Erstellung und Einsatz von Test-Videos für die Verständigung zwischen Kunden, Anforderungsingenieur, Tester und Entwickler

Jianwei Shi und Kurt Schneider
Leibniz Universität Hannover, Fachgebiet Software Engineering
Welfengarten 1, 30167 Hannover
{Jianwei.Shi, Kurt.Schneider}@inf.uni-hannover.de

Um Kundenbindung in der Softwareentwicklung zu gewährleisten, kann ein Kunde einen Testlauf beobachten und dann Rückmeldung geben. Allerdings programmieren die Kunden nicht und verstehen keine Testskripte. Ein Video kann anhand von Testskripten erstellt und mit Annotationen abgespielt werden. Kunden nehmen dann am Testen teil, indem sie das Test-Video sehen und Rückmeldung geben. Darüber hinaus kann das Test-Video für interne Kommunikation als Dokumentation benutzt werden.

1. Motivation

In agiler oder hybrider Softwareentwicklung sind Testen und Kundenbildung gleichermaßen wichtig. Testen und Fehlerbehebung halten Softwarequalität hoch. Kunden nennen ihre Anforderungen und geben Rückmeldungen zur Software. Allerdings sind Kunden oft nach der Anforderungserhebung nicht mehr involviert. Kunden sehen implementierte Funktionalitäten, die sich gar nicht wünschen. Das verschwendet Zeit und Geld. Wir möchten Kunden, im Sinne agiler Entwicklung, auch noch kurz nach dem Testen einbinden. Ein Video wird als Nebenprodukt von Testläufen erstellt. Kunden sehen das Video und äußern 1. ob die gewünschte Funktionalität korrekt umgesetzt wurde, und 2. ob sie neue Anforderungen haben.

Spillner et al. [1] meinen, dass „zwischen Entwicklung, Test und Anwendern durchaus Meinungsunterschiede bestehen können, ob ein bestimmtes Systemverhalten als Fehlerwirkung, als falsche Interpretation der zugrunde liegenden Anforderungen durch den Test oder als (berechtigter oder unberechtigter) Erweiterungs- oder Änderungswunsch zu betrachten ist“. Dieses Phänomen ist darauf zurückzuführen, dass Tester, Entwickler, und Anforderungsingenieur nicht das gleiche Verständnisniveau von den Anforderungen haben. Alle genannten Rollen sollten mit einem gemeinsamen Medium kommunizieren können: mit einem Test-Video. Dann kann das Video für die interne Kommunikation verwendet werden.

2. Hintergrund und Lösungswege

Unit-Test wird für Testgetriebene Entwickelung (TDD) und Testautomation (TA) eingesetzt. Unit-Tests werden in TDD vor der eigentlichen Entwicklung geschrieben. Bestehende Unit-Tests können in TA automatisch ausgeführt werden, um die Software regelmäßig zu prüfen. Wir schlagen vor, Unit-Tests darüber hinaus weiterzuverwenden: wir zeigen Kunden den Unit-Testlauf. Allerdings prüft ein Unit-Test eine kleine Funktion der Software und läuft schnell. Daraus ergeben sich zwei Probleme: Ein Kunde könnte den Test 1. schwer verstehen und 2. kaum verfolgen.

Um das erste Problem zu lösen, benutzen wir **GUI-Unit-Tests**: **Das sind Tests auf einer grafischen Benutzeroberfläche (GUI), die mehrere Interaktionsschritte mit nebeneinander stehenden Steuerelementen dieser Oberfläche enthalten.** GUI-Unit-Test ist eine Erweiterung von Unit-Test. GUI-Unit-Tests werden zu einer Testsuite kombiniert. Wenn die Testsuite läuft, sieht man GUI-Interaktionen statt Quellencode. Die Lösung des zweiten Problems besteht vor allem darin, den Testlauf zu verlangsamen. Sinnvolle Informationen können im Video gezeigt werden, z.B. Hervorhebung der GUI-Interaktion, kurze textuelle Beschreibung eines GUI-Unit-Tests.

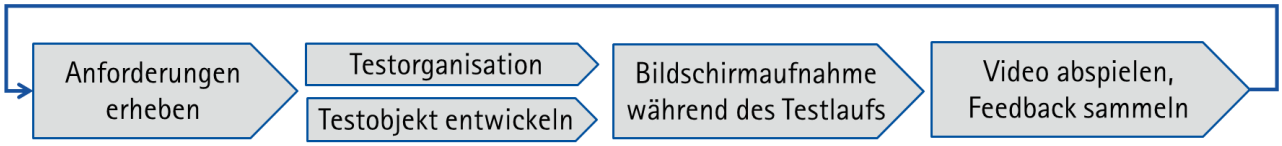

Abb. 1. Erstellung und Einsatz von Test Videos

3. Vorgehen

Abb. 1 zeigt eine Übersicht für Erstellung und Einsatz von Test-Videos. Zunächst werden Anforderungen erhoben. Danach wird ein Testobjekt (Software mit GUI) entwickelt. Parallel werden Tests ausgewählt und organisiert. Ein Video wird während des Testlaufes aufgenommen. Kurz danach sehen Kunden das Video. Zuletzt sammelt das Entwicklungsteam Rückmeldungen als Anforderungen und startet nächste Runde.

