Praktikum WP Certified Tester SoSe 14

Aufgabenblatt 2

Prof. Dr. Bettina Buth <buth@informatik.haw-hamburg.de>
Raum 7.86b, Tel. 040/42875-8150

Bearbeitungshinweise:

- Die Bearbeitung der Aufgaben findet in **festen Vierergruppen** statt.
- Ein Teil der bearbeiteten Lösungen wird während der Praktikumsstunde abgenommen. Dazu werden Sie abwechselnd Ihre Lösungen vorstellen.
- Ein anderer Teil erfordert eine schriftliche Abgabe, diese geschieht über EMIL – der Abgabetermin wird im Aufgabenblatt explizit genannt.
- Es gibt **100% Anwesenheitspflicht** beim Praktikum. Beim Fehlen wegen z.B. Krankheit müssen Atteste eingereicht werden und ein Nachholtermin wird vereinbart.

Ziel des Praktikums:

- Grundtechniken des Blackbox-Testens an Beispielen anwenden, speziell: Äquivalenzklassen, Grenzwerte, Zustandsbasierte Testfälle
- Code-Coverage Analyse bei Tests kennenlernen und am Beispiel lernen, Test-Coverage zu erhöhen durch geeignete Testfäll

Vorbereitung vor dem Praktikum

Das Anwendungsbeispiel für diese Aufgabe stammt aus einem Softwareengineering-Kurs der North Carolina State University. Es handelt sich um eine Variante des *CoffeeMaker.*

Im pub und in Emil liegen die zip-Archive *CoffeeMaker_JUnit* und *CoffeeMaker_Coverage*.

Achtung: die Quellen zu CoffeeMaker_Coverage unterscheiden sich von denen für CoffeeMaker_Junit aus Aufgabenblatt 1 – daher muss das Projekt neu aufgesetzt werden

Im Rahmen der Aufgaben muss zunächst CoffeeMaker-Coverage als Eclipse-Projekt mit JUnit aufgesetzt werden:

- Das zip-Archiv *CoffeeMaker-Coverage* ins lokale Verzeichnis kopieren und entpacken.
- In Eclipse entweder das Projekt CoffeeMaker_Coverage importieren oder ein neues Projekt CoffeeMaker_Coverage anlegen und die Quellen von CoffeeMaker_Coverage dorthin importieren (funktioniert beides).
- Sicherstellen, dass JUnit3 im Buildpath liegt - ansonsten als external lib angeben.
- Neben der Bibliothek für JUnit muss zusätzlich das PlugIn EclEmma eingebunden werden.

Danach sollte das Projekt fehlerfrei aufgesetzt sein. Das kann dadurch geprüft werden, dass die vorhandenen JUnit-Tests ausgeführt werden; die JUnit Tests befinden sich im Verzeichnis unittests.

Zusatzinformationen:

- Zum CoffeeMaker als Anwendung: http://realsearchgroup.org/SEMaterials/tutorials/coffee_maker/
- Tutorials zur Coverage mit EclEmma: http://realsearchgroup.org/SEMaterials/tutorials/eclemma/
- EclEmma allgemein: http://www.eclemma.org/ (dort auch Info für die Einbindung in Eclipse)
- EclEmma alternativer Installationsweg: Source von http://sourceforge.net/projects/eclemma/ herunterladen. Inhalte (aus den Sourcen) des feature- und plugin-Ordners in den entsprechenden Ordner in Eclipse hineinkopieren. Mit einem Neustart von Eclipse ist EclEmma dann verfügbar.

Abgabetermin für die schriftlichen Aufgabenteile: Do, 1.5.2014

Black-Box Tests

Aufgabe 1.1: Äquivalenzklassen und Grenzwerte

Abgabeform: schriftliche Abgabe

Betrachten Sie die Methode int F (int x, int y) mit den Vorbedingungen

- V1: x soll kleiner oder gleich 3 sein
- V2: y soll echt zwischen -2 und 2 liegen

Die Spezifikation für F besagt, dass die Methode den Wert x*x liefern soll, wenn y <= 0 ist; x*y falls y >0 und x > 0 ist und 17 falls y >0 und x <= 0.

a) Bestimmen Sie die Äquivalenzklassen dieser Methode als abstrakte Testfälle in folgender Tabelle an und unterscheiden Sie zwischen gültigen und ungültigen Äquivalenzklassen

Id Äquivalenzklasse	**Definition der ÄqKlasse**
Äq1	
…	
Aq n	
UÄq 1	
…	
UÄq m	

b) Geben Sie für jeden abstrakten Testfall einen Repräsentanten als konkreten Testfall an sowie den erwarteten Sollwert

Geben Sie die Ergebnisse in einer Tabelle der folgenden Form an:

