– Verlagsbeilage –

GERMAN TESTING MAGAZIN
Das unabhängige Magazin zu Software-Qualität

LEAN QUALITY MANAGEMENT

HOUSE OF AGILE TESTING - QUALITÄTSSICHERUNG IM KONTEXT GROSSER AGILER SOFTWARE-ENTWICKLUNGSPROJEKTE

AGILITÄT UND QUALITÄT: VON DER IT IN DIE GESCHÄFTSBEREICHE

LEAN AGILE QUALITYMANAGEMENT - WIE INTEGRIERE ICH QUALITÄTSSICHERUNG IN DIE SKALIERT-AGILE IT-SYSTEMENTWICKLUNG

VOM SOFTWARETEST ZU QUALITY-ENGINEERING – INTERVIEW MIT PANKAJ RAI JAIN, GESCHÄFTSFÜHRER BEI ACCENTURE

ALMATO just add digital

App-Testing

Sie entwickeln, wir testen.

Agiles Testen durch nahtlose Integration in Ihre Entwicklungsprozesse

Qualitativer, manueller Test Ihrer App durch professionelle Tester hinsichtlich:

- **Funktionalität**
- **Usability**
- **Performance und**
- **Kompatibilität**

Testplan, Bugreport und Optimierungsempfehlungen

Umfangreicher, stets aktueller Testgerätepool

UX-Review nach ISO 9241-110

Zertifizierte Testmanager nach:

ISTQB® MOBILE APPLICATION TESTING
ISTQB® AGILE TESTER
UXQB® USABILITY & USER EXPERIENCE

PREIS-LEISTUNGS-SIEGER

Buchen Sie jetzt Ihr App-Testing!

www.almato.com

Sehr geehrte Leserinnen und Leser,

ganzheitliches Leben bedeutet das Leben mit allen Sinnen, mit intellektuellen und emotionalen Aspekten. Nur auf ein Pferd zu setzen, funktioniert selten gut. Moderne Konzepte setzen genau auf so einen gesunden und ganzheitlichen Mix aus körperlicher Arbeit, emotionaler Erfahrung und Denkarbeit – und schaffen dadurch Erfolge.

Wie lässt sich das auf die Welt in der Industrie und insbesondere auf die Qualitätssicherung übertragen? Heutige qualitätssichernde Prozesse sind oftmals verkopft und nicht (emotional) gefühlt in jedem Atemzug der Mitarbeiter. Wir brauchen das aber, damit alle das Ziel kennen und gemeinsam zur Zielerreichung beitragen. Wie lassen sich Test und Qualität also ganzheitlich umsetzen? Wie bringen wir Qualität in die DNA unserer Mitarbeiter? Wie starten wir schon zu Beginn, an die Dinge ganz am Ende zu denken? Müssen wir das überhaupt? Welche Entwicklungsmethoden sind am erfolgversprechendsten? Welche Rolle spielt der agile Ansatz und wie setzt man ihn um? Wie können wir einen entsprechenden Change-Prozess für unser Unternehmen aufsetzen und die daraus resultierenden Herausforderungen bewältigen? Wie skalieren wir agile Testmethoden in großen Organisationen? Und was zählt eher zu Muda?

Die Antworten auf diese und verwandte Fragen behandeln die Autoren in dieser Ausgabe des German Testing Magazins. Mit dabei sind Erfahrungsberichte, Methoden und Arbeitsmodelle beigetragen von mittelständischen und großen Unternehmen. Wir freuen uns, Ihnen dabei Artikel der Deutschen Bahn, von Volkswagen, der Raiffeisen Bank und von vielen mehr zu präsentieren. Zudem lässt uns Accenture in einigen ihrer Artikel an Erfahrungen und Erkenntnissen aus ihren Projekten teilhaben – vom House of Agile Testing über Qualität mit SAP bis hin zu agilen Vorgehensmodellen. Besonders möchten wir auf das Interview mit Pankaj Rai Jain aufmerksam machen – dem Geschäftsführer von Accenture in München. Ein Ausblick auf die Ergebnisse der Softwaretest-Umfrage 2020 aus der Zusammenarbeit des German Testing Boards, des Austrian Testing Boards und der TH Köln und der Hochschule Bremerhaven rundet die Ausgabe ab.

Wir wünschen Ihnen viele nutzbringende Anregungen, aber auch viel Spaß beim Lesen der vielen interessanten Artikel. Auf dass diese Ausgabe ein ganzheitliches Lesevergnügen für Sie wird!

Ihre

Dr. Stephan Weißleder und Richard Seidl (Chefredaktion)

Inhalt

»Testing auf dem Weg in ein „neues Level"«

In diesem Artikel möchten wir Ihnen einen Überblick und Einblicke in die zukünftigen Werkzeuge, Herausforderungen und Methoden des Testens geben. In diesem Zusammenhang fällt immer wieder der Begriff „Lean Quality Management". In unseren Augen ist Lean Quality Management der „Hidden Champion" – der „verborgene/unsichtbare Champion" auf dem Weg zum Idealzustand der Qualitätssicherung und der Enabler für „Business Agility". Wir freuen uns, Ihnen mit diesem Überblicksartikel unseren Blick auf das zukünftige Arbeitsumfeld der Softwareindustrie zu schildern und Ihnen damit die inhaltliche Grundlage aller folgenden Artikel näherzubringen.

„Wo ist unser Test geblieben?" – Alle sprechen von Qualität, der Wichtigkeit der IT sowie Automatisierung. Ist *Lean Quality Management* die Antwort auf die Frage nach der Zukunft des Tests? *Ja, auf jeden Fall!* Die Arbeit in unseren Qualitätsdepartments hat sich in den letzten Jahren und Jahrzehnten stetig weiterentwickelt und steht nun davor, wieder eine neue Schwelle zu erreichen. Um diesen Übergang und um die Methoden und Werkzeuge, die uns dahin bringen, soll es in dieser Ausgabe des German Testing Magazins gehen.

Testing im Wandel der Zeit

Wir sehen drei Level, die in vielen Programmen bis dato erreicht wurden, und ein viertes, mit dem wir uns aktuell konfrontieren.

Level 1: Initial arbeiteten wir in *Testing-Silos im Wasserfall* oder ähnlichen Konstellationen. Hier ging es um das reine Testing: Vorgegebene Anforderungen wurden einem Team gegeben, dort regelmäßig – meist mit manuellen Tests – umgesetzt und ausgeführt. Oft relativ unabhängig, in eigenen Abteilungen „bestenfalls" unabhängig vom Entwicklungsgeschehen, um die Testphasen optimal vorzubereiten, auszugestalten und nachzubereiten.

Level 2: In den 2000er-Jahren begann in vielen Bereichen die *agile Reise*, das Testing rückte viel näher an die Entwicklung und an das Anforderungsmanagement, respektive User-Story-Management heran. Hier wurde sichtbar, welche Herausforderungen es mit sich bringt, die Transformation weg vom reinen Testing anzuvisieren. Der Mind Change sowie auch das technische Verständnis und Wissen musste in vielen Bereichen erst aufgebaut werden. Genauso die Beteiligung im Team und der Umgang mit den menschlichen Aspekten und Herausforderungen in diesen Bereichen wurden immer sichtbarer. Einige Softwareentwicklungsprogramme stecken hier noch sehr tief drin und arbeiten daran, diese Herausforderung zu meistern. Die Unternehmen beschäftigten sich auf diesem Level mit Fragen wie:

- Was bedeutet es, Qualität im ganzen Entwicklungsprozess zu verankern?
- Wie kann Softwaretest als gemeinsame Verantwortung im Team gelebt werden?
- Wie steigere ich die Selbstorganisation und Motivation für den Test im Team?
- Welche Strategien nutze ich für die Automatisierung?

Antworten auf diese und viele weitere Fragen finden sich in den beiden Artikeln zum House of agile Testing, einem Qualitätstransformationsframework, welches Ihr Unternehmen bei der Reise von Level 1 bis Level 4 und darüber hinaus begleitet.

