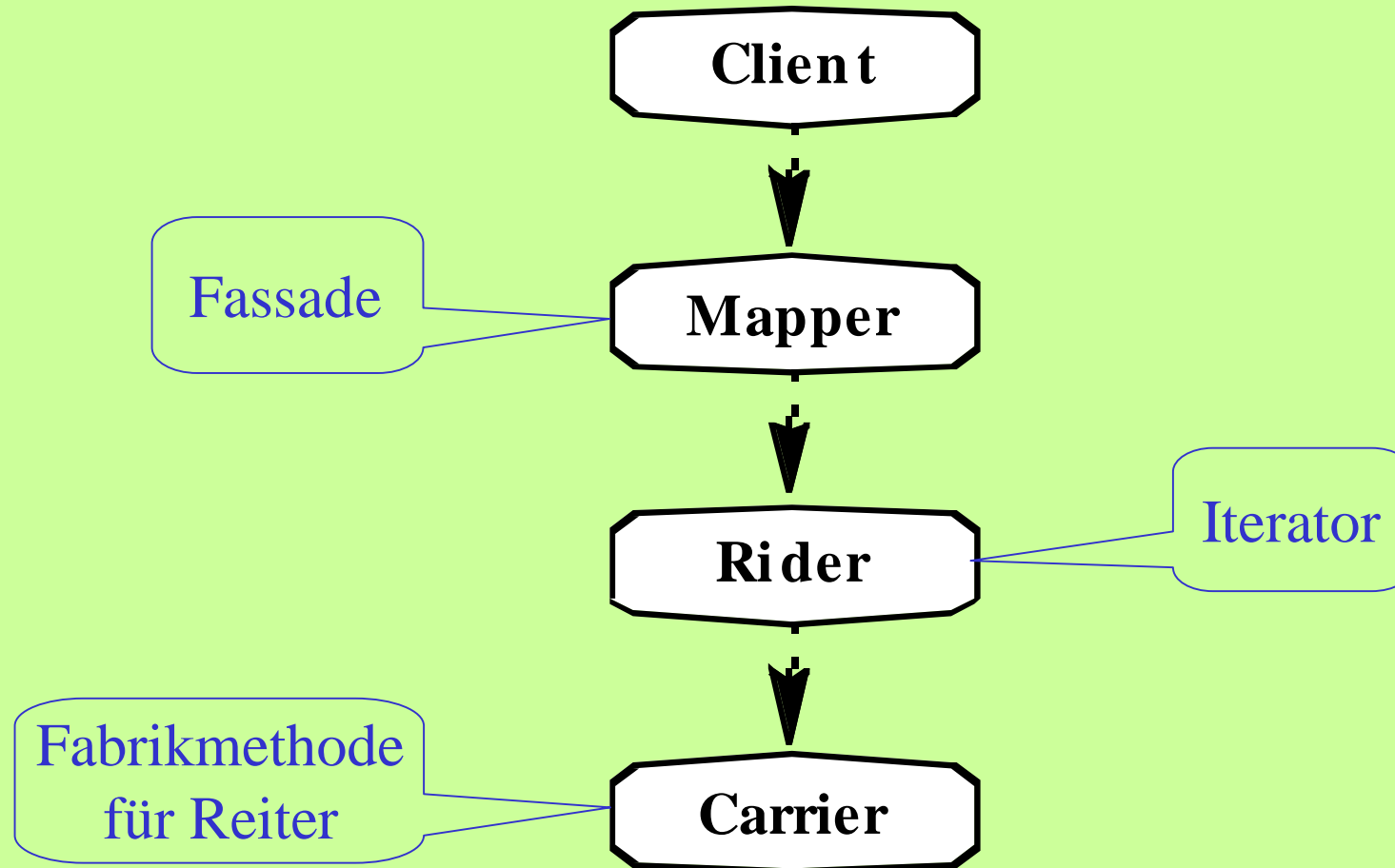
Konzeptklassen

Schnittstellenklassen

Implementationsklassen

Carrier-Rider-Mapper

Allgemein

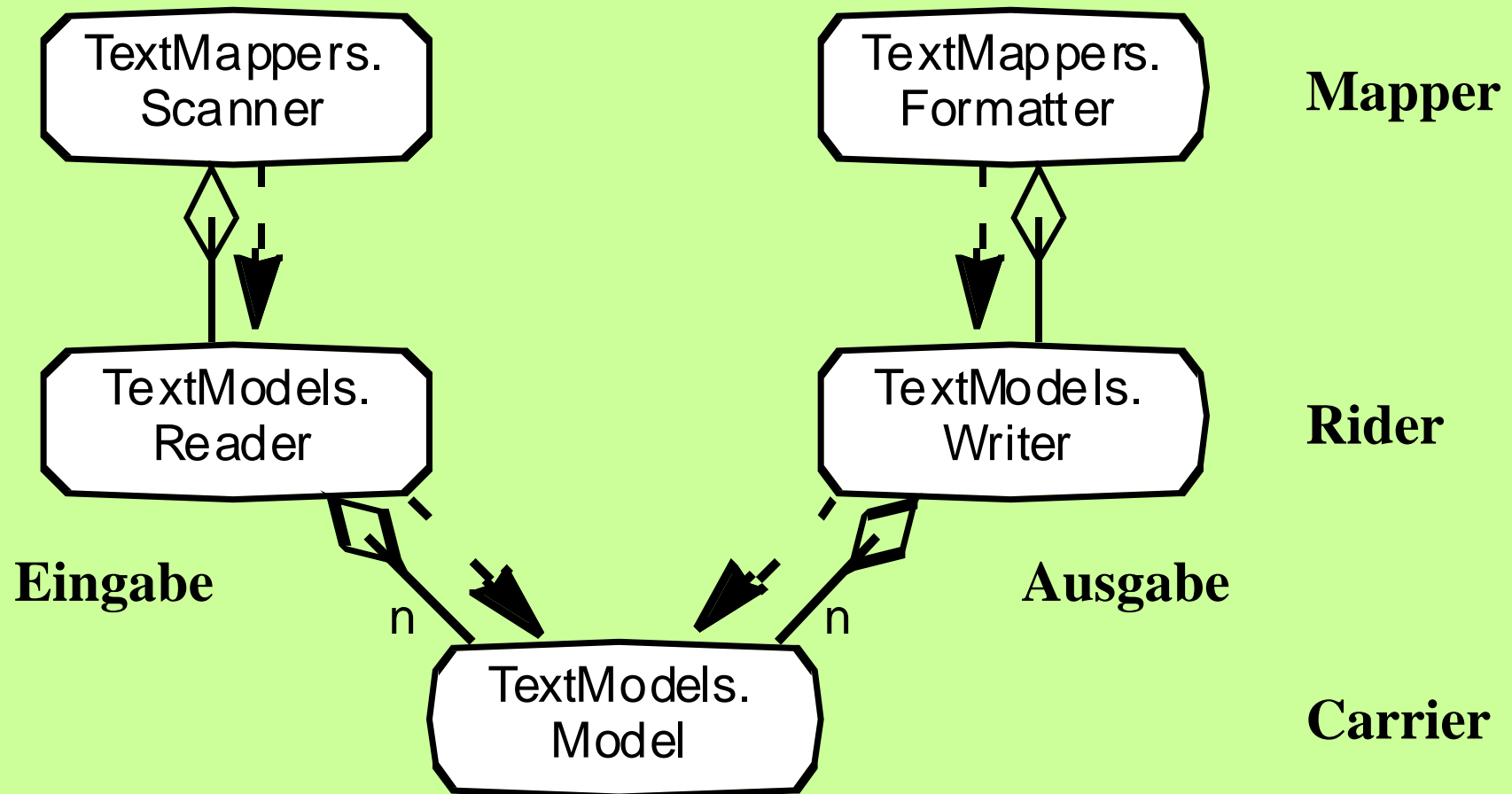


Carrier-Rider-Mapper

- **Träger** (*carrier*)
 - enthält logisch linear geordnete Daten (Bytes)
 - erlaubt rohe Zugriffe mit Positionsangaben
 - Modelle können Träger sein
- **Reiter** (*rider*)
 - hat vom Träger unabhängige aktuelle Position auf Trägerdaten
 - bietet Zugriffe auf Daten von Grundtypen
- **Abbilder** (*mapper*)
 - veredelt rohe Schnittstelle seines Reiters
 - erlaubt Zugriffe auf Daten beliebiger Typen

