Testen vs. Verifikation – Warum automatisiertes Testen eine Illusion ist

Thomas Papendieck, OPITZ Consulting Deutschland GmbH

Aus unserem täglichen Leben sind Computersteuerungen nicht mehr wegzudenken. Wir sind umgeben von unzähligen Geräten, die von Software gesteuert sind. Das Funktionieren dieser Geräte ist essenziell für unsere moderne Lebensweise. Die Folgen eines Versagens reichen von kleineren Unannehmlichkeiten, beispielsweise wenn die Kaffeemaschine falsch dosiert, bis hin zu Todesfällen, von denen das Versagen der Assistenzsysteme bei Tesla [1] oder der Flugzeugsteuerung in der Boing 737-Max [2] nur die Spitze des Eisberges mit medialer Beachtung darstellen. Je nach Gefährdungsklasse sind daher umfangreiche Tests notwendig, um das Verhalten eines Geräts und seiner Softwaresteuerung in allen, besonders aber in kritischen Situationen des Anwendungsbereichs, sicherzustellen. Obwohl es wahrscheinlich niemanden gibt, der dem widersprechen würde, ist es in der Software-Industrie so, dass gerade das Testen der Software häufig als Kostentreiber mit dem geringsten Einfluss auf das Endprodukt betrachtet wird. Die Automatisierung von Tests bietet dabei die Möglichkeit, ein erhebliches wirtschaftlichen Potenzial freizusetzen, ohne die Sicherheit des Produkts aufs Spiel zu setzen. In diesem Artikel möchte ich die Grenzen dieses Ansatzes aufzeigen.

Probleme manuellen Testens

Das Personal

Das Testen von Software – in erster Linie manuelles Testen – ist eine komplexe Aufgabe, die umfangreiche Kenntnisse und bestimmte Persönlichkeitseigenschaften erfordert *[3]*. Diese Tatsache wird leider gerade in Unternehmen, die weniger im Fokus der Öffentlichkeit stehen, ignoriert. Oft wird manuelles Testen der Software genutzt, um junge Mitarbeiter in das Projekt „einzuführen", obwohl sie die notwendigen Kenntnisse nicht haben, weder über das Testen selbst noch über die Fachlichkeit, die zu testen ist. Die Erwartung ist vielmehr, dass das Testen des Produkts die Einarbeitung in die Fachlichkeit darstellt. Die notwendigen Kenntnisse über das Testen werden dabei regelmäßig unterschätzt. Ob der neue Mitarbeiter sich von seiner Persönlichkeitsstruktur her überhaupt zum Tester eignet, wird in der Regel überhaupt nicht bedacht. Dies führt dazu, dass Tests nicht mit der nötigen Sorgfalt ausgeführt werden.

Die Ergebnisse der Testläufe können aufgrund der fehlenden Fachkenntnisse falsch interpretiert werden. Darüber hinaus findet dann die Dokumentation der Testläufe unvollständig statt. Die von nicht qualifiziertem Personal durchgeführten manuellen Tests sind also von fragwürdigem Wert. Es bleibt zu hoffen, dass sich diese Erkenntnis in den softwareproduzierenden Unternehmen weiter ausbreitet und die Verantwortlichen bereit sind, das Personal zu investieren, das ihr Produkt testen soll, um letztlich die Kosten der Tests zu senken, beispielsweise durch Automatisierung der Testläufe.

Die Wiederholbarkeit

Eine wichtige Testart bei der Prüfung von Software sind Regressionstests *[4]*. Deren wichtigste Eigenschaft ist die Wiederholbarkeit. Das bedeutet konkret, dass mehrere aufeinander folgende Testläufe zum selben Ergebnis führen müssen, sofern sich das Programm (im getesteten Bereich) nicht geändert hat. Leider sind wir Menschen nicht gut darin, komplexe Dinge in exakter Weise zu wiederholen. Zudem werden solche „Wiederholungstests" oft anhand schriftlicher Aufzeichnungen gemacht. Das sind meistens Listen mit Anweisungen dazu, welche Eingaben zu machen und welche Ausgaben als „korrekt" zu bewerten sind. Diese Aufzeichnungen enthalten häufig unpräzise Angaben, sodass sich für den Tester Interpretationsspielraum auftut, den er entsprechend seinem oft mangelhaften Wissen über die fachlichen Anforderungen ausschöpft.