Level 3: Hier sehen wir die erfolgreiche Umsetzung der gerade genannten Themen, wir haben es geschafft, ein gemeinsames Qualitätsverständnis im Team und über mehrere Teams hinweg zu schaffen. Dabei geht es nicht nur darum, den Testern die neue Welt zu zeigen, sondern genauso gut darum, die Entwickler in Richtung Test zu coachen. Die Tester-Rolle wechselt seitdem immer mehr vom reinen Doing hin zum Coaching, zur Entwicklung der QM-Strategie und schließlich hin zum *Quality Engineering*. In den Artikeln „Quality Engineering geht jeden etwas an" und „Distributed Agile Testing? Ja, es funktioniert!" werden das breite Tätigkeitsfeld und einige Praxiserfahrungen näher beschrieben. Auf diesem Level gehen die typischen Fragen meist stark in Richtung Skalierung und Compliance-Anforderungen.

Level 4: In der heutigen Zeit der disruptiven Technologien ist der Bedarf nach erfolgreichen IT-System-Entwicklungen enorm gestiegen. Während die Zeiträume, die dafür zur Verfügung stehen, immer kürzer werden, wächst gleichzeitig der Zwang, die Kundenzufriedenheit optimal zu erfüllen, denn das Angebot an Softwarelösungen auf dem Markt nimmt stetig zu. Um hier erfolgreich zu sein, benötigt es eine weitere Professionali-

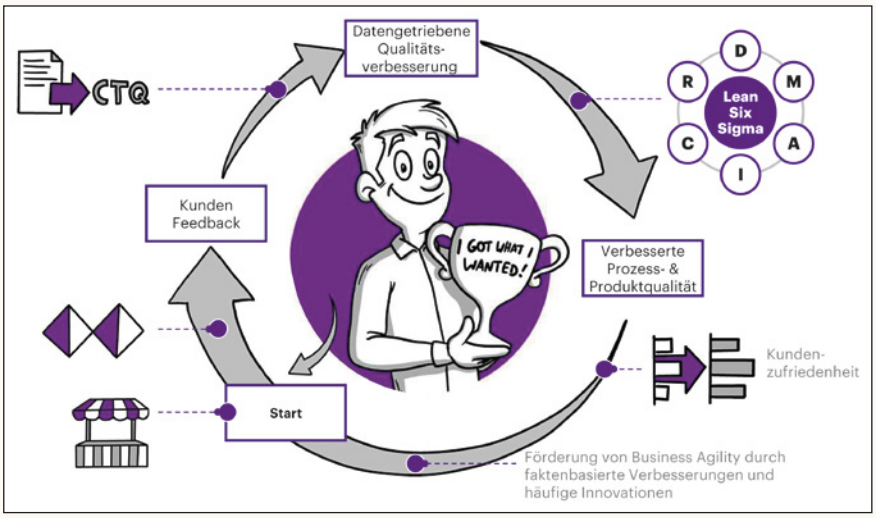


Abb. 1: Lean Quality Management als Accelerator für Business Agility

Referenzen

- [ASTA] o. A., 14th Annual State of Agile Report, 2020, siehe: **https://explore.digital.ai/state-of-agile/14th-annual-state-of-agile-report**
- [Sta15] o. A., Relevanz von Lean-Management-Methoden für die Industrie 4.0 in der Schweiz 2015, siehe: **https://de.statista.com/statistik/daten/studie/490131/umfrage/relevanz-von-lean-management-methoden-fuer-die-industrie-40-in-der-schweiz/**
- [Sta18] o. A., Umfrage zur Relevanz neuer Arbeitsmethoden für die tägliche Arbeit 2018, siehe: **https://de.statista.com/statistik/daten/studie/987155/umfrage/umfrage-zur-relevanz-neuer-arbeitsmethoden-fuer-die-taegliche-arbeit/**

sierung der Qualitätseinheiten hin zu Level 4, dem *Lean Quality Management*.

Darin enthalten ist ein professioneller Umgang mit Metriken wie beispielsweise der *CTQs* (critical to quality KPIs), ebenso wie deren vorausschauende und zielgerichtete Verwendung im kontinuierlichen Verbesserungsprozess. Auch die *Vermeidung von Verschwendung* und ein stärkerer Fokus auf das Thema Prozessqualität sind essenzielle Bestandteile von Lean Quality Management. All diese Maßnahmen finden dabei im Kontext einer *wertstromorientierten Ausrichtung* der Aufgaben mit dem Kunden im Zentrum statt. Dies ermöglicht letztlich eine gesunde Skalierung der lean-agilen Vorgehensweise, da die unterschiedlichen Systeme in der IT und der Organisation selbst ganzheitlich betrachtet werden. **Abbildung 1** zeigt unseren Ansatz, wie durch die sinnvolle Kombination von unterschiedlichen lean-agilen Methoden die *Prozessqualität* verbessert wird und dadurch *Business Agility* begünstigt und die Kundenzufriedenheit gesteigert wird.

Interessante Studien

Eine kurze Analyse der Markt- und Studienlage bestätigt unseren Ansatz. Nachfolgend sind beispielhaft einige interessante Studien aufgeführt. Die Ergebnisse einer Umfrage zur „Relevanz neuer Arbeitsmethoden für die tägliche Arbeit" [Sta18] zeigen eindrucksvoll die hohe Bedeutung von lean-agilen Methoden, so wurden „Agiles Projektmanagement", „Agile Entwicklung", „Design Thinking" und „Lean-Startup" in den Top 5 der Antworten gelistet.

Die „Relevanz von Lean-Management-Methoden für die Industrie 4.0" [Sta15], und damit auch für die Softwareindustrie, wird in einer weiteren Studie eindrucksvoll aufgezeigt:

- Über 70 Prozent der befragten Unternehmen gaben an, dass die „Wertstromorientierte Organisation von Produktion und Entwicklung" wichtig oder sehr wichtig ist.
- Über 80 Prozent der Befragten gaben an, dass die „Reduzierung der Durchlaufzeiten" wichtig oder sehr wichtig ist.
- Über 70 Prozent der Unternehmen gaben an, dass die „Verbrauchssteuerung z. B. mit Kanban" wichtig oder sehr wichtig ist.

Diese Punkte finden sich in einem ähnlichen Wortlaut auch in den großen agilen Skalierungsframeworks wieder. Die Ergänzung der unterschiedlichen agilen Methoden mit Lean-Management-Methoden stellt einen wegweisenden Durchbruch dar. Lean-agile Arbeitsweisen sind nun auf der Portfolio- und Strategieebene erfolgreich angekommen und es wird vorwiegend über „Business Agility", „Organisational Agility" und „Enterprise Agility" diskutiert. Der „Hidden Champion" auf dem Weg zu diesen Idealzuständen ist das Lean Quality Management und die damit verbundene Verbesserung der Prozessqualität! Denn 45 Prozent der befragten Unternehmen im „14th Annual State of Agile Report" [ASTA] gaben an, dass inkonsistente Prozesse und Methoden eine Hauptschwierigkeit bei der Skalierung agiler Arbeitsweisen sind.

Lean Quality Management kann helfen, diese Herausforderungen zu meistern, und damit auch zu einer Verbesserung der Softwarequalität führen, was immerhin für 42 Prozent der befragten Unternehmen der Hauptgrund für die Einführung agiler Arbeitsweisen ist [ASTA]. Lösungsansätze und Praxiserfahrungen hierzu werden unter anderem in den Artikeln „Lean Agile Quality Management – Wie integriere ich Qualitätssicherung in die skaliert-agile IT-Systementwicklung" und „Lean Agile Quality Management in einer SAFe® SAP-Umgebung – Die Bedeutung der Architectural Runway aus Managementsicht" beschrieben.

Fazit

Lassen Sie uns mit Lean Quality Management gemeinsam den Bereich Testing auf ein neues Level führen um „Business Agility" in den Unternehmen Wirklichkeit werden zu lassen.