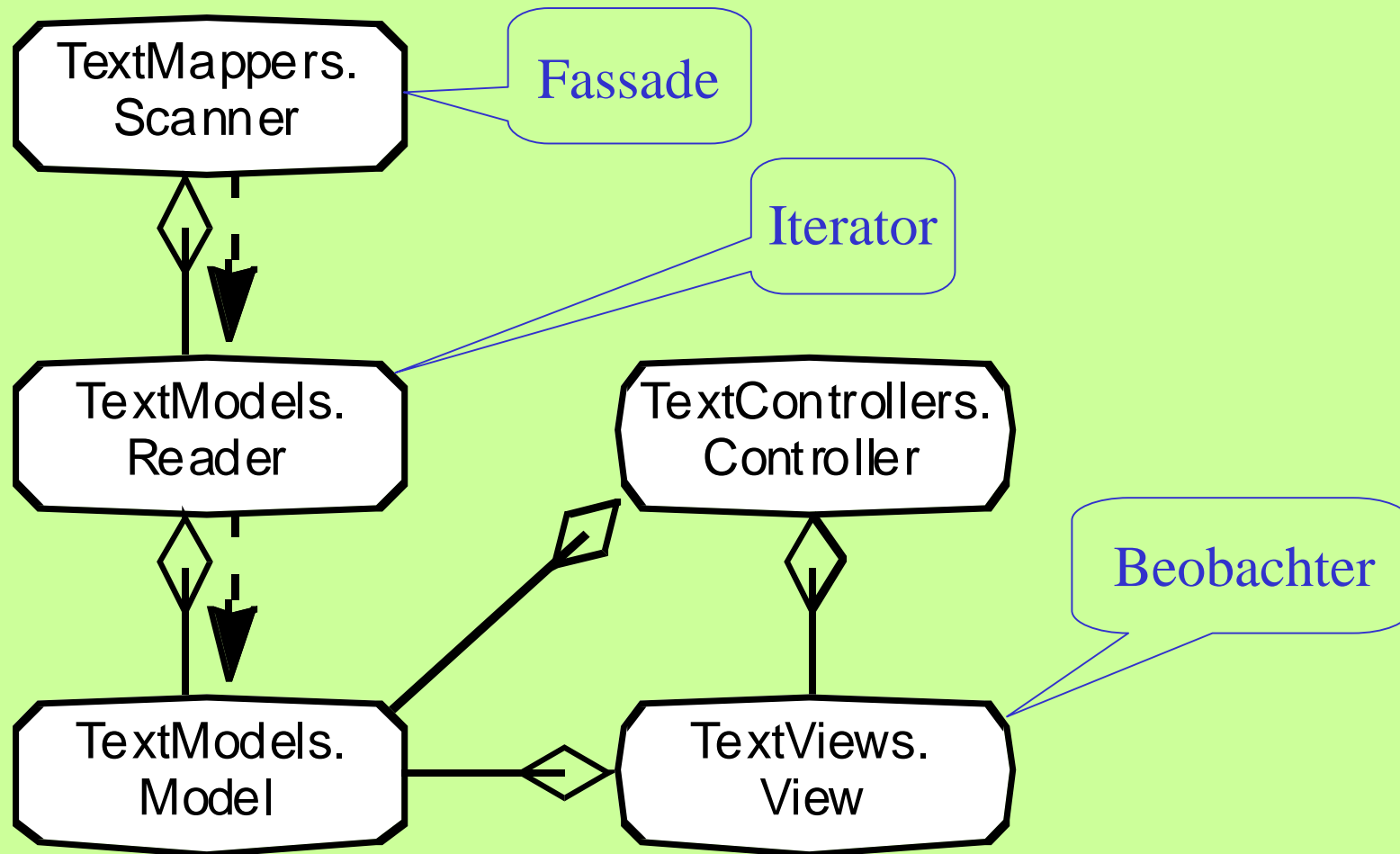
Carrier-Rider-Mapper

Text in BlackBox



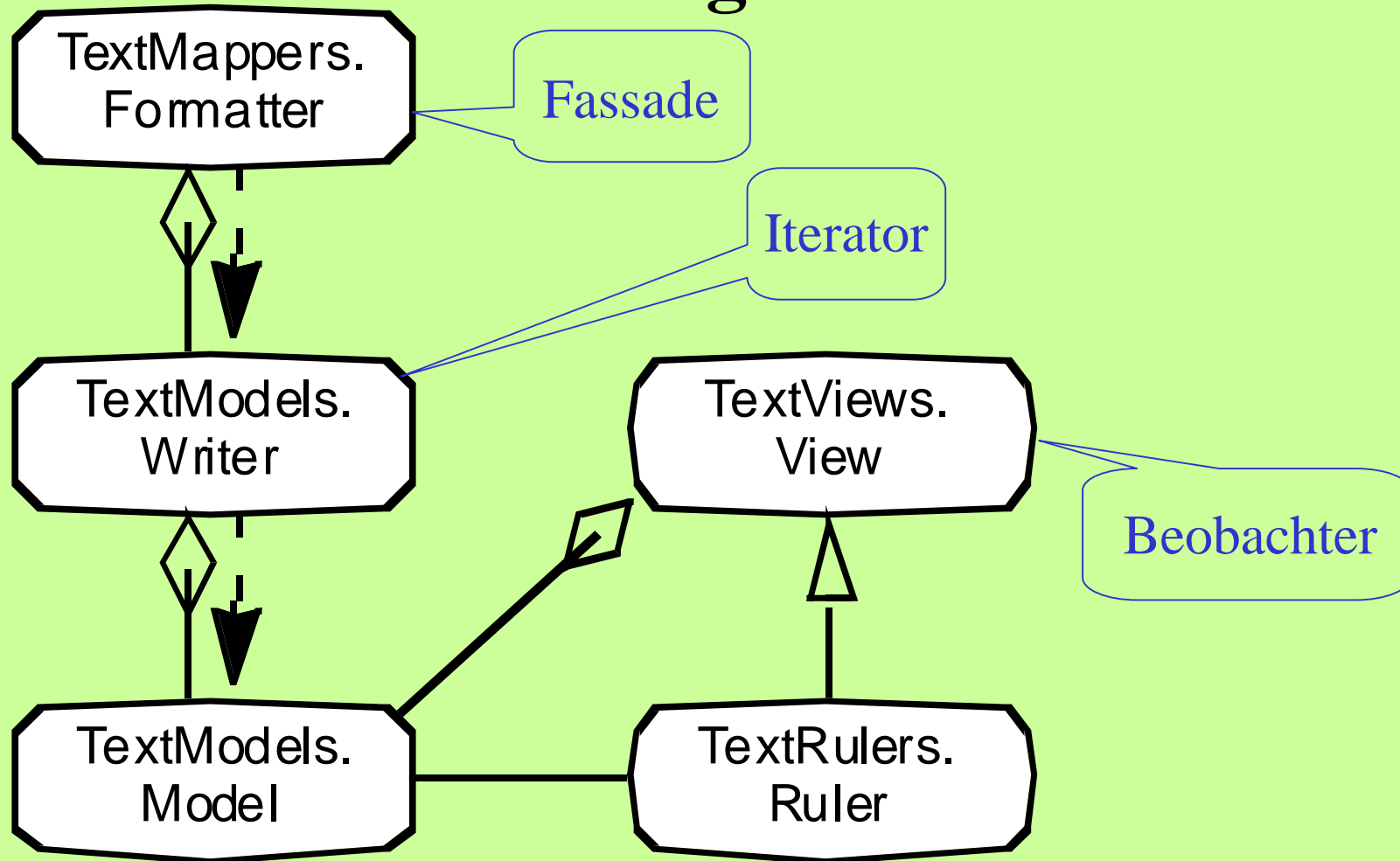
CRM + MVC

Eingabe

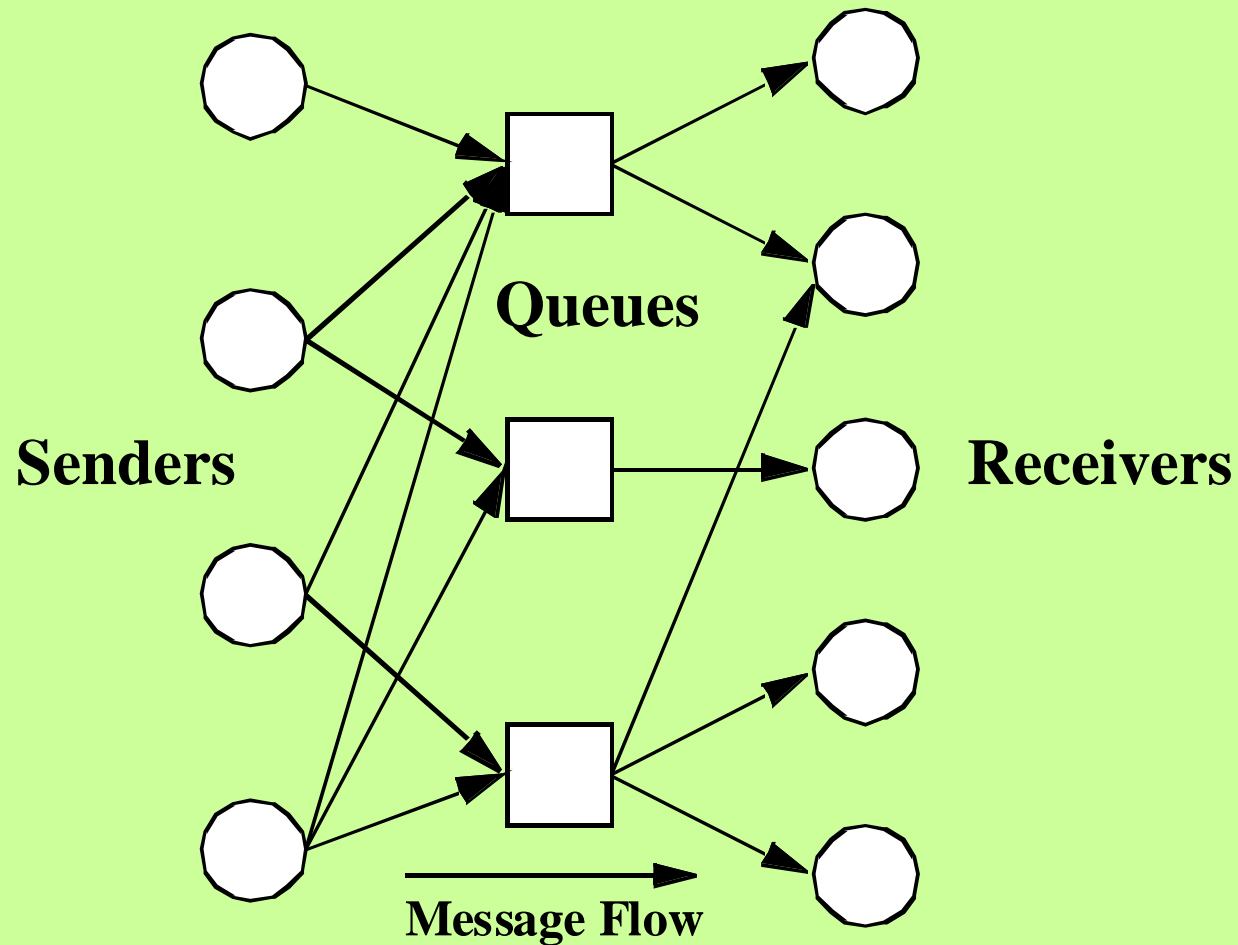


CRM + MVC

Ausgabe

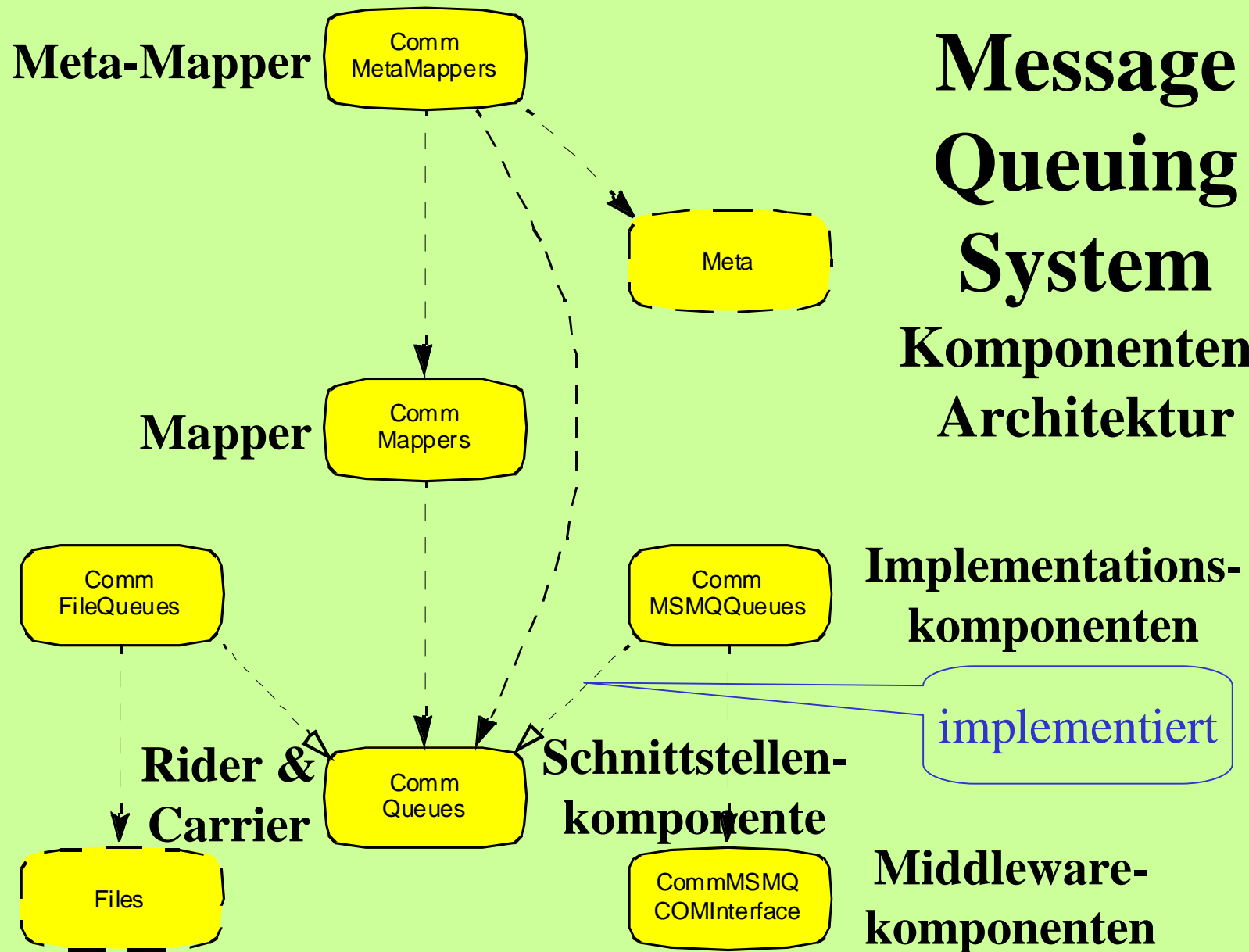


