Übungen zur Vorlesung

Softwaretechnologie

- Wintersemester 2012/2013 -

Dr. Günter Kniesel

Übungsblatt 9

Zu bearbeiten bis: 16.12.2012

Bitte fangen Sie **frühzeitig** mit der Bearbeitung an, damit wir Ihnen bei Bedarf helfen können. Checken Sie die Lösungen zu den Aufgaben in Ihr SVN-Repository ein, „Erklärungen" bitte als Textdatei. Fragen zu Übungsaufgaben/Vorlesung können Sie auf der Mailingliste swt-tutoren@lists.iai.uni-bonn.de, bzw. swt-vorlesung@lists.iai.uni-bonn.de stellen.

Aufgabe 1. *Systementwurf* (7 Punkte)

Die Softwarefirma entscheidet sich, eine **Model-View-Controller-Architektur** für die Modellierung der Kasse aus Aufgabe 3 von Blatt 7 zu verwenden. Die Anzeigen (views) sollen sich im Rahmen ihrer Initialisierung beim Modell anmelden, um Aktualisierungs-Benachrichtigungen zu erhalten.

a) Überlegen Sie sich eine sinnvolle Gruppierung der Analyseklassen in Komponenten (Subsysteme) und definieren Sie Dienste. Zeichnen Sie ein passendes **Komponentendiagramm.**

b) Diskutieren Sie, wie die Wahl der **Model-View-Controller-Architektur** folgende Entwurfsziele erfüllt oder verletzt:

- Erweiterbarkeit (z.B. neue „Views")
- Reaktionszeit (Zeit zwischen einer Benutzereingabe und dem Abschluss der Aktualisierung aller Views)
- Änderbarkeit (z.B. die Erweiterung des Modells um zusätzliche Attribute)
- Zugriffs-Kontrolle (d.h. die Sicherstellung, dass nur berechtigte Benutzer auf bestimmte Teile des Modells zugreifen können)

Aufgabe 2. *Entwurfsmuster* (4 Punkte)

In Aufgabe 1 sollten Sie auf eine zyklische Abhängigkeit zwischen der View- und der Model-Komponente gestoßen sein.

a) Mit welchem Entwurfsmuster können Sie – ohne die Kommunikation zu ändern – eine daraus folgende Abhängigkeit auf Klassenebene vermeiden?

b) Wie ist dieses Pattern **im Allgemeinen** aufgebaut? Zeichnen Sie ein entsprechendes Klassendiagramm, das die wichtigsten Klassen und Methoden sowie Assoziationen und Multiplizitäten enthält.

c) Wie lässt sich das Pattern **im konkreten Fall der Situation aus Aufgabe 1** einsetzen? Zeichnen Sie, analog zu Teilaufgabe b, ein Klassendiagram und halten Sie die Bezeichnungen konsistent zu Ihren Lösungen in Aufgabe 1.

Aufgabe 3. *Entwurfsmuster* (8 Punkte)

Für die Verwaltung der gemeinsamen Termine aller Studierenden hat die Fachschaft mit der Entwicklung einer „Appointment-Anwendung“ begonnen. Das Programm soll Appointments (Termine), die aus einer Beschreibung und einem Datum bestehen, verwalten. Eine erste Version dieses Programms finden Sie im SVN-Repository im Ordner „readonly“ als „AppointmentManager.zip“.

Inzwischen haben sich so viele Termine angesammelt, dass auch eine Gruppierung der Termine unterstützt werden soll.

a) Überarbeiten Sie das Programm mit Hilfe eines Entwurfsmusters. Es soll

   a. möglichst wenig an den bestehenden Klassen geändert werden. Vor allem soll die Funktionalität der Klasse *AppointmentItem* erhalten bleiben.

   b. die Möglichkeit bestehen, Gruppen, die wiederum (Unter-)Gruppen oder natürlich auch *Appointment*-Einträge enthalten, einzufügen.

   c. möglich sein, überall Gruppen anzutreffen, wo *Appointment*-Einträge erwartet werden und umgekehrt. Insbesondere muss eine Gruppe dieselben Methoden unterstützen, die auch ein einfacher Eintrag bietet.

b) In der Klasse *AppointmentManagerTest* findet sich ein erstes Programm, das ein paar Beispieleinträge erzeugt und dann auf die Konsole ausgibt. Stellen Sie sicher, dass nach Ihrer Anpassung des Models die Ausgabe noch funktioniert. Kopieren Sie die Klasse und passen Sie die Kopie so an, dass die Einträge in Gruppen geordnet werden. Es sollen zusätzlich zu den einfachen Einträgen auch die Gruppeneinträge und ihre Unterelemente ausgegeben werden, ohne dass der entsprechende *println*-Aufruf oder die *AppointmentManager*-Klasse geändert werden.