Thomas Karl

thomas.karl@accenture.com
ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

Nico Liedl

nico.liedl@accenture.com
ist Quality Engineer und Advisor mit Schwerpunkt auf agilem Qualitätsmanagement, Testautomatisierung, Prozessverbesserung und organisatorischem Wandel. Er hat Erfahrungen bei verschiedenen komplexen Großprojekten gesammelt. Als Teammitglied, Scrum Master, Teststratege, Releasemanager und Trainer hat er praktische Erfahrung und theoretisches Fachwissen zu bieten und möchte sie mit einem breiten Publikum teilen.

»Vom Softwaretest zu Quality-Engineering«

Aus dem Testen von Software ist eine komplexe technische Disziplin innerhalb des Entwicklungsprozesses geworden. Die Gründe sind vielschichtig, weiß Pankaj Rai Jain von Accenture. Er spricht von einem Software-Qualitäts-Engineering, das sich sehr schnell weiterentwickeln wird. Im Interview teilt Jain einige seiner Vorstellungen davon, wie diese Entwicklung aussehen könnte.

Software ist heute allgegenwärtig. Anders als früher finden wir sie nicht mehr nur in den Rechenzentren und auf dem PC, sondern zum Beispiel als App auf jedem Smartphone, in den Haushaltsgeräten des Smarthomes, in modernen Verkehrsmitteln oder in den Maschinen und Robotern der Fabriken.

Diese wenigen Beispiele machen klar: Software muss funktionieren! Fehlerfrei, sicher und performant. Es gibt aber durchaus noch andere Kriterien, die wichtig für den Softwareeinsatz sind. Das sind Eigenschaften wie flexibel, schnell und einfach in Betrieb zu nehmen oder intuitiv nutzbar.

Software spielt eine zentrale Rolle in unserem täglichen Leben. Es ist schwer, sich einen Aspekt des Lebens vorzustellen, der immer noch ohne Software auskommt. Dabei sind die Anforderungen an Software höchst unterschiedlich. Das ist kein Wunder angesichts der Erwartungshaltungen ihrer Nutzer, angesichts der Anforderungen aus allen Ecken einer zunehmend digitalen Welt, angesichts der Einsatzfelder von der Unterhaltung bis hin zu geschäftskritischen und lebenserhaltenden Systemen und angesichts der schieren Menge an Software, die immer schneller produziert und geändert werden muss. All diese Unterschiede und der permanent wachsende Zeitdruck machen das Software-Engineering so kompliziert und das Testen so anspruchsvoll.

Selbst Experten wie Pankaj Jain können nicht exakt vorhersehen, wie die Softwareentwickler in zehn Jahren arbeiten werden. Seiner Meinung nach ist eines jedoch sicher: Das „Softwaretesten" im nächsten Jahrzehnt wird völlig anders aussehen als heute. Der Grund dafür liegt auf der Hand: Die derzeitigen Methoden der Qualitätssicherung werden nicht ausreichen, um die Qualität zu erreichen, die von einem viel komplexeren, hochgradig integrierten und praktisch grenzenlosen IT-System und all den miteinander verbundenen Produkten verlangt wird. Der Anspruch an die Softwarequalität wächst dabei nicht nur rein funktional, sondern auch nichtfunktional in Bereichen wie Benutzererfahrung, Sicherheit, Datenschutz oder Integrationsfähigkeit.

Das erhöht den Druck auf das Quality-Engineering, denn künftig müssen viel bessere Produkte mit völlig neuen, auch nichtfunktionalen Spezifikationen viel schneller als bisher geliefert werden. All das hat laut Jain gravierende Konsequenzen darauf, wie, was und wann getestet werden muss. Im Interview zeigt er auf, wie dieser Wandel gestaltet werden kann. Das Interview haben wir auf Englisch geführt und veröffentlichen hier eine übersetzte Fassung.

„Der Softwaretest hat sich in den vergangenen vier, fünf Jahren grundlegend gewandelt – und ist zu einem ‚Quality-Engineering' gereift!"

Herr Jain, wie würden Sie die Entwicklung des Softwaretestens in den letzten Jahren beschreiben?

In den letzten 5 bis 10 Jahren hat sich Softwaretesten enorm weiterentwickelt. Aus rudimentären Ansätzen entstand eine regelrechte Ingenieurdisziplin. Doch das reicht bei Weitem nicht aus. Das wird allein schon daran deutlich, dass Software heute praktisch überall zu finden ist – nicht nur auf Computern, sondern auch im Auto, in den Küchengeräten und auf unserem Smartphone. Und natürlich in den Fabriken, Krankenhäusern und im Einzelhandel.

Hinzu kommen innovative Technologien wie KI, Blockchain oder IoT. Diese Technologien erfordern jeweils eigene, völlig neue Strategien für den Softwaretest. Nur so können wir uns die möglichen Innovationen auch voll und ganz zunutze machen.

Was heißt das für den Softwaretest?

Der kontinuierliche Fortschritt in der IT macht deutlich, dass wir Software-Ingenieure sehr innovativ bleiben müssen. Wir müssen neue, intelligente Wege, Methoden, Werkzeuge und Techniken finden, um diese Fülle von Software mit nie dagewesener Komplexität effizient testen zu können.

Wir brauchen dazu, wie gesagt, völlig neue Strategien. Diese Strategien müssen sowohl die mit dem Softwareeinsatz verbundenen Risiken senken als auch das Nutzererlebnis sowie die Reaktionszeit verbessern.

„Wir brauchen mehr Tests, wir brauchen bessere Tests, wir brauchen ein übergreifendes Quality-Engineering!“

Sehen Sie bereits neue Strategien beim Softwaretest, Herr Jain?

Oh ja! Es gibt eine Fülle von Verbesserungen bei dem, was ich Quality-Engineering von Software nenne. Ein Beispiel dafür ist automatisiertes Testen, das fest in die DevOps-Pipeline integriert ist. Diese Integration ermöglicht das Anstoßen von Testfällen auf Basis der geplanten Software-Changes. Diese Praxis ist schon weit verbreitet – und nur ein Beispiel von vielen.

Wir sind ja Zeitzeugen des Einzugs moderner KI- und ML(Maschinelles Lernen)-Techniken in den Softwaretest. KI versetzt uns in die Lage, die riesige Menge bekannter Vorgaben und Nutzungsbedingungen, aber auch die Definition und Simulation unbekannter, völlig unerwarteter Ereignisse im Test automatisiert abzudecken.

Wir beobachten die Entwicklung von Techniken zur automatischen Extraktion von Testfällen aus der Softwarespezifikation – und zwar hochautomatisiert mithilfe von Natural Language Processing. Dies sorgt mittlerweile dafür, dass wir automatisch Testfälle generieren, die kleinere Softwareänderungen direkt berücksichtigen.

Inzwischen nutzen wir beim Testen nicht nur virtuelle Technologien wie Robotic Process Automation, sondern auch reale Roboterarme, die mit Software-Testmaschinen arbeiten. Reale und virtuelle Robots werden physische, aber auch funktionale Produkttests weiter automatisieren.

Es ist offensichtlich, dass mit der Anzahl verbundener Systeme und Prozesse auch die Anzahl unvorhergesehener Situationen so schnell wächst, dass sie nicht einmal im Labor simuliert, geschweige denn getestet werden kann. Daher müssen wir die Prozesse der Qualitätssicherung, die Fehlerprognose und -vermeidung verstärkt angehen. Dabei hilft die Analyse der historischen Entwicklungs- und Produktionsdatenprotokolle, denn sie erlaubt frühzeitig die Fokussierung auf die wichtigsten potenziellen Fehlerquellen.

Es geht also um Root-Cause-Detection und -Prevention?