Message Queuing System



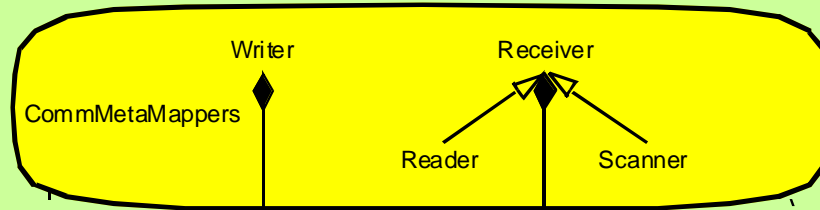
Message Queuing System

Komponenten-Architektur

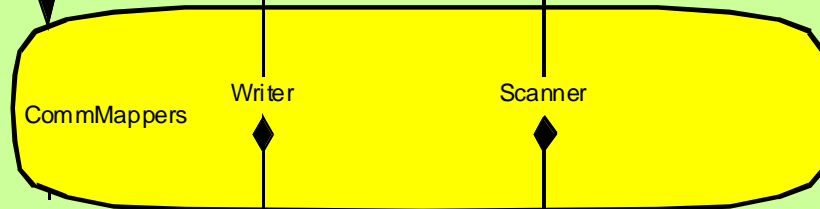


Message Queuing System Klassen-Architektur

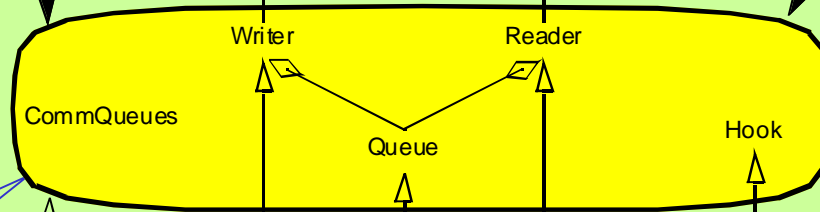
