Universität Karlsruhe (TH)
Forschungsuniversität · gegründet 1825

Institut für Programmstrukturen und Datenorganisation (IPD)

Prof. Dr. Walter F. Tichy

Vorlesung Softwaretechnik 1

Übungsblatt 4
Sommersemester 2009

Ausgabe: 05.06.2009

Abgabe: 18.06.2009, 11:00 Uhr

UML, Entwurfsmuster, Java Threads

Aufgabe 1: Modellierung mit UML (4 Punkte)

Gegeben sei folgendes Szenario:

> *Ein Internet-Händler bietet Musiktitel sowie die dazugehörigen Musikvideos zum Kauf an. Ein Musiktitel enthält den Titel des Musikstückes sowie Informationen über den Interpreten, das Genre sowie die Dauer des Titels. Der Händler unterscheidet zwischen Musiktiteln ohne Video und Musiktiteln, zu welchen optional das Musikvideo geordert werden kann. Zu einem Musiktitel kann es maximal ein Musikvideo geben. Die gewünschten Musiktitel können auf verschiedenen Medien bestellt werden. Zur Auswahl stehen Medien für Musiktitel ohne Video sowie Musiktitel mit Video. Musiktitel ohne Video können auf Vinyl, CD, DVD, BluRay bestellt oder als Download heruntergeladen werden. Musiktitel mit Video können nicht auf Vinyl bestellt werden. Die Medien für Musiktitel mit Video werden mit einem Wasserzeichen versehen. Dieses kann sowohl ein wahrnehmbares Wasserzeichen als auch ein nicht wahrnehmbares Wasserzeichen sein. Auch die einzelnen Musiktitel und Videos sind mit Wasserzeichen versehen. Im Gegensatz zu Videos, welche sowohl mit wahrnehmbaren, als auch mit nicht wahrnehmbaren Wasserziechen versehen werden können, werden Musiktitel nur mit nicht wahrnehmbaren Wasserzeichen versehen. Die Wasserzeichen dienen dazu, den Urheber, den Verkäufer sowie den Käufer identifizieren zu können.*

Finden Sie in obigem Szenario alle beteiligten Klassen sowie Assoziationen zwischen den Klassen und zeichnen Sie ein Klassendiagramm. Berücksichtigen Sie dabei Multiplizitäten, Rollen und Leserichtungen.

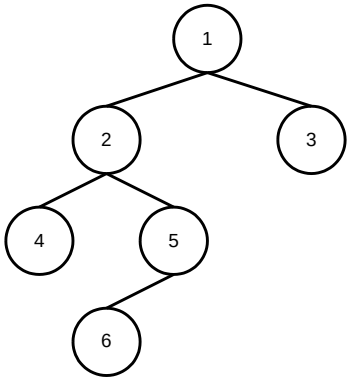
Hinweis: Nähere Informationen zu wahrnehmbaren und nicht wahrnehmbaren Wasserzeichen finden Sie auf Wikipedia: http://de.wikipedia.org/wiki/Digitales_Wasserzeichen

Aufgabe 2: Entwurfsmuster (Iterator, Besucher) (5 Punkte)

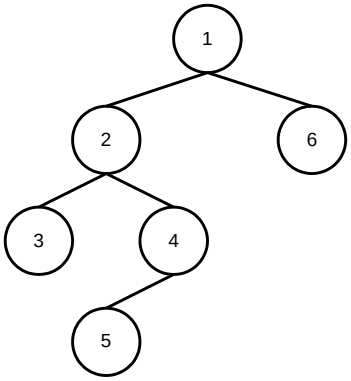
Gegeben sei folgendes Szenario:

Sie möchten eine Anwendung implementieren, welche ihre Daten in einer Datenstruktur in der Form eines Baumes speichert. Für das Verarbeiten des Baumes möchten Sie verschiedene Varianten implementieren, wie der Baum mit einem `Iterator` traversiert werden soll. Für das aktuelle Einsatzgebiet der Anwendung genügen Ihnen zwei verschiedene Traversierungsarten (`Traversal`):