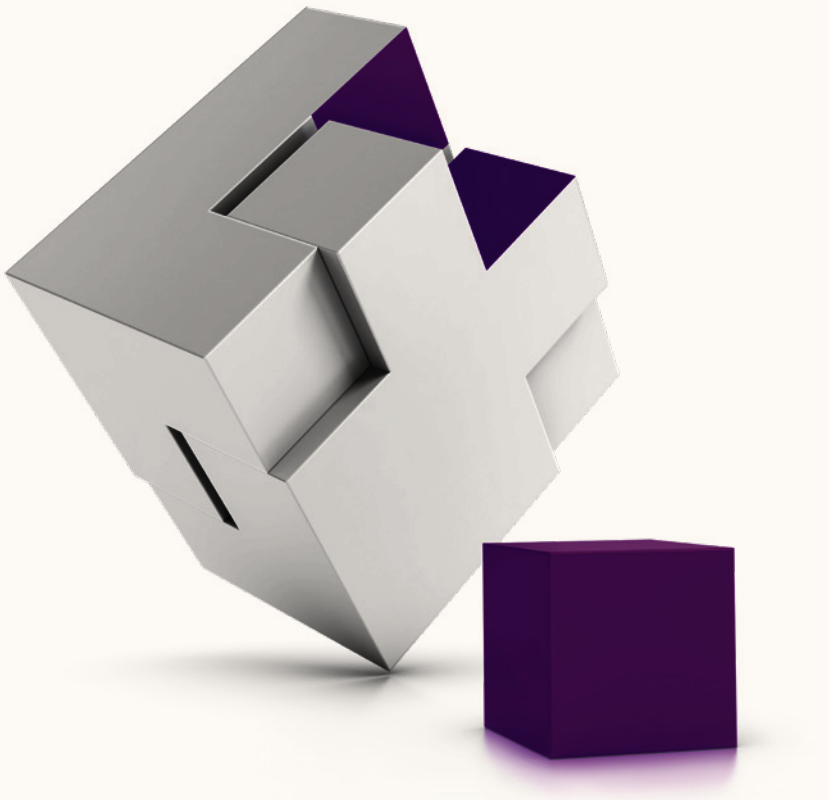
Genau! Daran wird unter Termindruck fälschlicherweise gespart. Das Gegenteil ist richtig: Je früher im Entwicklungsprozess wir Fehler finden und Fehlerursachen ausschalten, desto schneller haben wir eine funktionstüchtige Software. Und desto günstiger wird sie sein. Denn es ist zeitaufwendig und kompliziert, Fehler im Nachhinein auszubügeln. Vor allem dann, wenn die Software schon an die Kunden ausgeliefert ist.

Wie lösen wir das fundamentale Problem von Softwarequalität?

Wir brauchen einen holistischen Ansatz: Quality-Engineering. Das Testen einzelner Systeme reicht in unserer vernetzten Welt nicht mehr aus. Unser holistischer Ansatz berücksichtigt auch, dass sich Umgebungsbedingungen ändern, dass falsche Inputs geliefert werden oder dass die Software unautorisiert verändert werden könnte. All das wird mit zunehmender Digitalisierung und Virtualisierung unserer Welt immer wahrscheinlicher. Genau deshalb sind wir Qualitätsingenieure gefragter denn je. Die Kosten und Auswirkungen von Softwarefehlern sind höher als früher und steigen weiter.

Sie fordern einen holistischen Ansatz für Softwarequalität, Herr Jain. Wie sollte dieser Ansatz aussehen?

Generell werden sich Softwareentwickler eine kundenorientierte Denk- und Arbeitsweise aneignen müssen. Das heißt: Die



funktionalen und technischen Risikoprofile einer Software bleiben auch in Zukunft wichtige Inputs für den Test. Zudem wird die Kundenerwartung viel stärker als bisher berücksichtigt. Oft gilt noch die Performanz der Software als die wichtigste Eigenschaft, die der Kunde erwartet. Bei Bankern könnte es aber auch die Sicherheit sein, im Gaming-Sektor die Innovation oder im E-Commerce die User Experience (UX). Die Liste unterschiedlicher Kundenerwartungen ließe sich beliebig verlängern. Eines ist klar: Die beiden Inputs Risikoprofil und Kundenerwartung werden die Qualitätssicherungsaufgaben in Zukunft massiv bestimmen.

„Wir suchen die Schwachstellen im Design und eliminieren sie, bevor sie überhaupt Fehler verursachen können"

Wir sollten außerdem in Zukunft nicht nur auf Mängel in der Software selbst achten, sondern außerdem die Risiken der Inbetriebnahme reduzieren. Wir Software-Ingenieure müssen daher alle Risikoparameter identifizieren und gewichten – und diese Risiken dann möglichst eindämmen. Genau hier muss das Quality-Engineering der Zukunft ansetzen.

Woran machen Sie diese Verbesserungen in Richtung Quality-Engineering konkret fest, Herr Jain?

Zum Beispiel am Automationsgrad. Nehmen Sie nur die Automation der Regressionstests. Nicht nur die Testausführung, sondern auch die Fehleridentifikation und -dokumentation, die Root-Cause-Analyse, die Klassifizierung und die Verteilung von Fehlern können in der Regel automatisch durchgeführt werden. Der Quality-Engineer kann sich also voll und ganz auf die Gestaltung von neuen Testszenarien und auf die Analyse der Ergebnisse konzentrieren. Zeitraubende Routineaufgaben, wie etwa Beschaffung und Prüfung der Testdaten und Testergebnisse, laufen weitestgehend automatisch ab.

Anderes Beispiel: In der digitalisierten Welt hängen praktisch alle Geschäftsprozesse von reibungslos funktionierender Software ab. Diese Abhängigkeit nimmt mit dem Grad der Digitalisierung ebenso rasant weiter zu wie ungeahnte Abhängigkeiten der Software selbst – etwa von anderen Systemen, von der Hardware oder von unvorhersehbaren Umwelteinflüssen. Darauf müssen wir uns im Quality-Engineering vorbereiten.

„Qualitätsingenieure sind gefragter denn je, denn Softwarefehler können wir uns künftig nicht mehr leisten!"

Wie kann das gelingen, Herr Jain?

Wir müssen neue Formen des Tests von KI-Systemen erarbeiten, solche wie beispielsweise Chatbots, Cross Up-Selling (Verbundverkauf) Empfehlungen und Kampagnenmanagement-Systeme. Das Testen von KI-Systemen, um systematische Fehler zu vermeiden und falsche Ergebnisse zu identifizieren, oder das stetige Training der selbstlernenden Algorithmen stellen neue Herausforderungen für den Softwaretest dar. Damit müssen die Tester umgehen können. Anderes Beispiel: Es kann sein, dass die Software von Menschen genutzt werden soll, die im Umgang damit zuvor nicht geschult worden sind. Auch das müssen wir beim Test berücksichtigen. Die Bedienung der Software muss intuitiver und einfacher werden.

Drittes Beispiel: Die Unternehmen verfügen über enorme Datenmengen. Die Frage ist: Wie können wir daraus aussagekräftige Testdaten generieren? Testdaten, mit denen wir das Fehlerrisiko minimieren und die „Hot Spots" der Software, an denen Fehler besonders gravierende Konsequenzen hätten, gut und intensiv testen können. Und wie können wir es vermeiden, dass Fehler erst in letzter Sekunde entdeckt werden? Auch hier kann KI unterstützen.

Zu guter Letzt können wir die Qualitätssicherung effizienter gestalten; wurden bei der Softwareentwicklung früher 30 bis 35 Prozent Aufwand für die Qualitätssicherung reserviert, kommen wir heute dank KI und Automatisierung mit 25 Prozent aus – Tendenz weiter sinkend. KI wird uns aber noch viel weiterführen, etwa in containerisierten Softwareumgebungen. Hier versuchen wir, automatisch End-to-End-Testfälle mit KI-Unterstützung zu generieren. Bisher müssen diese Testfälle manuell erstellt werden, was ineffizient und fehleranfällig ist.

Sollte nicht auch das Testen selbst einfacher und agiler werden?