Meta- Mapper



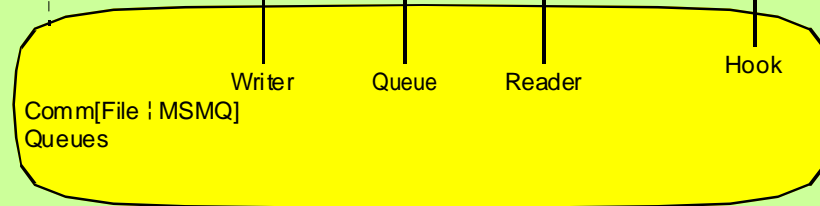
Mapper



Rider & Carrier



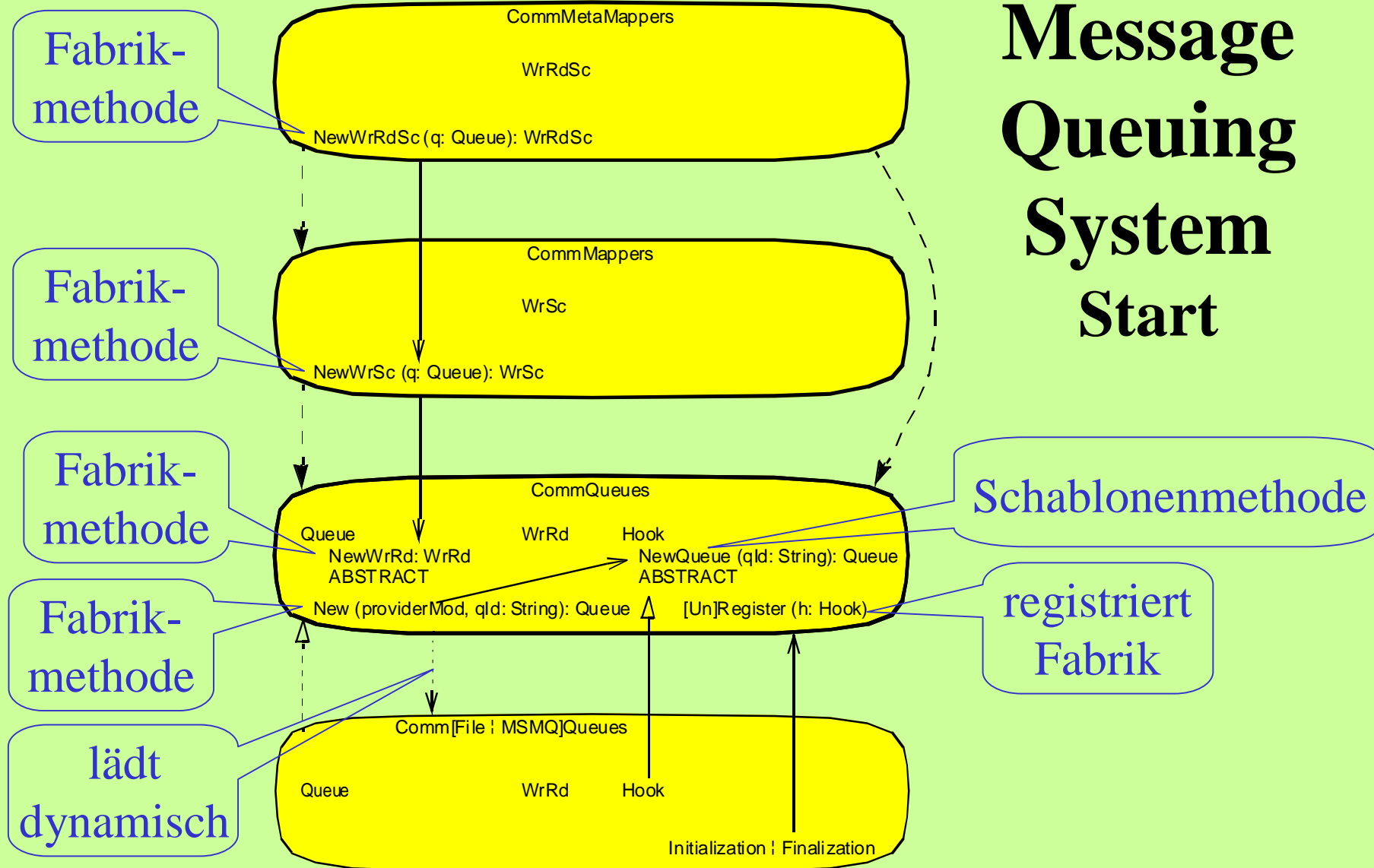
abstrakte Fabrik



Schnittstellen-komponente

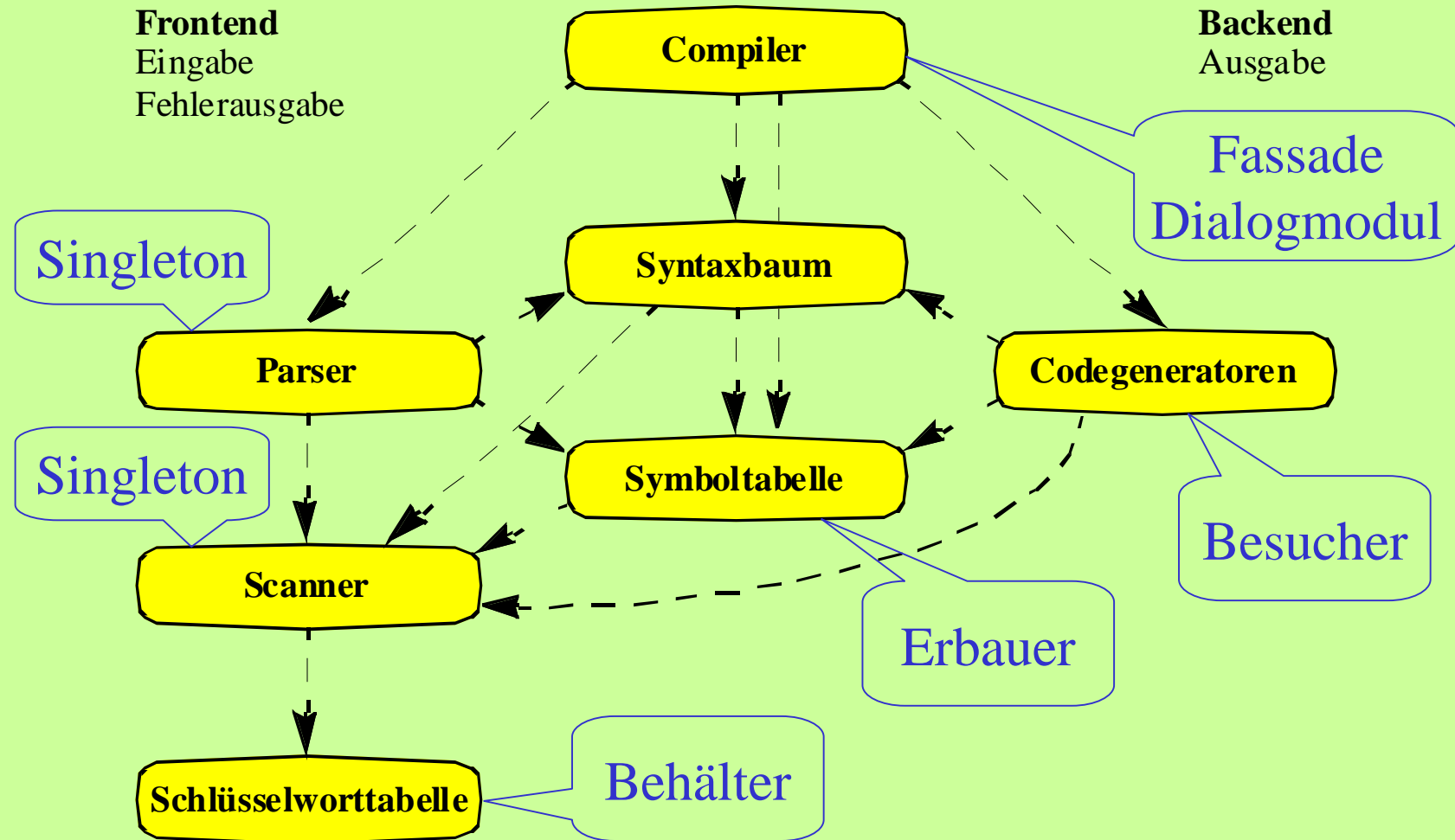