Aus diesem Grund ist bei einem fehlgeschlagenen manuellen Regressionstest immer zu prüfen, ob sich tatsächlich Fehler in das Programm eingeschlichen haben oder ob der Fehler „nur" in der Testausführung liegt. Auf der anderen Seite kann ein manueller Testlauf als erfolgreich eingestuft werden, obwohl tatsächlich ein Fehler im Produkt vorliegt, weil die Ausführung des Testlaufs fehlerhaft war und so „um den Fehler herum" getestet wurde. Diese Unsicherheit schlägt sich nicht nur direkt auf den zu betreibenden Aufwand und damit auf die Kosten des Tests nieder, sondern auch auf die Produktsicherheit.

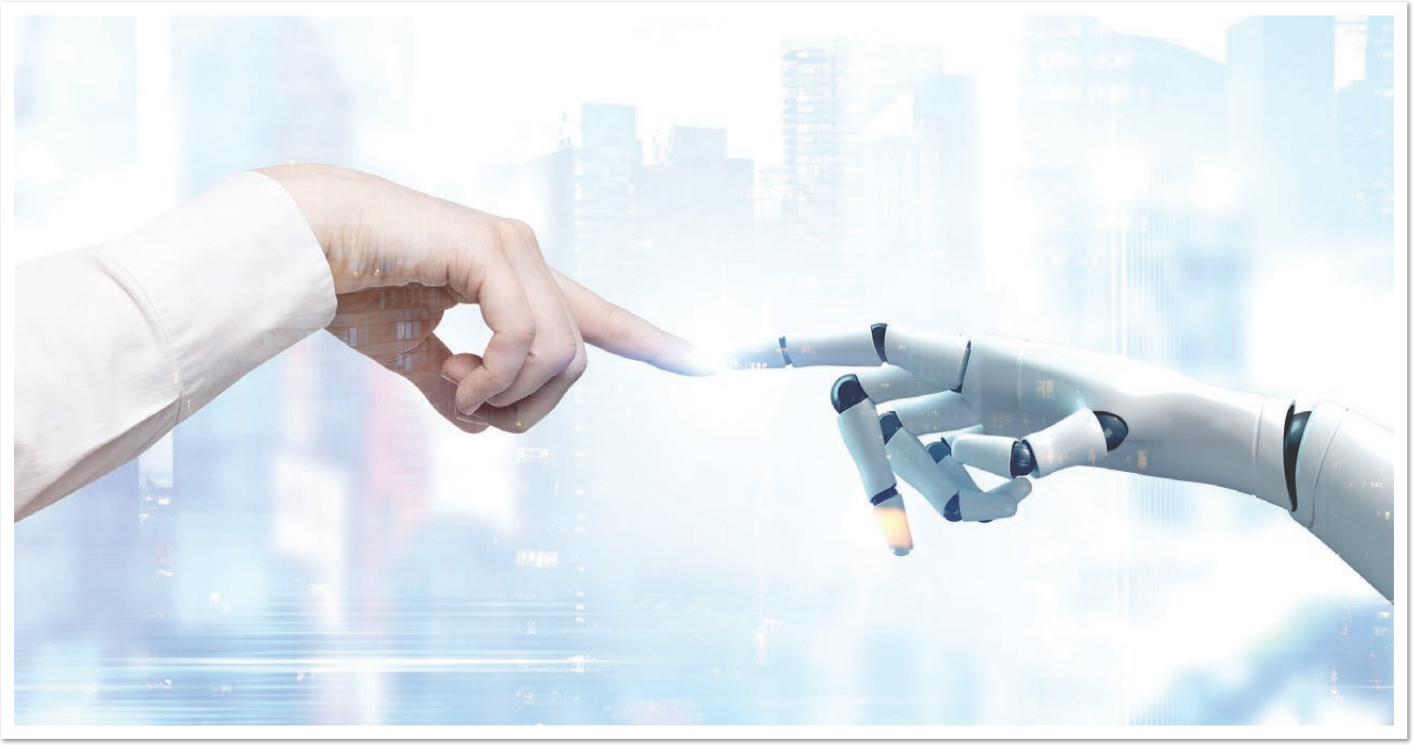
Der Zeitfaktor

Genau wie im vorigen Abschnitt ist auch dieser Nachteil manueller Tests direkt in der Limitierung menschlicher Fähigkeiten begründet.

Dabei sind zwei Einflussfaktoren zu unterscheiden, die den Zeitfaktor als kritischen Aspekt manuellen Testens in den Fokus rücken. Da ist zunächst die Tatsache, dass der Mensch sowohl kognitive als auch mechanische Tätigkeiten nur mit einer begrenzten individuellen Geschwindigkeit ausführen kann *[5]*. Aufgrund dessen kann ein Mensch nur eine begrenzte Anzahl von Testfällen in einer vorgegebenen Zeiteinheit ausführen. Diese Geschwindigkeit lässt sich im Prinzip dadurch steigern, dass der Tester die manuellen Tests häufiger durchführt. Diese durch „Training" zustande gekommene Steigerung hat aber Grenzen. Zudem birgt dieses „Training" auch Risiken, die sich auf die Testqualität auswirken. Ein Großteil des Geschwindigkeitsgewinns wird durch eine Verringerung der Aufmerksamkeit erreicht. Und ein weniger aufmerksamer Tester kann leichter einen Fehler bei der Testausführung machen oder ein problematisches Testergebnis falsch interpretieren.

