Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. M. Hölzl, J. Rinke

Übungen zu Softwaretechnik: Programmierung und Software-Entwicklung

Aufgabe 8-1 **Entwurfsmuster - Composite** *Präsenz*

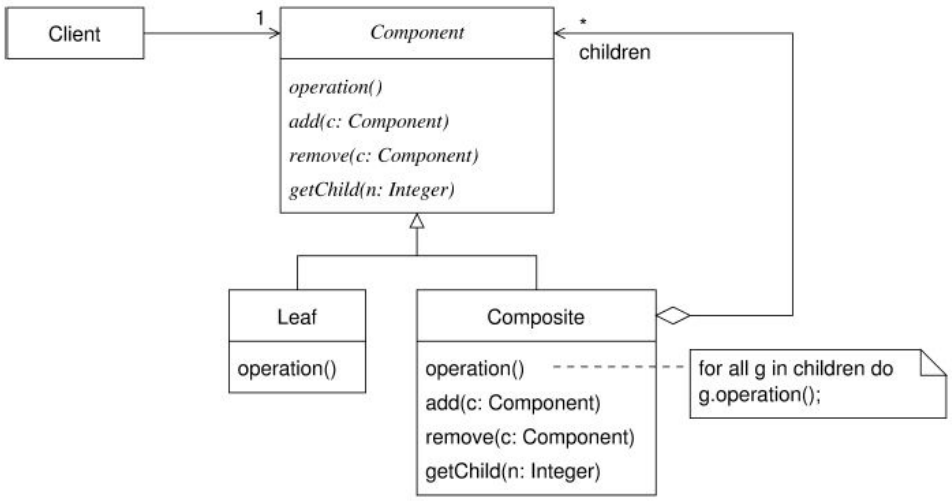

Abbildung 1: Composite Pattern: ein Composite-Item besteht aus beliebig vielen Components, die wiederum entweder Leaf- oder Composite-Items sind

In der vergangenen Woche haben wir uns bereits flüchtig mit dem Composite-Pattern vertraut gemacht. Zur Erinnerung, das Composite-Pattern ist ein Strukturmuster das laut „Gang of Four" folgendes macht:
„Compose objects into tree structures to represent part-whole hierarchies. Composite lets clients treat individual objects and compositions of objects uniformly."
In Abbildung 1 ist die Architektur des Composite-Pattern, so wie Sie es in der Vorlesung kennen gelernt haben, abgebildet.

a) Welche (allgemeinen) Beispielsituationen, die sich unter der Anwendung des Composite-Pattern modellieren lassen, fallen Ihnen ein?
Welche (allgemeinen) Beispiele für Objekte, die Kompositionen aus anderen Objekten der gleichen Objekt-Klasse sind, fallen Ihnen ein?

b) Welche konkreten Anwendungsfälle des Composite-Pattern sind ihnen bereits begegnet? Geben Sie ein Beispiel einer konkreten Implementierung an und überlegen Sie sich, wie dort das Composite-Pattern umgesetzt wurde.

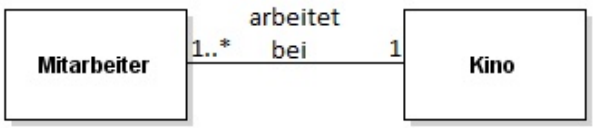

Abbildung 2: Ein Kino hat mindestens einen Mitarbeiter und ein Mitarbeiter arbeitet in genau einem Kino.

c) Auf Übungsblatt05 wurde ein Kino modelliert. Im Zuge dessen wurde die in Abbildung 2 dargestellte Assoziation festgestellt. Bislang wurde also nicht näher spezifiziert welche Rollen ein Mitarbeiter genau haben kann. Es steht lediglich fest, dass ein Kino Mitarbeiter hat. Folgende Rollen kann ein Mitarbeiter haben:
Die Mitarbeiter eines Kinos sind entweder Verkäufer (assistant), Schichtleiter (supervisor) oder Manager. Jeder Verkäufer hat einen direkten Chef, einen Schichtleiter. Ein Schichtleiter wiederum hat einen Chef und das ist ein Manager. Da Mitarbeiter offensichtlich unterschiedliche Rollen einnehmen und untereinander in Relationen stehen, bietet sich eine Implementierung unter Verwendung des Composite-Pattern geradezu an. Modellieren Sie ein entsprechendes Klassendiagramm, das mindestens die Klassen Employee, Assistant, Supervisor, und Manager enthält und implementieren Sie dann die entsprechenden Klassen in Java.

d) Fügen Sie den Klassen Employee, Assistant, Supervisor, und Manager die folgenden Methoden sinnvoll hinzu:

- print()
- getName()
- addEmployee(Employee e)
- removeEmployee(Employee e)
- getSalary()
- getChild(int i)
- getShift()
- setShift(Shift s)

e) Welche Vor- und welche Nachteile die durch die Verwendung des Composite-Pattern entstehen fallen Ihnen auf?

Aufgabe 8-2 **Entwurfsmuster - Reflection** *Präsenz*

Das Reflection-Pattern ist ein Architekturmuster und entstammt nicht der „Gang of Four“, sondern aus dem Buch „Pattern-Oriented Architecture: A System of Patterns“. Es stellt eine Möglichkeit dar Struktur und Verhalten von Softwaresystemen dynamisch zu verändern.

a) Das Refelction-Pattern definiert zwei Ebenen: Meta-Level und Base-Level.
Welche Aufgaben erfüllen die beiden Ebenen jeweils?

b) Ein Merkmal des Reflection-Pattern ist, dass die beiden Ebenen wechselseitig die bereitgestellten Schnittstellen verwenden können. Dennoch sind nur Änderungen auf einer Ebene vorgesehen. Welche Ebene sollte nach der Implementierung nicht mehr verändert werden und bei welcher sollten Änderungen problemlos funktionieren?

c) Neben den beiden genannten Ebenen wird außerdem das Metaobject-Protocol (MOP) definiert. Was könnte das MOP sein und welchen Zweck hat es?

d) Warum ist die zusätzliche Implementierung eines MOP sinnvoll? Macht es Sinn bei der Implementierung des MOP ein bereits bekanntest Designpattern anzuwenden?

e) Jetzt sind alle Komponenten einer Reflection-Architektur bekannt. Skizzieren Sie die Architektur des Reflectionpatterns als UML-Klassendiagramm.