Ja und nein, während der aktuelle Testing Scope einfacher und schneller wird, kommen neue Herausforderungen und gesteigerte Anforderungen hinzu. Das für Testing zur Verfügung stehende Zeitfenster wird aufgrund der agilen Methoden der Softwarebereitstellung immer kleiner. Daher sind neben einer massiven Automatisierung neue Ideen gefragt, wie und was genau getestet werden sollte.

Was die Sache nicht einfacher macht: Betroffen ist ein sehr breites Spektrum unterschiedlichster Softwaretechnologien. Es müssen sowohl Legacy-Systeme als auch völlig neue, sehr innovative Anwendungen getestet werden. Das ist insbesondere bei Transformations- und Migrationsprojekten ein großes Thema, wie bei der Umstellung auf SAP S/4HANA oder der aktuellen Entwicklung, IT-Systeme in die Cloud zu transferieren.

Trotz all dieser Aufgaben müssen die Tester ihre Resultate immer schneller liefern?

Absolut, ja!! Die Zeiten, in denen man uns mehrere Monate für den Durchlauf der notwendigen Testzyklen gewährt hat, sind längst passé. Dank Testautomation kommen wir mittlerweile schon mit Tagen statt Monaten für den Durchlauf aller Testzyklen aus. Doch das reicht in Zukunft nicht mehr aus. Wir brauchen schnellere, intelligentere Tests quasi in Echtzeit – und wir brauchen proaktives Testen von Einsatzszenarien, an die der Entwickler noch gar nicht gedacht hat. Zusätzlich müssen wir in der Lage sein, schnellstmöglich eine Teststrategie für diese zu entwickeln.

Software hat sich in allen Branchen zur disruptiven Größe entwickelt und besitzt das Potenzial, Geschäftsstrategien entscheidend zu beeinflussen. Dieses Potenzial gilt es zu nutzen. Damit Unternehmen aber wirklich nachhaltig intelligenter und weniger störanfällig werden, müssen sie bei Design, Entwicklung und Test ihrer Software völlig neue Wege gehen.

Angesichts der wachsenden Verbreitung von IoT- und KI-Systemen wächst die Bedeutung des Testens weiterhin rasant. Welche Art von Tools, Techniken und Best Practices brauchen wir in Zukunft?

Das Testen von KI-Systemen ist ein sich entwickelnder Bereich. Ein sehr relevantes Thema ist beispielsweise, wie man auf unbewusste menschliche Vorurteile testet, die sich in diese Systeme einschleichen. Wir haben unsere Testplattform für KI-Systeme, „Teach & Test", die in solchen Szenarien hilfreich ist. Das IoT-Feld ist ähnlich umfassend. Wir müssen die Geräte auf die Genauigkeit der Aktionen, Signale und Daten testen, die zum Zeitpunkt der Installation erzeugt werden, und dann kontinuierlich überprüfen, ob das Gerät weiterhin einwandfrei funktioniert. Und diese Tests müssen wir zum Beispiel bei Geräten durchführen, die physisch nicht erreicht werden können, da die Anwendung von IoT-Geräten von der Kaffeemaschine zu Hause bis hin zu einem Kernreaktor reicht. Ich habe keine persönliche Erfahrung mit IoT-Tests in einem so breiten Szenario, aber ich erlebe, dass Testsysteme zur Simulation von Geräten und intelligente Produkttests sehr gefragt sind und immer mehr zunehmen. Das Gebiet ist noch sehr neu und entwickelt sich schnell weiter.

Was sind also Ihre Ratschläge und Vorschläge für die Softwaretester, Herr Jain?

Wir brauchen nicht nur mehr Tests, wir brauchen bessere und intelligentere Tests, wir brauchen ein übergreifendes Quality-Engineering! Und ein Verständnis, dass wir mehr als die letzte Verteidigungslinie sind, sondern maßgeblicher Bestandteil des gesamten Prozesses.

Die Qualität der Software kann ein Unternehmen über Nacht zum Erfolg oder zur Schließung führen. Stellen Sie sich nur vor: Ein Fehler bei einer Kreditkartentransaktion oder in der Lieferkette kann eine Bestellung verhindern, ein Fehler im Videokonferenzsystem kann ein wichtiges Gespräch verhindern, ein Fehler in der eingebetteten Software eines medizinischen Geräts kann eine lebensrettende Entscheidung verzögern, ein Fehler im Flugsystem kann viele Menschen in Gefahr bringen. Seien Sie daher sehr stolz auf das, was Sie durch die Vermeidung von Softwarefehlern erreichen.

Die Erwartungen an die Softwarequalität werden immer komplexer und schwieriger zu erreichen. Wir müssen daher ständig innovative Wege finden, um diese Erwartungen zu erfüllen. Ich kann heute nicht sagen, wie wir das machen werden. Nur eines: Es gibt keine Wahl. Wir müssen erfolgreich sein!

Herr Jain, vielen Dank für das Gespräch!

Zur Person

Bei Accenture ist **Pankaj Rai Jain** seit 2004 Geschäftsführer. Außerdem ist er aktuell Leiter des Bereiches Quality-Engineering in Europa. Der erfahrene IT-Experte unterstützt Kunden bei Entwicklung und Umsetzung ihrer IT-Strategien & Globalen IT-Programme. Eines seiner Ziele ist es, für die Kunden mehr Effizienz und Agilität über die reine Automatisierung von Prozessen hinaus zu erreichen. Ihm geht es aber auch darum, die Qualitätskosten zu reduzieren, die Einführung neuer Produkte zu beschleunigen und eine möglichst hohe Qualität der IT-Services auch in komplexen digitalen IT-Landschaften sicherzustellen.
Accenture ist ein weltweit tätiges Beratungsunternehmen, führend in Digitalisierung, Cloud und Security. 506.000 Mitarbeitende bringen ihre umfassende Erfahrung und spezialisierten Fähigkeiten in mehr als 40 Branchen und in über 120 Ländern ein. Sie bieten Dienstleistungen aus den Bereichen Strategy & Consulting, Interactive, Technology und Operations – gestützt auf das weltweit größte Netzwerk aus Centern für Advanced Technology und Intelligent Operations.

»Mehr Durchblick im Konzernprogramm durch Dashboards«

In diesem Artikel berichten wir von den Erfahrungen, die wir in den letzten drei Jahren in den Bereichen Business Intelligence und Lean Quality Management im Umfeld agiler Softwareentwicklung gewonnen haben. Wir stellen vor, was aus unserer Sicht wirklich erfolgreich ist und bei welchen Themen wir kontinuierlich nach besseren Lösungen suchen.

Der „Hype" um Lean- und Agile-Methodik in der Softwareentwicklung ist seit Jahren ungebrochen und wird von Fachbereichsseite und Management, in vielen Unternehmen, zum Teil mit überzogenen Erwartungen und zum Teil unter kritischer Beobachtung begleitet. Es wird oft verkannt, dass in agilen Vorgehensmodellen Mechanismen genutzt werden, die aus dem klassischen Qualitätsmanagement abgeleitet sind. Bei genauer Betrachtung ist zu erkennen, dass der Deming-Zyklus [Wiki] aus Plan – Do – Check – Act (PDCA) bereits in agilen Methoden integriert ist.

Als Lean Quality Management unterstützen wir die Entwicklungsteams, indem wir die Bereitstellung von Kennzahlen und Auswertungen automatisieren und somit die notwendige Transparenz für die Weiterentwicklung und Optimierung unserer Prozesse herstellen.

Weshalb überhaupt messen und nachverfolgen – passt das zu Agile?

Die Komplexität hinter Kennzahlen muss zielgruppengerecht aufgeschlüsselt, vermittelt und visualisiert werden.

Im Zuge eines der größten IT-Konzernprogramme der Deutschen Bahn [DB17] wurde eine Dashboard-Lösung umgesetzt, welche sich abhebt vom bisherigen klassischen Reporting und uns Kennzahlen und Visualisierungen auf Agile-Enterprise-Ebene liefert. Es war eine große Herausforderung, die richtige Balance zwischen einem übergreifenden Regelwerk und förderlichen Freiheitsgraden von über 30 Agile-Teams zu finden.