Auf der anderen Seite ist das manuelle Testen von Software in besonderer Weise mental anstrengend, eben weil das Aufmerksamkeitslevel hier höher sein muss als bei anderen Tätigkeiten im Software-Entwicklungsprozess. Stellt man diese mentale Anstrengung der eines Programmierers gegenüber, zeigt sich, dass diese sich in ihrer Struktur unterscheiden.

So stehen dem Programmierer verschiedene Werkzeuge zur Verfügung, die dafür sorgen, das Flüchtigkeitsfehler schnell erkannt werden und so keinen Einfluss auf das Ergebnis nehmen können. Das beginnt bei der IDE und dem Compiler, die beispielsweise Tippfehler im Code aufdecken, bis zum Test-Driven-Development, das als Arbeitsmethode gegen versehentliche Änderungen bereits bestehender Funktionalität schützen kann. Das führt dazu, dass der Programmierer seine kognitiven Ressourcen in eine Richtung lenken kann, die sich vorwiegend mit der Erweiterung des Produktes und weniger mit der eigenen Fehlbarkeit beschäftigen. Auf diese Weise ist die Arbeit des Programmierers, obwohl ebenfalls kognitiv sehr anspruchsvoll, weniger erschöpfend. Ein Tester kann



bei der Ausführung manueller Tests keine adäquaten Hilfsmittel einsetzen, weil es sie schlicht nicht gibt.

In der Konsequenz muss die Zeit, in der ein Tester manuelle Tests der Anwendung ausführt, von mehr oder weniger regelmäßigen Pausen unterbrochen werden, damit seine Konzentrationsfähigkeit erhalten bleibt. Jedoch führt jede Unterbrechung einer vorwiegend kognitiven Tätigkeit dazu, dass die Person nach dieser Unterbrechung sich erneut in die Situation „hineinfinden" muss, was zu einer Verzögerung über die reine Pausenzeit hinausführt [6].

