Hinweise zur Abgabe des Testplans für Software–Projekt 2

Prof. Dr. Rainer Koschke

7. Mai 2014

Abgabe

Das Format der elektronischen Fassung des Testplans muss PDF sein. Es muss zu jedem Abschnitt die/der AutorIn bzw. die Autoren genannt werden. Die Schnittstellentests müssen als JUnit-Tests im Quellcode abgegeben werden, zusammen mit den Schnittstellen (Interfaces), die sie testen. Diese Dateien sollten zusammen in eine .zip-Datei gepackt werden. Aus dem Quellcode muss der Autor/die Autorin jedes Tests klar hervorgehen (JavaDoc). Wenn externe Bibliotheken benutzt werden, sollen diese auch mit abgegeben werden. Des Weiteren ist eine formlose Anleitung, wie die Testklassen zu kompilieren sind, abzugeben. Die Testklassen müssen kompilieren!

Inhalt des Testplans

Inhalt und Aufbau des Testplans entsprechen der vereinfachten Form des *IEEE Standard for Software Test Documentation 829-1998*, wie im Rahmen der Vorlesung eingeführt.

1. Einführung: Wozu dient dieses Dokument und an wen richtet es sich? Beziehung zu anderen Dokumenten: Verweise auf die Anforderungsspezfikation, den Testplan oder andere relevante Dokumente.
2. Systemüberblick: Überblick über das System hinsichtlich jener Komponenten, die durch Komponententests geprüft werden sollen. Granularität der Komponenten (einzelne Klassen, ganze Pakete oder Systeme etc.) und ihre Abhängigkeiten. Hier kann auf die Architekturspezifikation Bezug genommen werden.
3. Merkmale, die getestet/nicht getestet werden müssen: Konzentration auf funktionale Gesichtspunkte beim Testen: identifiziert alle Merkmale und Kombinationen von Merkmalen, die getestet werden müssen. Beschreibt auch diejenigen Merkmale, die nicht getestet werden und gebt Gründe dafür an.

4. Abnahme- bzw. Testendekriterien: Spezifiziert objektiv überprüfbare Abnahmekriterien für die Tests bzw. wann das Testen beendet wird (z.B. Testabdeckung, Verhältnis Fehler/Lines-of-Code, Budget).
5. Vorgehensweise: Allgemeine Vorgehensweisen beim Testablauf (z.B. Black-Box- versus White-Box-Tests); Integrationsstrategien; Arten der Leistungstests.
6. Aufhebung und Wiederaufnahme: Kriterien, wann Testaktivitäten ausgesetzt werden (z.B. wenn Fehlerrate einen Schwellwert übersteigt). Testaktivitäten, die wiederholt werden müssen, wenn das Testen wieder aufgenommen wird.
7. Zu prüfendes Material (Hardware-/Softwareanforderungen): Notwendige Betriebsmittel: physikalische Eigenarten der Umgebung sowie Anforderungen an Software, Hardware, Testwerkzeuge und andere Betriebsmittel (z.B. Arbeitsraum).
8. Testfälle: Auflistung aller Testfälle anhand individueller Testfallspezifikationen. Schwerpunkt sollen hier Integrations- und Leistungstests sein. Ihr könnt auf die genaue Spezifikation der Komponententests verzichten - diese werden durch die JUnit-Testfälle gegeben. Uns genügt hierzu eine Beschreibung, welche Komponenten/Klassen wie, wann und durch wen getestet werden sollen. Die zugehörigen JUnit-Tests werden, wie oben beschrieben, separat abgegeben und müssen im Testplan nicht als Code aufgeführt werden. Wir sind im Testplan mehr daran interessiert, welche Arten von Integrationstests und Leistungstests Ihr vorseht und wie Ihr diese genau ausgestalten wollt.
9. Testzeitplan: entfällt in SWP-1, muss aber in SWP-2 ausgefüllt werden.

Die JUnit-Tests werden später im Rahmen Eurer Testaktivitäten noch um die White-Box-Testfälle für den Komponententest ergänzt. Eine LaTeX-Vorlage mit weiteren Hinweisen findet Ihr im Stud.IP.

Mindestanforderungen und Aufgabenteilung

Bei der Zuordnung von Verantwortlichkeiten ist Folgendes zu beachten. Jedes Gruppenmitglied testet mindestens zwei Klassen seiner Wahl im Komponententest mit den folgenden Randbedingungen:

- TesterIn ist nicht zugleich AutorIn der Klassen (d.h. es werden Klassen anderer Personen getestet);
- die ausgewählten Klassen implementieren wichtige und nicht-triviale Funktionalität Eures Systems (also keine einfachen Datenklassen, die nur Setter und Getter haben);
- eine Klasse wird durch Black-Box-Tests getestet (Abgabe jetzt);
- die andere Klasse wird durch White-Box-Tests getestet (Abgabe mit Implementierung bei Endabgabe);

- das Rahmenwerk *JUnit* bzw. *Android JUnit Extension* wird für den Test verwendet;
- jede nicht-triviale Methode der Klasse soll geeignet getestet werden. Eine triviale Methode ist eine, die ein Attribut nur setzt oder liest (Setter oder Getter).

Dies stellt die Mindestanforderung dar. Ihr müsst alle wesentlichen Komponenten testen, um eine sehr gute Note in SWP-2 zu erhalten. Eine Komponente ist wesentlich, wenn ihr Versagen die Benutzbarkeit des Systems stark einschränken würde (z.B. Benutzer kann sich nicht einloggen und dadurch keine Funktionen ausführen) oder gar hohe Schäden anrichten könnte (z.B. fehlerhafte Funktion löscht den ganzen Datenbestand oder unautorisierter Benutzer kann beliebig Daten manipulieren). Bei einer Bibliothekssoftware wäre das zum Beispiel eine Komponente, die die Ausleihe regelt. Ein Beispiel für eine weniger kritische Funktion wäre die Versendung von Informationen über zurückgegebene Bücher, die ein Leser vorgemerkt hat.

Die Qualität der Implementierung wird am Ende mit bewertet und dazu gehört der Aspekt Zuverlässigkeit. Insofern ist dringend anzuraten, über das geforderte Mindestmaß hinaus zu gehen. Die Testfälle in Kapitel 8 sind mit Abstand der wichtigste Teil hinsichtlich der Bewertung, da dieser Teil etwa 50 % der Note ausmacht. Bearbeitet auch die Merkmale in Kapitel 3 sorgfältiger.

Werkzeuge

Für die Komponententests sollt Ihr Junit/Android Testing Framework benutzen. Eine Sammlung von Werkzeugen zur Bestimmung der Testabdeckung findet Ihr hier.