ConceptPeople Test-Toolbox

Automatisierte Tests bilden das Herzstück eines guten Software-Entwicklungsprozesses. Sie ermöglichen den Einsatz von Continuous Integration Produkten, die dem Entwicklungsteam permanent Feedback über die Qualität des Systems liefern. Dies erhöht die Sicherheit bei der Software-Entwicklung, da Auswirkungen von Änderungen oder komplexen Refactorings schnell sichtbar werden. Dies steigert die Qualität der Software und somit auch die Kundenzufriedenheit. Continuous Integration Umgebungen sind mittlerweile den Kinderschuhen entwachsen und lassen sich auch ohne viel Erfahrung schnell einrichten. Eine Test-Umgebung ohne Tests liefert keine Fehler - allerdings auch keine Informationen. Der Grund für fehlende Tests ist vielfältig. Oft werden Zeitgründe vorgeschoben oder Komponenten als untestbar eingestuft. Was ist also wichtig, damit die Testabdeckung durch automatisierte Tests steigt?

Tests müssen schnell zu schreiben sein

Die Hemmschwelle automatisierte Tests zu schreiben steigt, wenn dazu ein hoher Overhead anfällt. Das Herstellen der Testvoraussetzungen muss schnell möglich sein, damit der Fokus auf den eigentlichen Testfall gelegt werden kann. Das Überprüfen der Sollergebnisse muss dementsprechend einfach möglich sein. Ein weiterer wichtiger Aspekt ist die Analysemöglichkeit während der Testerstellung. Fehler beim Schreiben des Tests müssen schnell erkannt und gelöst werden, damit der Test fertig gestellt wird und nicht aus Zeitgründen abgebrochen wird.

Tests müssen übersichtlich sein

Eine Gefahr bei unübersichtlichen Tests besteht darin, dass diese gelöscht werden, wenn sie aufgrund von Software-Änderungen fehlschlagen. Da der Entwickler den Ablauf des Tests nicht nachvollziehen kann, wird der Test nicht angepasst sondern auskommentiert oder entfernt. In der Regel sollte die Anzahl der Tests jedoch steigen und nicht abnehmen.

Die ConceptPeople Test-Toolbox erweitert bestehende Open-Source-Produkte, um die schnelle Erstellung von übersichtlichen Unit-Tests zu vereinfachen. Die Toolbox-Frameworks sind aus der Praxis für die Praxis entstanden und haben sich bereits in mehreren Projekten unterschiedlichster Art bewährt.

Gern stellen wir Ihnen die ConceptPeople Test-Toolbox detailliert in einer Präsentation vor!

Integrationstest – Datenbank

Kaum eine Anwendung in der Java-Welt kommt heute ohne eine Datenbankanbindung aus. In der Regel wird dazu eine eigene Datenbankzugriffsschicht definiert, die alle Interaktionen mit der Datenbank kapselt. Es spielt dabei keine Rolle, ob O/R-Mapper wie Hibernate involviert sind oder reines JDBC verwendet wird. Das Grundprinzip ist in allen Fällen das gleiche: Daten werden aus der Datenbank gelesen oder in der Datenbank manipuliert. Das Testen dieser Funktionalität stellt sich in der Praxis häufig schwierig dar, da das Bereitstellen von initialen Testdaten und das Prüfen der Ergebnisse den Großteil der Arbeit einnehmen. Dies bedeutet, dass die meisten Zeilen des programmierten Unit-Tests darauf verwendet werden, die Rahmenbedingungen für den Test herzustellen, während der fachliche Test in wenigen Zeilen abgehandelt ist. Dies führt unweigerlich dazu, dass die Testmethode unübersichtlich wird. Aufgrund dieses Overheads ist die Gefahr groß, dass dieser Test gar nicht geschrieben wird oder später nicht mehr angepasst wird.

Um diese Probleme zu umgehen, bietet die ConceptPeople Test-Toolbox ein Framework, was auf dem Open-Source-Produkt DBUnit (www.dbunit.org) aufbaut. Das Framework kann einfach in beliebige Tests aufgenommen werden, ohne dass dazu Klassenhierarchien angepasst werden müssen. Weiterhin wird das einfache Einspielen von Testdaten in Form von XML-Dateien unterstützt. Das Framework ermittelt pro Test-

ConceptPeople Test-Toolbox

methode die einzuspielenden Testdaten (Convention over Configuration), sodass in der Testmethode sofort mit dem fachlichen Test begonnen werden kann. Für die Prüfung der Ergebnisse bietet das Framework weitere Methoden, sodass diese mit zwei Zeilen Programmcode erledigt werden kann. Vollständige Testszenarien sind häufig nicht länger als 10 bis 20 Zeilen.

Anhand der aktuellen Datenbank erzeugt das Framework eine DTD, die zur Beschreibung der Testdaten verwendet werden kann. Die erzeugte DTD ermöglicht wiederum in den gängigen Software-Entwicklungsumgebungen eine Auto-Vervollständigung bei der Erstellung der Testdaten. Dies beschleunigt das Erstellen der Testdaten massiv.

Liefert eine neu entwickelte Testmethode nicht das erwartete Ergebnis, muss zunächst überprüft werden, ob der Fehler in der Anwendung oder im Test selbst liegt. Die Test-Toolbox unterstützt den Entwickler bei dieser Analyse, damit auch dieser Schritt schnell erledigt werden kann.

Integrationstest – Web GUI

Das automatisierte Testen von Web-Frontends wird heutzutage bereits durch Open-Source-Produkte gut unterstützt. Die ConceptPeople Test-Toolbox setzt auf dem Produkt Selenium (seleniumhq.org) auf, um die Testerstellung weiter zu vereinfachen. Die Integration des Frameworks ist mit wenigen Zeilen in jede bestehende Testklasse möglich. Änderungen an der Klassenhierarchie sind selbstverständlich nicht erforderlich. Die Stärke des Frameworks liegt in der Verwendung eines eigenen Modells zur Abbildung der zu testenden Web-Anwendung. Das Aufzeichnen von Aktionen auf der Web-Oberfläche wird bewusst nicht eingesetzt, da die Pflege und Wartung dieser aufgezeichneten Code-Fragmente in der Praxis kaum zu bewerkstelligen ist. Die Verwendung des Modellansatzes erhöht im Gegensatz dazu die Lesbarkeit der Unit-Tests und minimiert die erforderlichen Anpassungen bei Änderungen der Web-Anwendung.

Natürlich lassen sich die ConceptPeople Test-Toolbox Frameworks kombinieren, was im Falle von DBUnit und Selenium einen erheblichen Produktivitätsschub mit sich bringt: Durch die Kombination der Frameworks lassen sich GUI-Tests in beliebiger Granularität entwickeln. Es können einzelne Aspekte (Validierung, Vorbelegung, Dialogverhalten), einzelne Use-Cases oder ganze User-Acceptance-Tests durchgeführt werden. Die Praxis zeigt, dass während der Entwicklung kürzere Unit-Tests zur Sicherstellung der Funktionalität entwickelt werden, während in nachgelagerten Testphasen große Testblöcke verwendet werden.