Eine weitere Restriktion, die den Zeitverlauf manueller Tests beeinflusst, ist der arbeitsrechtliche Rahmen, in den der Tester eingebunden ist. Die Arbeitszeit des Testers ist die begrenzende Zeiteinheit, in welcher der Tester die manuellen Tests durchführt. Nun kann man durchaus versuchen, diese Restriktion durch den Einsatz von Personal in anderen Zeitzonen auszugleichen, dies setzt jedoch eine gewisse Mindestgröße des Unternehmens voraus und ist daher kein universeller Ansatz. Außerdem hat der Einsatz mehrerer Personen wieder eigene Probleme, die ein schlichtes Addieren der Arbeitszeiten zu einer Milchmädchenrechnung degradieren.

Computer schlägt Mensch

Alle oben aufgeführten Probleme werden durch Testautomatisierung gelöst. Im Gegensatz zu uns Menschen sind Computer sehr gut darin, einmal festgelegte Abläufe beliebig oft exakt und mit hoher Geschwindigkeit zu wiederholen. Automatisierte Tests können unabhängig von Arbeitszeitgrenzen rund um die Uhr ausgeführt werden und lassen sich problemlos auf mehreren Computern gleichzeitig ausführen. Ein zusätzlicher Server ist meist kostengünstiger, als einen weiteren menschlichen Tester einzustellen. Wenn man Zeit sparen muss, ist bei automatisierten Tests der KIWI-Ansatz („Kill it with Iron") also durchaus eine Option, besonders, wenn man dabei auf virtuelle Rechner in der Cloud zurückgreifen kann.

Wo liegt das Problem?

Das Problem ist zunächst einmal ein sprachliches. Im üblichen Sprachgebrauch wird nämlich der Begriff des Testens falsch eingesetzt. Wenn man sich außerhalb des Kreises der „Wissenden" über das Testen unterhält, meint man nur einen Teilbereich dessen, was laut ISTQB (International Software Testing Qualifications Board) als „Testen" definiert ist. Deren Definition für das Testen lautet:

> ***„Der Prozess, der aus allen Aktivitäten des Lebenszyklus besteht (sowohl statisch als auch dynamisch), die sich mit der Planung, Vorbereitung und Bewertung eines Softwareprodukts und dazugehöriger Arbeitsergebnisse befassen. Ziel des Prozesses ist sicherzustellen, dass diese allen festgelegten Anforderungen genügen, dass sie ihren Zweck erfüllen, und etwaige Fehlerzustände zu finden [7]."***

Im allgemeinen Sprachgebrauch reduzieren wir das Testen dann aber auf den Teil, der als „dynamische Tests" bezeichnet wird, wobei wir nicht zwischen der Erstellung der Testfälle und deren Aus-

führung unterscheiden, sondern lediglich Letzteres im Blick haben. Dabei kennt das ISTQB einen gesonderten Teilprozess für die Testausführung, den sie aus der ISO 9000 übernommen haben, nämlich die Verifizierung:

> *„Bestätigung durch Bereitstellung eines objektiven Nachweises, dass festgelegte Anforderungen erfüllt worden sind [8].“*

Warum ist es nun wichtig, auf diese, auf den ersten Blick erbsenzählerisch erscheinende, Unterscheidung zu bestehen? Der Grund ist, dass die im Sprachgebrauch vernachlässigte Erstellung der Testfälle im Gegensatz zu deren Ausführung, also der Verifizierung, Kreativität erfordert. Ohne diese Unterscheidung müssen wir zwangsläufig eine falsche Vorstellung darüber entwickeln, was Testautomatisierung tatsächlich leisten kann.