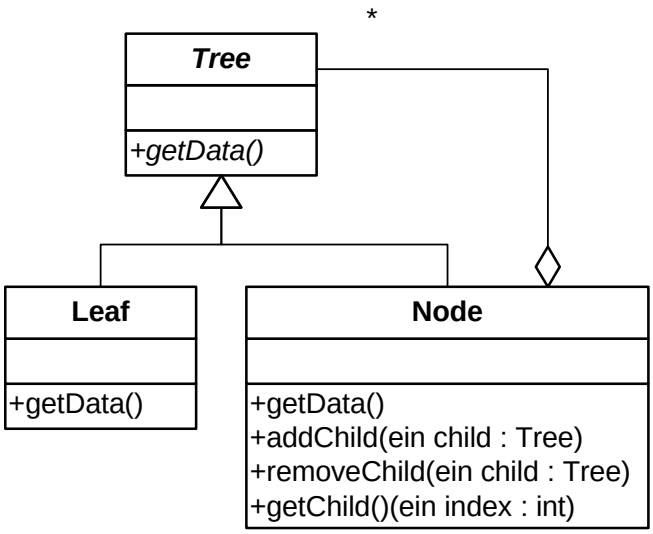
Ebenenweise (`BreadthTraversal`*):* Beginne bei der Wurzel und durchlaufe dann alle Ebenen nacheinander von links nach rechts (vgl. Breitensuche).



Hauptreihenfolge (`DepthTraversal`): Durchlaufe die Wurzel und anschließend zuerst den linken, dann den rechten Teilbaum (vgl. Tiefensuche).



Die verwendete Baum-Datenstruktur ist wie folgt aufgebaut:



a) Verwenden Sie die Entwurfsmuster *Iterator* und *Besucher* und erstellen Sie für das gegebene Szenario ein UML-Klassendiagramm, welches die Klassen `Iterator`, `Traversal`, `BreadthTraversal` und `DepthTraversal` enthält.
Erweitern Sie die Klassendefinitionen um die notwendigen Methoden und Attribute. Machen Sie geeignete Annahmen, sofern dies erforderlich ist. (4 Punkte)

b) Begründen Sie kurz die Nützlichkeit des Entwurfsmusters *Besucher* in dem gegebenen Szenario. (1 Punkt)

Aufgabe 3: Java Threads (4 Punkte)

a) Gegeben sei folgender Java-Quelltext aus der Vorlesung, welcher von 2 Ausführungsfäden gleichzeitig ausgeführt wird:

```
if (globalvar > 0) {
  globalvar--;
}
```

Wie auch in der Vorlesung ist die Variable `globalVar` eine globale Variable vom Typ `int`, welche den Wert 1 enthält und auf welche von beiden Fäden zugegriffen werden kann. Zusätzlich sei diese Variable nun mit `volatile` definiert:

```
volatile int globalVar = 1;
```

Kann mit dieser Änderung auch eine Wettlaufsituation auftreten? Begründen Sie Ihre Entscheidung in wenigen Sätzen. (2 Punkte)

b) Gegeben seien eine sequentielle und eine parallele Implementierung eines Algorithmus, welcher die Summe aller Elemente in einem Feld bildet und auf dem Teile-und-Herrsche Prinzip basiert.

```
public int sum(int[] array) {
  if (array.length <= 1) {
    return array;
  }

  int r1 = sum(getLeftSubArray(array));
  int r2 = sum(getRightSubArray(array));

  int result = r1 + r2;

  return result;
}
```

Angenommen, in der parallelen Implementierung des Algorithmus wird für jeden Aufruf von `sum(…)` einer neuer Faden erzeugt und gestartet. Wie viele Fäden werden gestartet, wenn die parallele Implementierung der Methode `sum(…)` mit einem Feld der Größe `n` aufgerufen wird? Wieso ist die sequentielle Implementierung i.d.R. schneller (bezogen auf die Laufzeit des Algorithmus) als die parallele Implementierung mit Java Threads? (2 Punkte)

Aufgabe 4: Merge-Sort (12 Punkte)