Agile Forecasting mit Dashboard-Lösungen

Das Management will auf der einen Seite, dass agile Methodik umgesetzt und gelebt wird – auf der anderen Seite längerfristige Schätzungen, Meilensteine und Verbindlichkeit. Kurzfristig betrachtet bieten agile Vorgehensmodelle wie Scrum zum Beispiel Burndown Charts als Lösung an – wie sieht es jedoch mit den genannten längerfristigen Zielen und Schätzungen aus?

Wir glauben, eine innovative Lösung für diese Herausforderung gefunden zu haben, welche sich nun seit Jahren bewährt hat. Die gesamten Entwicklungsschritte sind feingranular auf verschiedenen Detailstufen in einem Ticket-System abgebildet. Angefangen bei der Portfolioebene hin zu Entwickleraufgaben. In den höheren Ebenen werden Aufgaben nicht geschätzt, sondern nach *WSJF* (Weighted shortest Job first) priorisiert. In den Ebenen darunter können wir auf Quartalsebene unsere Programminkremente planen. Dieses Vorgehen ist im Rahmen von *SAFe* (Scaled Agile Framework) bereits länger bekannt.

Mithilfe des Business Intelligence-Tools Qlik Sense [Qlik] haben wir eine Lösung implementiert, welche uns ein Forecasting auf Quartalsebene ermöglicht – mit einer Abweichung von rund 10 Prozent pro PI (Program Increment). Diese Lösung ermöglicht es, faktenbasiert und nachvollziehbar die Entwicklung zu verfolgen.

Die Kennzahlen beispielsweise für die Anzahl der Story Points werden automatisiert aus den Quellsystemen geladen und können in verschiedenen Aggregationsstufen dargestellt werden, zum Beispiel die Gesamtsicht oder pro Team und Sprint. Sämtliche Bereiche des SDCL (Software Development Life Cycle) sind kennzahlentechnisch abgebildet.

Agile Forecasting kann nur dann gut funktionieren, wenn eine gewisse Menge an Erfahrung bezüglich der Leistungs- und Schätzfähigkeit in den Teams vorhanden ist und eine ausreichende Anzahl von Datensätzen über

die geleistete Arbeit vorliegt. Ein weiterer Aspekt ist das Umfeld und die Art der Softwareentwicklung. Neue Softwareprodukte, welche maximale Agilität benötigen, da man zeitnah auf Ereignisse am Markt oder Feedback reagieren will, werden weniger von einer langfristigen Sicht profitieren können. Je dynamischer die erwartete Zukunft, desto ungenauer wird diese.

Klassisches Reporting fand in regelmäßiger Taktung statt, das Agile Reporting ermöglicht durch die Automatisierung jederzeit Zugriff auf die aktuellen Zahlen. Ebenfalls können die Informationen im Dashboard durch die Nutzer individuell auf ihren Informationsbedarf angepasst werden – statt sich mühsam durch Reports mit bis zu 100 Seiten durchzuarbeiten. Durch eine Datenindexierung können Beziehungen zwischen Kennzahlen über verschiedene Dimensionen verstanden werden und neue Erkenntnisse fördern, die auch durch Machine Learning-Algorithmen unterstützt werden. Bisherige Reports oder Dashboards waren eine Ansammlung von statischen Daten und Visualisierungen.

Im Rahmen des Konzernprogramms wird eine Architekturerneuerung für die bestehende Vertriebsplattform durchgeführt. Daraus ergibt sich, dass *ein Teil* des Scope fix und im Backlog hinterlegt ist. Die zugehörigen Epics sind bereits *grob* geschätzt. So kann unter anderem auf Basis der Velocity der letzten Programminkremente rollierend ein Forecast erstellt werden, wann die Teams diesen Scope bearbeitet haben. Ziel ist es, dass der errechnete Termin vor dem voraussichtlichen Programmende liegt und noch einen Puffer für die Integration von notwendigen fachlichen Weiterentwicklungen bietet. Als Ableitung aus diesem Forecast können Rückschlüsse über das notwendige Sizing der einzelnen Entwicklungsteams gezogen werden. Dies ermöglicht eine Diskussion mit den zuständigen Scrum Mastern und gegebenenfalls eine Anpassung der Planung.

Klassische Projektkennzahlen spielen weiterhin eine wichtige Rolle

Das Konzernprogramm läuft über mehrere Jahre und beschäftigt viele verschiedene Teams und Stakeholder. Entsprechend sind auch die klassischen Projektkennzahlen zu Scope, Budget, Time, Quality und Risk abgebildet. Diese werden nicht mehr als Statusreport verteilt, sondern sind in einem Dashboard transparent einsehbar und individuell auswertbar. Ein großer Vorteil dieser Lösung ist die Interaktion mit den Daten beziehungsweise Kennzahlen. Durch intelligente Filter- und Suchmöglichkeiten können neue Erkenntnisse von den Nutzern selbst erlangt werden.

Auf dem Weg zur Definition eines einheitlichen Leistungsbegriffs hatten wir fruchtbare Diskussionen in den Teams und mit der Programmleitung. Diese waren wichtig, um ein gemeinsames Verständnis herzustellen, und haben das Interesse, die Leistung feingranular aufzuschlüsseln, gezeigt. Die Leistung der Teams messen wir nun nicht nur an der reinen Scope-Bearbeitung nach Plan, sondern erfassen auch weitere Leistungsfaktoren wie die Arbeit an nichtfunktionalen Anforderungen und Innovationsthemen sowie die Korrektur von Defects.

Was waren die wichtigsten Erkenntnisse auf technischer Ebene?

Die Auswertung von historischen Daten in Bestandssystemen ist hochkomplex und muss gut geplant werden. Teilweise werden sehr große Datenmengen verarbeitet, welche die Schnittstellen von angebundenen Systemen schnell an ihre Grenzen bringen können. Entsprechend wichtig ist die Einbindung der jeweiligen Fachabteilungen sowie das Abstimmen von Vorgaben und angepassten Schnittstellen. Darüber hinaus sind auch die Anforderungen des Datenschutzes zu beachten, sodass die Auswertung von Daten einzelner Mitarbeiter in den Dashboards ausgeschlossen ist. Um dies zu gewährleisten, sollten diese sensiblen Daten erst gar nicht exportiert werden.

Je nach Beschaffenheit der Systeme und des Umfelds stellt sich oft die Frage: Data Warehouse oder Data Lake? Wir haben einen Hybrid-Weg gewählt und nutzen beide Technologien im Parallelbetrieb. So können wir von den Vorteilen aus beiden Welten profitieren **(siehe Abbildung 1)**.