Implementations-komponenten

Message Queuing System Start

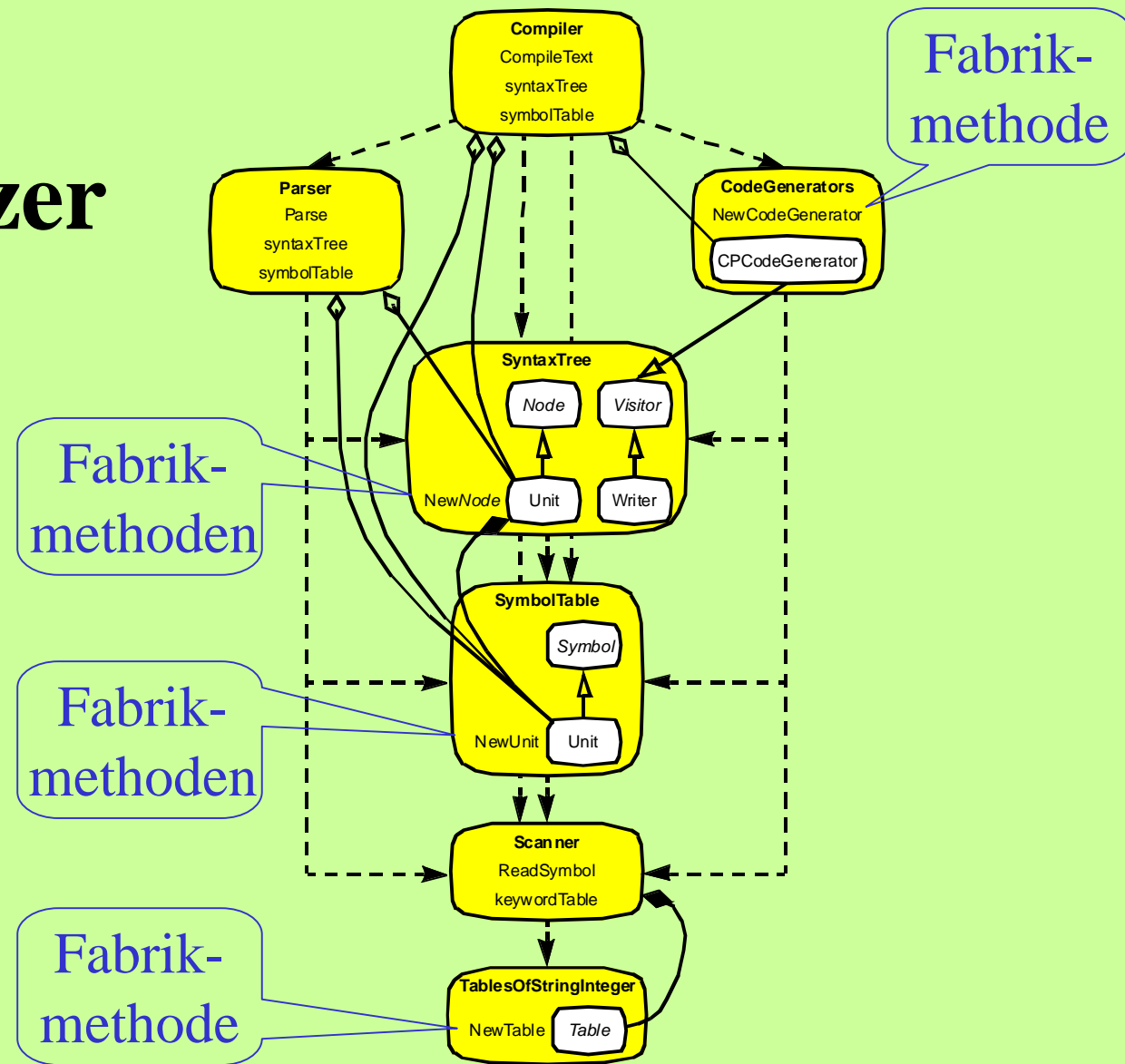


Cleo-Übersetzer

Modul-Architektur

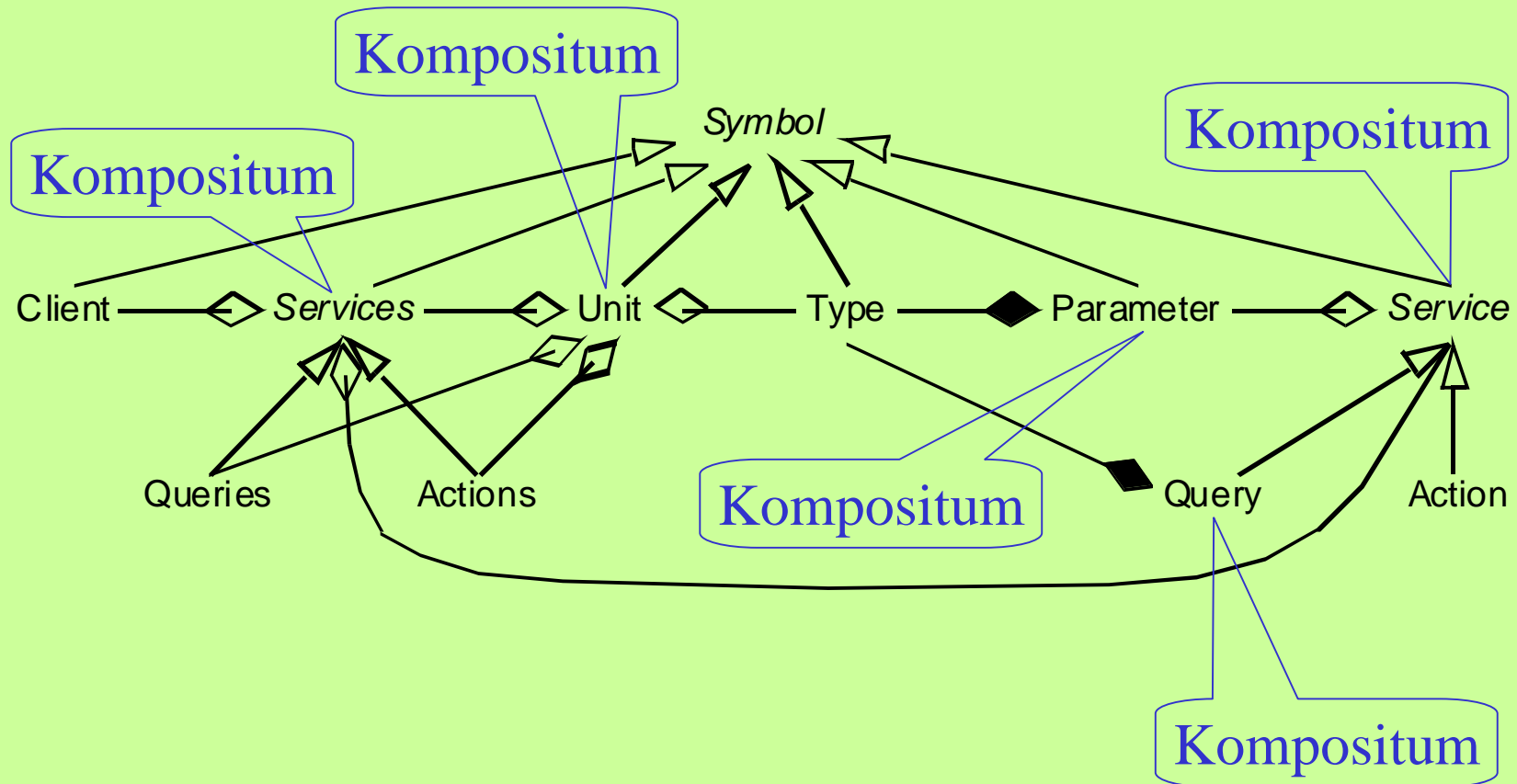


Cleo- Übersetzer Klassen- Architektur

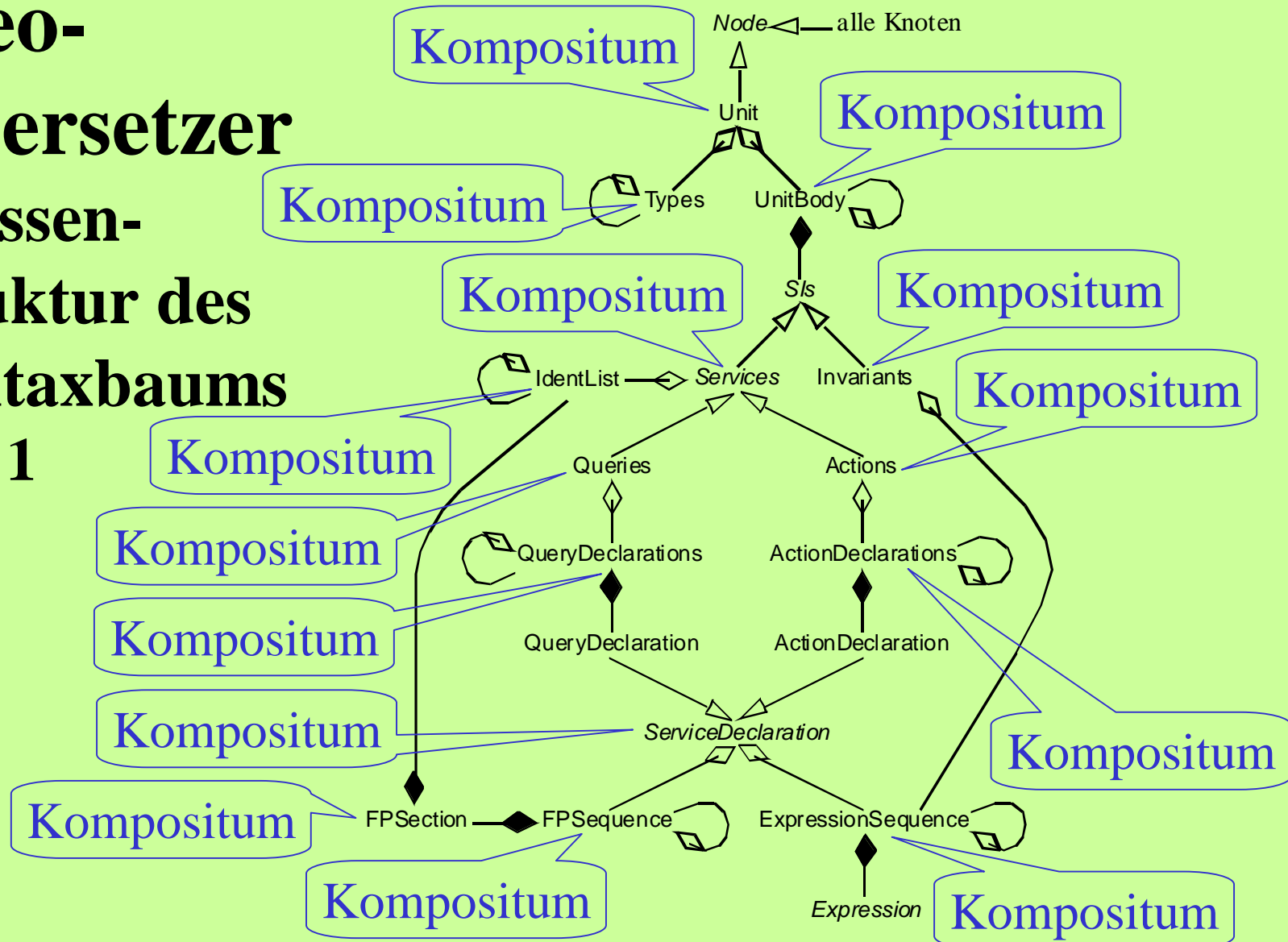