Für diese Aufgabe steht Ihnen eine Klasse `IOHelper` zur Verfügung, welche Methoden zur Ein- und Ausgabe von Daten zur Verfügung stellt. Sie finden diese Klasse unter:

https://svn.ipd.uni-karlsruhe.de/lehre/vorlesung/SWT1/SS09/studies

Verwenden Sie die zur Verfügung gestellte Klasse, um Eingaben entgegenzunehmen (`getNextIntegerArray()`) und Ihre Ergebnisse auszugeben (`printResult(int[] array)`). Bei jeder Ausführung der Anwendung müssen Sie ein Integer-Feld aufsteigend, d.h. aus ‚5 6 3' wird ‚3 5 6', sortieren und das sortierte Feld ausgeben. Verwenden Sie das Entwurfsmuster *Schablonenmethode* zur Lösung der Aufgaben.

a) Implementieren Sie eine sequentielle Variante des Merge-Sort-Algorithmus zum Sortieren von Integer-Feldern (`int[]`).
b) Parallelisieren Sie die sequentielle Variante des Merge-Sort-Algorithmus aus Teilaufgabe a). Verwenden Sie hierzu Java Threads.

Bei der Ausführung der Anwendung soll standardmäßig die parallele Variante des Merge-Sort Algorithmus ausgeführt werden! Die Ein- und Ausgabe soll über die mitgelieferte Klasse stattfinden.

Hinweis: **Bitte lesen Sie sich den folgenden Text aufmerksam durch. Missachtung der Anweisungen wird Sie Punkte kosten!**

Die Lösung geben Sie bitte nicht nur als E-Mail sondern auch über den Praktomaten ab. Die Anmeldung am Praktomaten erfolgt unter folgender Adresse:

https://swt1.ipd.uka.de/

Die Oberfläche zum Einreichen der Lösungen finden Sie unter

https://swt1.ipd.uka.de/praktomat/

Den Quelltext Ihres Programmes drucken Sie bitte aus und geben ihn zusammen mit dem Übungsblatt ab. Zusätzlich müssen Sie sowohl den Quelltext als auch die Java-Anwendung an folgende E-Post-Adresse senden:

swt1@ipd.uka.de

Als Betreff geben Sie bitte **[Uebungsblatt 4] <TUTNR> <MATRNR> <VORNAME> <NACHNAME>** an.

Hängen Sie den Java-Quelltext als ZIP-Archiv an die E-Mail an. Der Name des ZIP-Archivs muss folgendes Format haben: **<TUTNR>_<MATRNR>_<VORNAME>_<NACHNAME>.zip**. Das ZIP-Archiv darf nur den Java-Quelltext (d.h. Dateien mit der Endung .java) und evtl. benötigte Ressourcen (wie bspw. Icons) enthalten. Sofern Sie mit Paketen gearbeitet haben, muss diese Ordnerstruktur ebenfalls in Ihrem ZIP-Archiv vorhanden sein.

Verwenden Sie in Ihrem Java-Quelltext nur ASCII-Zeichen und vermeiden Sie IDE-spezifische Java-Annotationen. Die lauffähige Java-Anwendung hängen Sie als **ausführbares JAR-Archiv** zusätzlich zum ZIP-Archiv an die E-Mail an. Es werden nur Java-Anwendungen akzeptiert, welche als ausführbares JAR-Archiv an die E-Mail angehängt sind! Das JAR-Archiv muss als separater Anhang gesendet werden. Der Name des JAR-Archivs muss wie folgt lauten: **<TUTNR>_<MATRNR>_<VORNAME>_<NACHNAME>.jar**

Beispiel: Sie heißen Hans Müller, gehen in das Tutorium 7 und Ihre Matrikelnummer ist 123456. Ihr Betreff sieht dann wie folgt aus:

[Uebungsblatt 4] 07 123456 Hans Mueller

Die Anhänge heißen dann:

07_123456_Hans_Mueller.zip / 07_123456_Hans_Mueller.jar

Die E-Mail mit beiden Anhängen muss bis zum offiziellen Abgabetermin bei uns eingegangen sein. Vergewissern Sie sich, dass Sie alle Code-Dateien, die zur Übersetzung der Java-Anwendung notwendig sind, mitschicken und der Quelltext übersetzt werden kann. Prüfen Sie, dass die Java-Anwendung aus dem JAR-Archiv lauffähig ist. Andere Dateiformate werden nicht akzeptiert.

Bei Missachtung der obigen Anforderungen oder Teilen davon wird die gesamte Aufgabe mit 0 Punkte bewertet.