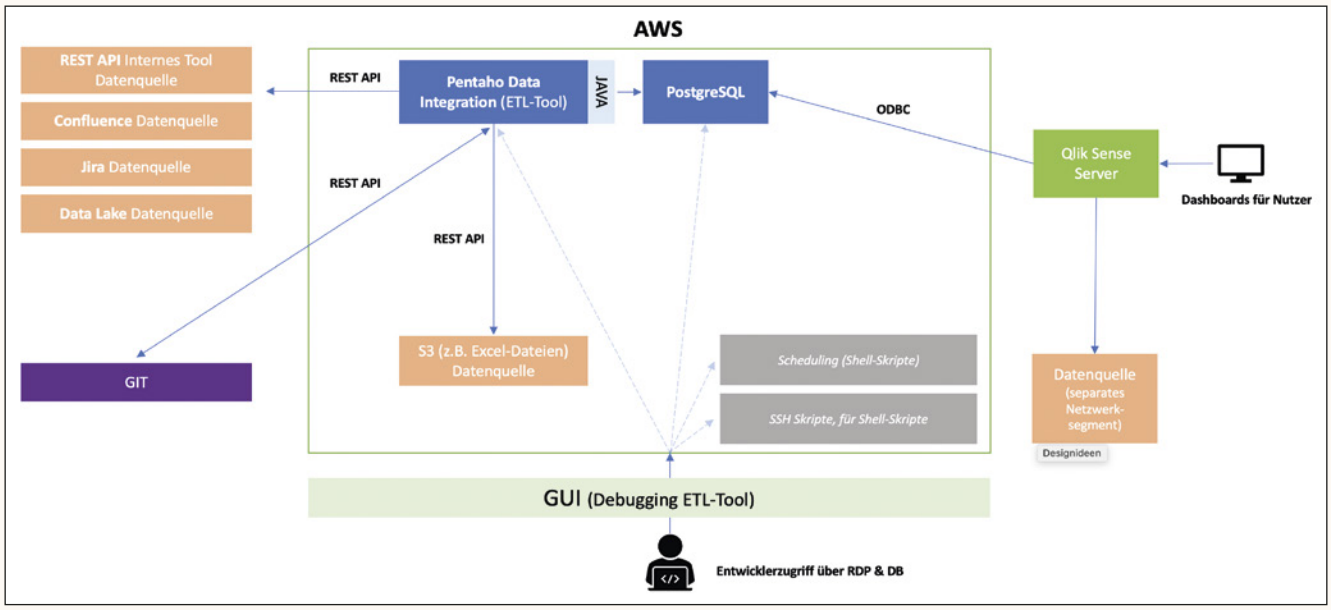


Abb. 1: Systemlandschaft ETL-Strecke (Data Lake als Datenquelle)

Im Verlauf des Projekts haben wir die Erkenntnis gewonnen, dass es im Rahmen des Reportings wenig Gründe gibt, *vollständig* auf einen Data Lake zu setzen, sofern man nicht *deutlich* mehr Datenquellen (und Daten) verarbeiten muss. Besonders interessant wird der Einsatz erst dann, wenn Daten aus verschiedenen Systemen parallel durchsucht und verarbeitet werden müssen, beispielsweise für die dynamische Erstellung von Dashboards. Ein weiterer Punkt ist der erhöhte Aufwand, die Daten in einer heterogenen Systemumgebung mit spezifischen Anforderungen an Sicherheit und Datenschutz zusammenzuführen.

Anfänglich war die Systemperformanz kein Thema. Jedoch sollten die Anforderungen und Kapazitäten der Datenquellen beachtet werden. Gegebenenfalls entstehen Mehrkosten, wenn diese ausgebaut werden müssen. Ein Refactoring kann dazu beitragen, diese Kosten zu minimieren. Durch Anpassungen der Abfragen konnten wir die Ladezeit verkürzen und die Last auf die Datenquellen oder APIs *deutlich* reduzieren.

Welches waren die wichtigsten Erkenntnisse auf fachlicher Ebene?

Eine vergleichende Performanzmessung zwischen verschiedenen Agile Teams ist nicht zielführend und ohne Rahmenbedingungen wie der Normalisierung von Story Points auch nicht sinnvoll möglich. Eine Alternative bietet die Orientierung am geschaffenen Geschäftswert, jedoch muss zwingend die unterschiedliche Komplexität von Technologien beachtet werden – ansonsten vergleicht man Äpfel mit Birnen. Je schneller funktionierende Software beim Endkunden landet, desto schneller kann man geleisteten Entwicklungsaufwand ins Verhältnis zum geschaffenen Geschäftswert stellen!

Die Nivellierung von geleisteten Story Points über mehrere Programminkremente, respektive generell Zeitintervalle, ist eine Methode, welche in klassischen Wasserfall-Projekten (mit Personentagen) Orientierung bieten kann, jedoch in einer agilen Umgebung und einem dynamischen Markt mit stetig neuen Herausforderungen nicht angewendet werden sollte, um Aussagen zur Performanz zu treffen. Davon ausgenommen ist die Verwendung solcher Daten für Prognosen und Schätzungen – mit entsprechender Unschärfe. Letztere benötigen entsprechend eine saubere Datenbasis – je weniger strukturiert und konsistent die Datenbasis gepflegt wurde, desto höher die Aufwände für die Datenanalyse und spätere Extraktions- und Transformationsprozesse auf technischer Ebene.

Diverse Faktoren müssen berücksichtigt werden, die folgenden drei sind Beispiele:

- Wie hat sich die Velocity in den letzten Monaten entwickelt? Wir haben einen Algorithmus entwickelt, welcher neben der Anzahl der geleisteten Story Points beziehungsweise der Velocity auch statistische Methoden zur Korrektur nutzt. Ein Beispiel ist die Korrektur der Normalabweichung.
- Wie ist das Verhältnis von Mitarbeiteranzahl zur Velocity? Werden Feiertage oder Innovations-Sprints berücksichtigt?
- Sollte eine Normierung von Story Points, im Zuge einer besseren Vergleichbarkeit, durchgeführt werden?

Ein weiterer essenzieller Aspekt bei der Betrachtung von Entwicklungsteams ist eine Reihe von Kennzahlen, welche sich in der *Praxis* bewährt haben:

- Deployment-Frequenz,
- Lead-Time für Anpassungen/Änderungen,
- Rückrollgeschwindigkeit von Releases und Fixes (MTTR),
- Verhältnis von Änderungen zu Fehlern (Change Failure Rate).

Um die Motivation der Entwicklungsteams längerfristig zu gewährleisten, ist die Empfehlung, sich nicht ausschließlich auf quantitativ zu erhebenden Maßeinheiten zu fokussieren, sondern den Teams abstrakte, aber klare Ziele vorzugeben – die Lösungen sollten von den Teams kommen. Eine Empfehlung wäre hier die Nutzung von *Objektive Key Results* mit den genannten Kennzahlen.

Fazit

Es ist essenziell, bestehende Kennzahlen, Prozesse und Ergebnisartefakte immer wieder strukturiert zu hinterfragen. Das Mindset steht hier im Vordergrund. Entsprechend zielen die Maßnahmen hinter unserer PDCA-Implementierung nicht auf Kontrolle ab, sondern basieren einerseits auf Vermittlung von Konzepten zur Verbesserung der Leistungsfähigkeit, andererseits auf Transparenz der eigentlichen Leistung. Diese Maßnahmen betrachten jedoch immer die Gesamtleistung. Im Fokus stehen Qualität und Lieferversprechen.

Referenzen

- [DB17] https://digitalspirit.dbsystel.de/so-wird-der-vertrieb-fit-fuer-die-zukunft-gemacht
- [Qlik] https://www.qlik.com/de-de
- [Wiki] https://de.wikipedia.org/wiki/Demingkreis