Grenzen der Testautomatisierung

Die Testautomatisierung stellt Werkzeuge bereit, mit denen von Menschen in einer maschinenlesbaren Form definierte Testfälle von diesen Testwerkzeugen ausgeführt und die Reaktionen des getesteten Systems bewertet werden. Das Produkt wird also gegen die in den Testfällen definierten Kriterien verifiziert. Diese Definition eines Testfalls beinhaltet immer die Kriterien für die Bewertung der Reaktion des getesteten Systems. Diese Kriterien müssen derzeit zwingend von einem Menschen gemacht werden.

Einige Werkzeuge scheinen zwar in der Lage zu sein, diese Entscheidung selbst treffen zu können, beispielsweise solche, die das zu testende System mit zufälligen Eingaben füttern, aber bei genauer Betrachtung zeigt sich, dass auch hier ein Mensch bei der Erstellung des Tests den Bereich der Eingabewerte einschränken muss, da das Werkzeug nicht selbstständig entscheiden kann, welche Reaktion des Systems auf die zufälligen Eingabewerte gewünschtes Verhalten ist und welches nicht. Auf absehbare Zeit wird es die künstliche Intelligenz nicht schaffen, dem Menschen diese Entscheidung abzunehmen.

Fazit

Testautomatisierung ist ein großartiges Mittel, um nicht nur die Kosten und die Dauer der Testphase zu senken, sondern auch die Qualität der Tests zu erhöhen. Sie schafft dies, indem sie den Menschen mit seinen Einschränkungen als Störfaktor und Fehlerursache aus der Testausführung eliminiert. Sie übernimmt den Teil, der nach der Definition des ISTQB als Verifizierung bezeichnet wird.

Auf der anderen Seite werden weiterhin hochqualifizierte Personen mit dem Testen von Software beschäftigt sein, indem sie die Werkzeuge der Testautomatisierung nutzen, um neue Testfälle zu definieren. Die Werkzeuge der Testautomatisierung können die Tester zwar bei der Erstellung von Testfällen unterstützen, indem sie über Vorlagen und Schablonen formale Aspekte der Testfall-Generierung sicherstellen, aber in absehbarer Zeit werden sie den kreativen Akt nicht übernehmen können, in dem ein Mensch Testfälle aus den Anforderungsdokumenten extrahiert. Bis auf Weiteres wird automatisiertes Testen in dem Sinne, dass selbst die Testfälle selbstständig von den Werkzeugen erstellt werden und der Tester den Prozess nur noch überwacht, eine Illusion bleiben.

Quellen

[1] Gregor Hebermehl: Zwei Personen im gerammtem Honda Civic sterben *https://www.auto-motor-und-sport.de/elektroauto/tesla-autopilot-unfall-honda-civic/*

[2] Frank Bäumer: Die vermeidbare Katastrophe *https://www.tagesschau.de/wirtschaft/boeing-235.html*

[3] ALPHAJUMP: Was macht ein Softwaretester? *https://www.alphajump.de/karriereguide/beruf/softwaretester*

[4] Wikipedia: Regressionstest *https://de.wikipedia.org/wiki/Regressionstest*

[5] CogniFit: Verarbeitungsgeschwindigkeit. *https://www.cognifit.com/de/wissenschaft/kognitive-fahigkeiten/Verarbeitungsgeschwindigkeit*

[6] Christian Steinbrich: Hol dir die Kontrolle über deine Konzentration zurück! – Wirksame Ablenkungs-Killer für besseres Lernen *https://www.121watt.de/produktivitaet/besser-lernen-mehr-konzentration/*

[7] International Software Testing Qualifications Board: Glossar „Testen” *http://glossar.german-testing-board.info/v3.21/#t,*

[8] International Software Testing Qualifications Board: Glossar „Verifizierung”

Thomas Papendieck

OPITZ Consulting Deutschland GmbH

Thomas.Papendieck@opitz-consulting.com

Thomas Papendieck ist seit 20 Jahren in der Softwareentwicklung tätig, davon seit 15 Jahren bei OPITZ CONSULTING Deutschland GmbH. Er ist gelernter Elektrotechniker und studierte zunächst Radar-Technik und später Informatik.