Die verhaltensgetriebene Entwicklung (BDD) fokussiert auf das Soll-Verhalten eines Systems bei Kundenansicht. Wir möchten mit BDD einen Testlauf kundenfreundlich machen:

1. Verhalten beschreiben. Ein Anforderungsingenieur beschreibt das Soll-Verhalten, z.B. mit Gherkin [2].

2. Mit Kunden verständigen. Anforderungsingenieur verständigt sich über beschriebenen Verhalten mit Kunden und passt die Verhaltensbeschreibungen an.

3. GUI-Unit-Tests entwickeln. Laut Verhaltensbeschreibungen entwickeln Tester GUI-Unit-Tests.

4. GUI-Unit-Tests organisieren. Tests werden ausgewählt und in einer Testsuite kombiniert.

A. Organisation der GUI-Unit-Tests

Das Ziel ist, dass Kunden eine Testsuite verstehen können. Vor allem sollen entsprechende GUI-Unit-Tests in einer zeitlichen Reihenfolge sein, als würde man die Software bedienen. Der Übergang zwischen zwei nebeneinander GUI-Unit-Tests soll möglichst kontinuierlich sein, d.h. Oberfläche bei Beendung des vorderen Tests ist fast gleich wie die bei Start des hinteren Tests.

B. Erstellung von Test-Video

Das neue Konzept zu Erstellung eines Videos mithilfe von Testskripten wurde in unserer vorherigen Arbeit [3] vorgeschlagen. GUI-Interaktionen (z.B. Knopfdrücken, Feldeingabe) können hervorgehoben werden [3], damit man bei Wiedergabe Interaktion klarer sehen kann.

Das Video zeigt Testläufe von GUI-Unit-Tests. Beginn und Ende des Zeitpunktes jedes GUI-Unit-Tests wird abgespeichert. Damit kann man beim Betrachten des Videos schnell auf einen GUI-Unit-Test positionieren.

Im Test könnte langer Stillstand wegen langsamer Datenbank- bzw. Internetverbindung passieren. Diesen Stillstand verkürzen wir im Video, damit Kunden das Wichtigste in kurzer Zeit sehen.

C. Einsatz von Test-Video

Ein Anforderungsingenieur spielt das Video mit hilfreichen Informationen ab. Kunden betrachten das Video und geben gezielte Rückmeldungen. Nach Beginn und Ende jedes GUI-Unit-Tests wird das Video segmentiert. Man kann einen Titel zu jedem Segment sehen und kann durch Klicken einer Segmentleiste den Beginn positionieren. Andere Interaktionsmöglichkeiten sind auch möglich: Pham et al. [4] und Tandun [5] haben folgende Funktionalitäten implementiert: Zu vorherigem und nächstem Bild springen, Wiedergabegeschwindigkeit anpassen, Konsolenausgabe parallel abspielen.

Kunden können das Video mit langsamer Wiedergabegeschwindigkeit und Hervorhebungen der GUI-Interaktionen verfolgen. Rückmeldungen können gezielt zu einem Segment, d.h. zu einem GUI-Unit-Test, gegeben werden. Damit könnten Anforderungsingenieuren besser Rückmeldungen bzw. Anforderungen organisieren.

Tandun [5] hat eine Benutzerstudie unter Testern durchgeführt, ob Tester durch ein Test-Video den Fehler besser verstehen. Darüber hinaus können Anforderungsingenieur, Tester und Entwickler vom Video-Betrachten profitieren.

4. Fazit und Ausblick

Wir haben Prozesse vorgeschlagen, wie ein Entwicklungsteam Kunden durch Test-Videos auch kurz nach dem Test noch enger einbindet. Dadurch sammelt das Team Rückmeldungen und kann vor und während der Entwicklung Anforderungen sammeln und validieren.

Das Video dient auch als Diskussionsmedium im Team, damit Anforderungen richtig verstanden werden. Dazu planen wir eine Benutzerstudie.

Literaturen

[1] A. Spillner, T. Roßner, M. Winter, und T. Linz, *Praxiswissen Softwaretest - Testmanagement: Aus- und Weiterbildung zum Certified Tester - Advanced Level nach ISTQB-Standard*. Heidelberg: dpunkt, 2014.

[2] S. Rose, M. Wynne, und A. Hellesøy, *The Cucumber for Java book: behaviour-driven development for testers and developers*. Dallas, Tex [u.a.], 2015.

[3] J. Shi und K. Schneider, „Creation of Human-friendly Videos for Debugging Automated GUI-Tests“, in *Testing Software and Systems*, Cham, 2021, S. 141–147. doi: 10.1007/978-3-031-04673-5_11.

[4] R. Pham, H. Holzmann, K. Schneider, und C. Brüggemann, „Tailoring video recording to support efficient GUI testing and debugging“, *Software Qual J*, Bd. 22, Nr. 2, S. 273–292, Juni 2014, doi: 10.1007/s11219-013-9206-2.

[5] M. A. Tandun, „Report-Video Production for Quick Bug-finding in the Web Applications“, Bachelor Thesis, Leibniz Universität Hannover, Hannover, 2022. [Online]. Verfügbar unter: https://www.pi.uni-hannover.de/fileadmin/pi/se/Stud-Arbeiten/2022/BA_Tandun_2022.pdf