R. Hinze, S. Franck Sommer Semester 2005

Software Engineering

http://proglang.informatik.uni-freiburg.de/teaching/swt/2005/

Übungsblatt 11

Abgabe: 5. Juli 2005

Aufgabe 1 – Design Patterns (I): (4 Punkte)
Auf dem zweiten und dritten Übungsblatt haben Sie mir „Marvel“ gearbeitet. Marvel ist ein Tabellenkalkulationsprogram, das sogenannte „Workspaces“ unterstützt. Diese Workspaces können sowohl beliebig viele Basisdokumente (Tabellen) als auch weitere Workspaces enthalten.
Mit welchem Design Pattern kann man diese Struktur einfach umsetzen, insbesondere wenn man sie erweiterbar halten möchte?
Geben Sie die Interfaces der Klassen an (keine vollständige Implementierung).
Die Objekte müssen nicht ausgearbeitet sein, es reicht aus wenn sie einen Namen haben und grundlegende Verwaltungsmethoden bieten.

Aufgabe 2 – Design Patterns (II): (3 Punkte)
Es sollen auch sog. Sichten implementiert werden, die Projektionen von Tabellen sind. Eine Sicht zeigt nur einen Teil der Daten der Tabelle.
Wie können die Sichten in die existierende Struktur eingebettet werden? Mit welchem Design Pattern kann man die Sichten umsetzen?
Zeichnen Sie ein Klassendiagramm von der Struktur, die sich ergibt.

Aufgabe 3 – Design Patterns (III): (3 Punkte)
Die bisher erstellte Struktur soll sehr viele Funktionen ausführen. Diese werden nach und nach hinzugefügt, und die grundlegenden Klassen sollen nicht mit jeder neuen Methode verändert werden.
Es soll also ein Interface erstellt werden, das nur eine minimale Menge von Operationen ermöglicht und gleichzeitig gut erweiterbar ist, ohne dass an den bestehenden Klassen etwas geändert werden muss. Die Methoden sollen auch nicht im Anwendungsprogramm, das die Klassen benutzt, umgesetzt werden.
Welches Design Pattern bietet eine dritte Alternative?
Passen Sie den Code so an, dass dieses Design Pattern auf die Struktur angewendet werden kann.
Programmieren Sie eine `count` Funktion, die alle Objekte eines Workspaces rekursiv zählt (also auch die Objekte in unter-Workspaces zählt).