Testkonzept

Inhaltsverzeichnis

1. Einleitung

Um die Zuverlässigkeit und die Qualität der Software und des gesamten Systems zu verbessern, sind Tests durchzuführen. Die Testreihe lässt sich in drei Stufen einteilen, nämlich Komponententest, Integrationstest und Systemtest.
JUnit bietet als Java-Framework die Möglichkeit des Testens von Anwendungsklassen. Es erlaubt das Schreiben und Ausführen von automatisierten Unit-Tests.

2. Testsablauf

2.1 Komponententest

Komponententest erfolgen einzeln für jede implementierte Klasse und jedes Package. Diese Test werden von den Programmierern der entsprechenden Story mit Hilfe von JUnit durchgeführt. Dazu wird zu jeder Klasse eine Testklasse erzeugt. Diese beinhaltet die Testmethoden. Der Name der Testmethoden setzt sich aus “test“ und dem Namen der zu testenden Methode zusammen.

2.2 Integrationstest

Nach dem alle vorhandenen Komponenten einzeln erfolgreich getestet wurden, erfolgt der Integrationstest. Hier wird das Zusammenwirken und die Kommunikation der Klassen und Packages untereinander betrachtet. Ziel ist es Schnittstellenfehler aufzudecken, die in den Komponententests nicht feststellbar sind. Nach jedem Integrationsschritt ist zutesten, ob das Zusammenspiel der Komponenten korrekt ausgeführt wird. Damit erhöht sich die Wahrscheinlichkeit, dass nur Fehler in den zuletzt integrierten Packages und Klassen auftreten. Auch hier werden automatisierte Test mittels JUnit durchgeführt.

2.3 Systemtest

Der Systemtest überprüft einerseits die funktionalen Qualitätsmerkmale der Korrektheit und Vollständigkeit des kompletten Systems. Andererseits steht jedoch auch die Benutzbarkeit, die Dokumentation und Stabilität auf dem Prüfstand. Automatisierte Test werden überall dort durchgeführt, wo das System die entsprechenden Schnittstellen bietet. Alle anderen Testfälle werden manuell durchgeführt.

3. Methoden

3.1 White-Box-Test

Der Begriff White-Box-Test (auch Glass-Box-Test) bezeichnet eine Methode des Software-Tests, bei der die Tests mit Kenntnissen über die innere Funktionsweise des zu testenden Systems entwickelt werden. Im Gegensatz zum Black-Box-Test ist für diesen Test also ein Blick in den Quellcode gestattet, d.h. es wird am Code geprüft.
Ziel des Tests ist es, sicherzustellen, dass Testfälle im Bezug auf die Überdeckung des Quellcodes gewisse Hinlänglichkeitskriterien erfüllen. Gängig sind dabei u.a. folgende Maße:
Anweisungsüberdeckung: Ausführung aller Anweisungen
Kantenüberdeckung: Durchlaufen aller möglichen Kanten von Verzweigungen des Kontrollflusses
Bedingungsüberdeckung (mehrere Varianten): Bewertung der Bedingungen
Pfadüberdeckung (mehrere Varianten): Betrachtung der Pfade durch ein Modul

3.2 Black-Box-Test

Black-Box-Test bezeichnet eine Methode des Software-Tests, bei der die Tests ohne Kenntnisse über die innere Funktionsweise des zu testenden Systems entwickelt werden. Er beschränkt sich auf funktionsorientiertes Testen, d. h. für die Ermittlung der Testfälle wird nur die Spezifikation (gewünschte Wirkung), aber nicht die Implementierung des Testobjekts herangezogen. Die genaue Beschaffenheit des Programms wird nicht betrachtet, sondern vielmehr als Black Box behandelt. Nur nach Außen sichtbares Verhalten fließt in den Test ein.
Ziel ist es die Übereinstimmung eines Softwaresystems mit seiner Spezifikation zu überprüfen. Ausgehend von formalen oder informalen Spezifikationen werden Testfälle erarbeitet, die sicherstellen, dass der geforderte Funktionsumfang eingehalten wird. Das zu testende System wird dabei als Ganzes betrachtet, nur sein Außenverhalten wird bei der Bewertung der Testergebnisse herangezogen.