Whitepaper zu Unit-Tests

Autor: Stefan Roock (stefan.roock@it-agile.de)

Unit-Testen: Stand der Kunst

Die agilen Methoden haben insbesondere bei der Frage automatisierter Tests ein drastisches Umdenken in der Industrie bewirkt. Mindestens automatisierte Unit-Tests - auch Komponenten- oder Modul-Tests genannt- (i.d.R. mit xUnit-Tools wie z.B. JUnit) gehören heute zum Standard in immer mehr Softwareprojekten.

Zerbrechliche Unit-Tests

Die Unit-Tests werden von den Entwicklern erstellt (z.T. im Test-First-Vorgehen) und haben in der Regel mind. White-Box-Anteile. Die zwangsläufige Konsequenz: Die Unit-Tests zerbrechen schnell, wenn die getesteten Klassen geändert werden (z.B. im Rahmen von Refactorings). Viele zerbrochene Unittests würden also Refactorings behindern statt sie zu unterstützen.

.

Woran erkennt man gute Unit-Tests?

Auf dieser Basis lässt sich die zentrale Forderung an Unittests ableiten:
Wird Logik in einer Produktivklasse geändert, zerbricht genau ein Unittest. Diese Forderung ist idealisiert und in der Praxis größerer Projekte nicht zu erreichen.
Daher wird sie etwas aufgeweicht:
Wird Logik in einer Produktivklasse geändert, zerbrechen wenige Unittests.
Damit verschiebt sich die Bewertung der Qualität von Unit-Tests deutlich gegenüber dem, was man heute in vielen Projekten vorfindet: Unittests werden häufig nach dem Motto „viel hilft viel“ geschrieben. Dieses Vorgehen erhöht nicht nur die Aufwände für das Schreiben der Tests - auch Refactorings werden erheblich behindert.

akquinet it-agile GmbH
Paul-Stritter-Weg 5
22297 Hamburg
Fon +49 (0)40 88 173-300
Fax +49 (0)40 88 173-333

www.it-agile.de
Mail: info@it-agile.de

Handelsregister Hamburg HRB 92261

Wie testet man Unit-Tests?

Dabei kann man die Forderung nach passgenau fehlschlagenden Unit-Tests überraschend leicht prüfen (man braucht quasi den Akzeptanztest für die Unit-Tests). Man kann diese Prüfung exemplarisch oder automatisiert durchführen:

- Bei der exemplarischen Prüfung ändert man in einigen ausgewählten Klassen jeweils ein Stück Logik (z.B. eine if-Bedingung umdrehen) und lässt die Unittests danach ablaufen.
- Die automatisierte Prüfung läuft nach demselben Schema ab, Passende Open-Source-Werkzeuge stehen z.B. für Java (Jester), Python (Pester) und C# (Nester) zur Verfügung.

Passgenau fehlschlagende Unit-Tests

Natürlich möchte man nicht erst hinterher feststellen, ob die Unit-Tests passgenau fehlschlagen. Der Schlüssel für passgenau fehlschlagende Unit-Tests liegt in der Modularisierung des Produktivcodes. Massenhaft fehlschlagende Unit-Tests als Folge einer Logikänderung sind häufig nicht (nur) durch die Unittests selbst verschuldet. Sehr oft ist das Problem ein deutliches Indiz für Strukturschwächen im Produktivcode. Es fehlt ihm an Kapselung und Isolierung der Einzelteile.

Integrations- und Akzeptanztests

Die Forderung nach passgenau fehlschlagenden Unit-Tests macht es unmöglich, dass die Unit-Tests Funktionalität auf höherer Ebene oder die Integration vieler Klassen testen. Daher müssen alle Tests, die oberhalb der beschriebenen Unit-Tests ansetzen, deutlich von den Unit-Tests unterschieden werden.

Diese höherwertigen Integrations- und Akzeptanztests können mit den xUnit-Werkzeugen erstellt werden. Sie werden dann jedoch schnell unübersichtlich. Eine Alternative sind spezialisierte Werkzeuge. Hier hat in der letzten Zeit das Open-Source-Framework FIT eine relevante Verbreitung erlangt. FIT ist für viele gängige Programmiersprachen verfügbar (z.B. Java. .NET, Python, Perl, C++, Ruby, Smalltalk). Mit FIT beschäftigt sich ein eigenes Whitepaper (siehe unter Referenzen).

akquinet it-agile GmbH
Paul-Stritter-Weg 5
22297 Hamburg
Fon +49 (0)40 88 173-300
Fax +49 (0)40 88 173-333

www.it-agile.de
Mail: info@it-agile.de

Handelsregister Hamburg HRB 92261

Stabile Integrations- und Akzeptanztests

Die Integrations- und Akzeptanztests sollen im Gegensatz zu den Unittests Black-Box-Tests sein und stabil gegenüber Logikänderungen in einzelnen Klassen. Diese Forderung ist sehr einfach herzustellen, wenn man Integations- und Akzeptanztests strikt von den Unit-Tests trennt. Damit steht auch mit den Integrations- und Akzeptanztests auch ein zusätzliches Sicherheitsnetz für Refactorings zur Verfügung. Einige sehr radikale Umstrukturierungen des Systems können trotz aller Sorgfalt eine große Menge von Unit-Tests auf einen Schlag zerbrechen. In diesem Fall bleiben die Integrations- und Akzeptanztests, um die korrekte Durchführung der Refactorings zu prüfen. Dabei kann es mitunter auch sinnvoll sein, ganze Unit-Tests wegzuwerfen – wenn nämlich die Anpassung des Unit-Tests aufwändiger ist als das Neuschreiben.

Referenzen

- FIT: http://fit.c2.com
- FIT-Whitepaper: http://www.it-agile.de/fileadmin/docs/Akzeptanztests-FIT.pdf
- Jester, Pester, Nester:http://jester.sourceforge.net
- JUnit: http://www.junit.org
- xUnit: http://www.xunit.org

akquinet it-agile GmbH
Paul-Stritter-Weg 5
22297 Hamburg
Fon +49 (0)40 88 173-300
Fax +49 (0)40 88 173-333

www.it-agile.de
Mail: info@it-agile.de

Handelsregister Hamburg HRB 92261