R. Hinze, S. Franck Summer Semester 2005

Software Engineering

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

Exercise Sheet 11

Deadline: July 5th, 2005

Exercise 1 – Design Patterns (I): (4 points)
In the second and third exercise sheet you have been working with "Marvel". Marvel is a spread sheet application which offers so-called "Workspaces". These workspaces may contain arbitrary many base documents (spreadsheets) as well as other workspaces.
Which design pattern can be used to implement this structure easily, especially if you want the structure to be extendable?
Give the interfaces of the classes (no full blown implementation).
You don't have to create fancy objects - it is sufficient if they have a name and some basic management methods.

Exercise 2 – Design Patterns (II): (3 points)
We also want to have `Views`, which are projections of `Spreadsheets`. A `View` shows some part of the data of a `Spreadsheet`.
How can the `Views` be added to the existing structure? Which design pattern can be used in implementing the `View`?
Draw a class diagram of the current state of the object's structure.

Exercise 3 – Design Patterns (III): (3 points)
A lot of functionality will be added to this structure. This process will be evolutionary, and we don't want the basic structure to change with every new method.
So we want the interface to come up with a minimal set of operations that allows us to extend the structure without interfering with the base classes. But we also don't want to put all this methods into the client (the classes which use the structure).
Which design pattern implements a third alternative?
What do the interfaces look like after using this design pattern? How would you implement a `count` function, which counts all objects in a workspace recursively (e.g. also counts objects in subordinate workspaces)?