Cleo-Übersetzer

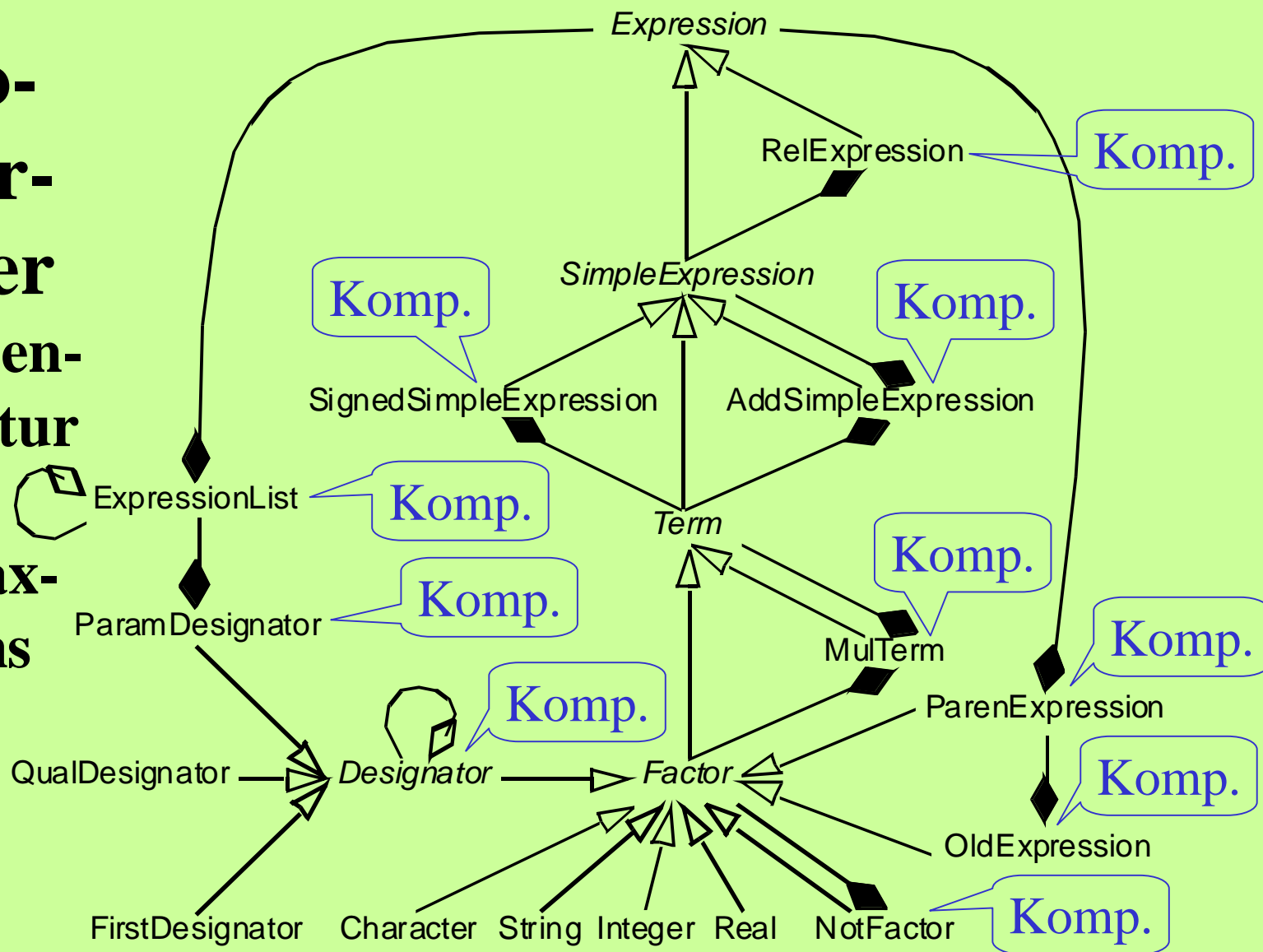
Klassenstruktur der Symboltabelle



Cleo- Übersetzer Klassen- struktur des Syntaxbaums Teil 1



Cleo- Über- setzer Klassen- struktur des Syntax- baums Teil 2



Entwurfsmuster in der Lehre

Weiterer Ansatz

- Warford/Nguyen:
Dynamische Objektstrukturen mit Entwurfsmustern implementieren, z.B. Zustandsmuster

Entwurfsmuster in der Lehre

Fazit

- ☺ Entwurfsmetaphern & -muster sind
 - ☺ sehr nützlich
 - ☺ früh lehrbar
- ☺ Vom kleinen Beispiel zum großen
- ☺ Vom konkreten Entwurf zum abstrakten Muster
- ☺ Vom Modul zur Klasse
- ☺ Vom Klassenensemble zur Komponente
- ☺ Vom Struktur- zum Verhaltensmuster

Entwerfen lernen

Danke für Ihre Aufmerksamkeit!