Äquivalenzklasse (Id)								
Repräsentant								
Sollwert								

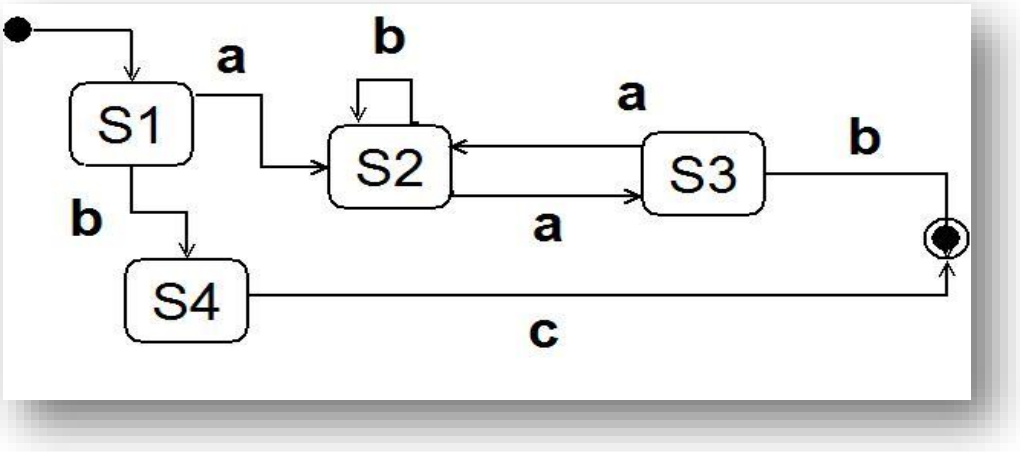
c) Geben Sie zusätzlich Testfälle an, die eine 100%ige Überdeckung der Grenzwerte der Äquivalenzklassen ergeben. Verwenden Sie hierfür folgende Tabelle:

Äquivalenzklasse (Id)								
Grenzwert								
Sollwert								

Aufgabe 1.2: Zustandsbasierte Testfälle

Abgabeform: schriftliche Abgabe

Gegeben sei der folgende Zustandsautomat



a) Geben Sie für diesen Automaten den Übergangsbaum für den Zustands-Konformanztests an
b) Geben Sie für diesen Automaten den Übergangsbaum für den Zustands-Robustheitstests an
c) Geben Sie auf der Basis der vorigen Teilaufgaben eine möglichst kleine Anzahl von Testfällen als Folge von Ereignissen und Zuständen an, die eine vollständige Zustandsüberdeckung zu erreichen
d) Geben Sie auf der Basis der vorigen Teilaufgaben einen möglichst kleine Anzahl von Testfällen an, die zusätzlich benötigt werden um eine 100%ige Zustandsübergangs-Überdeckung zu erreichen

JUnit Tests und Code Coverage

Nach: Open Seminar Software Engineering der North Carolina State University

Aufgabe 2.1: Analyse der Ausgangssituation

Abgabeform: Abnahme im Praktikum

Der Source Code des CoffeeMaker_Coverage ist vollständig, d.h. lauffähig; die mitgelieferten Tests sind in JUnit 3 geschrieben.

Analysieren Sie die ursprünglichen Testfälle im Projekt CoffeeMaker_Coverage bezüglich der Überdeckung des Source Codes der einzelnen Klassen. Erstellen Sie einen Coverage Report mit EclEmma und erläutern Sie diesen.

Aufgabe 2.2: Test-Coverage erhöhen

Abgabeform: Abnahme im Praktikum, schriftliche Abgabe des Coverage-Reports und Test-Dokumentation

Hinweis: während der Entwicklung und Durchführung der Tests darf der Code nicht verändert werden!

Ergänzen Sie die JUnit-Tests so, dass die Tests für die Klassen CoffeeMaker, Inventory und Repository erfolgreich sind (grüner Balken) und die Coverage bei EclEmma mindestens 90% beträgt für die Coverage-Maße

- instructions,
- branches,
- lines,
- methods

Dokumentieren Sie die Testfälle wie für Aufgabenblatt 1 im Test Case Document; außerdem erstellen Sie bitte vor und nach der Ergänzung der Tests die Coverage-Reports mit EclEmma zum Vergleich und erläutern diese.

Aufgabe 2.3: Test-Coverage analysieren

Abgabeform: Abnahme im Praktikum, schriftliche Abgabe der Begründung

Untersuchen Sie ob für die obigen Coverage-Maße auch eine Überdeckung von 100% möglich ist. Begründen Sie Ihre Aussage schriftlich.

Hinweis:

- Melden Sie sich für die Abnahme der Aufgabe 2.1 und 2.2 sobald die jeweilige Aufgabe fertig bearbeitet ist und bevor Sie mit den folgenden Aufgaben beginnen
- Aufgabe 1.1 und 1.2 könne unabhängig von den übrigen Aufgaben und außerhalb des Praktikumstermins bearbeitet werden.

Viel Spaß!