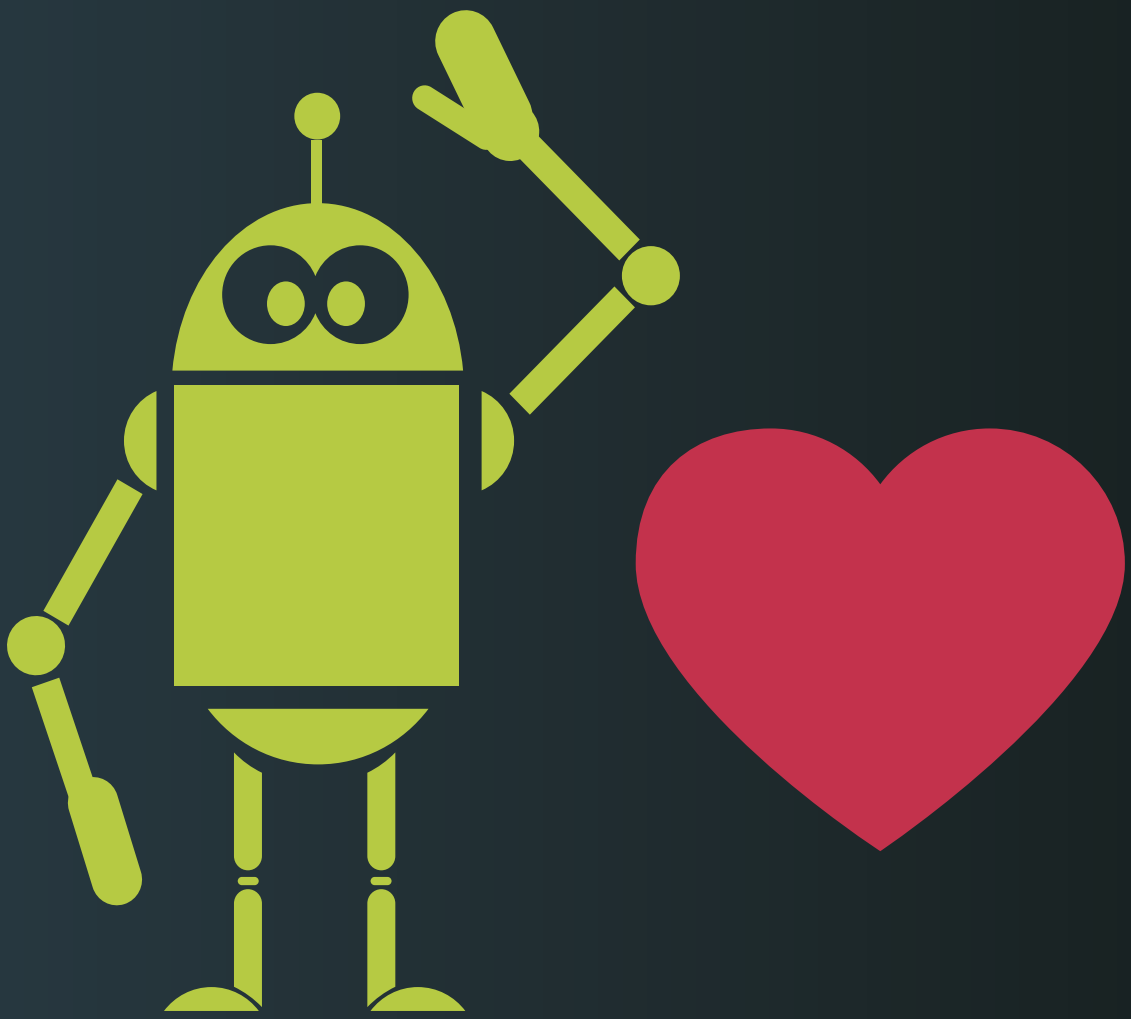
Daniel Lozynski

daniel.lozynski@deutschebahn.com
ist seit über 10 Jahren im IT-Umfeld tätig. Vom Dienstleister, Start-up bis hin zum Konzern ist alles dabei. Aktuell verantwortet er das Reporting für die Vertriebsplattform bei der DB Vertrieb GmbH und legt einen starken Fokus auf Innovations- und Strategiethemen.

Enrico Wenzel

enrico.wenzel@deutschebahn.com
ist seit über 20 Jahren im IT-Umfeld tätig und interessiert sich für alle Themen rund um Softwarequalität. Seit 2017 verantwortet er das Qualitätsmanagement für die Vertriebsplattform bei der DB Vertrieb GmbH.

servicetrace
SOFTWARE ROBOTICS



Software Testing

10x schneller

100% fehlerfrei

24/7 verfügbar

Servicetrace Software Robotic Solutions für Test Automation.
Schneller, besser, mehr testen.

www.servicetrace.de

»Fokussierung, Automatisierung, Expansion«

In einer Zeit, in der die Beschleunigung der Entwicklungs- und Release-Zyklen für viele Bereiche des Business zu einem entscheidenden Wettbewerbsfaktor geworden ist, wird das altbekannte Dilemma zwischen Geschwindigkeit und Qualität weiter auf die Spitze getrieben. Mit ganzheitlichen Herangehensweisen zur Umsetzung von Continuous Delivery hat sich hier im Softwareentwicklungsprozess im vergangenen Jahrzehnt mit der Nutzung von DevOps-Ansätzen bereits einiges getan.

Es stellt sich aber immer auch die Frage, wie die Qualität nicht nur eingebaut, sondern auch in so schnellen Zyklen überprüft werden kann. Das Quality Management muss also ebenfalls Lean daherkommen, damit es den Anforderungen von kurzen Durchlaufzeiten in modernen Softwareentwicklungsprozessen genügt.

In unseren Projekten haben wir im Wesentlichen drei Stoßrichtungen identifiziert, deren Bearbeitung zu einer erfolgreichen Umsetzung eines Lean Quality Management führen kann:

- *Fokussierung:* Das Quality Management muss sich auf die wichtigen Dinge fokussieren und von der Vollständigkeitsdenke wegkommen.
- *Automatisierung:* Automatisierung muss weiter vorangetrieben werden. Das wissen alle, aber nur wenige machen das auch konsequent.
- *Expansion:* Quality Management darf sich nicht nur auf das klassische Überprüfen der Umsetzung von Anforderungen durch Testing beschränken, sondern muss näher an den Nutzer des Produkts heranrücken.

Im Folgenden werden die Potenziale dieser drei Stoßrichtungen jeweils anhand eines konkreten Beispiels aufgezeigt.

Fokussierung

In vielen Beratungssituationen findet man eine völlig überdimensionierte Suite an Testfällen vor, deren Abarbeitung sich in keiner Weise mit dem Ziel kurzer Release-Zyklen vereinbaren lässt. Dadurch bleibt es in vielen Fällen bei halbjährlichen oder quartalsweisen Release-Zyklen, obwohl die bereits agilisierten Entwicklungszyklen in hoher Frequenz PSIs (Potentially Shippable Increments) abliefern. Nicht nur im Falle von manuellem Testing, sondern auch bei automatisierten Regressionstest-Suites begegnet man dieser Problematik sehr häufig.

Ein wichtiger Ansatz für die Reduktion dieser großen Testfall-Suites ist es, diese von Redundanzen zu befreien. In der Regel wird man darüber zwar die Anzahl der Testfälle verkleinern können, der große Wurf kann damit aber nicht gelingen. Es sind daher radikalere Einschnitte in die Testfall-Suite über eine risikobasierte Betrachtung von Nöten. Hier ist Kreativität, Verständnis für den Kunden und dessen Business sowie eine gute Kenntnis des zu testenden Systems erforderlich.

In einem Projekt bei einer großen Rückversicherung, deren zentrales Geschäftssystem naturgemäß einer regelmäßigen Wartung und Weiterentwicklung unterliegt, haben wir bei der Reduktion der Testfall-Suite mit der Strategie gute Erfahrungen gemacht, in den hochfrequenten Regressionstests diejenigen Testfälle nach und nach zu eliminieren, die regelmäßig keine Fehler finden.

In einem ersten Schritt wurden hierbei die Testreports eingesammelt und diese auf Anzahl von Testausführungen und Anzahl entdeckter Defects pro Functional Area hin ausgewertet. Nach Eliminierung der False Positives wurden die Functional Areas identifiziert, für welche die Testfälle Fehler gefunden haben. Bei den unmittelbar folgenden Testzyklen hat das Testteam dann auf die Durchführung der erfolglosen Teil-Suiten verzichtet und diese bezüglich Effektivität auf den Prüfstand gestellt **(siehe Abbildung 1)**.

Mit dieser Strategie war das Testteam in der Lage, die folgenden Potenziale zu erschließen:

- Verkleinerung der Durchlaufzeiten des Regressionstests,
- Eliminierung der Wartungs- und Betriebskosten für ineffektive Testfallszenarien,
- Erhöhung der Kapazitäten für die Entwicklung effektiverer Testfallszenarien.

Neben dieser eher businessorientierten Vorgehensweise zur Fokussierung gibt es auch vielversprechende Ansätze, die die Entwickler-Seite beziehungsweise den von den Entwicklern erstellten Code in die Betrachtung einbeziehen.

Durch Einsatz einschlägiger Tools lässt sich über Instrumentierung des Quellcodes identifizieren, welche Teile des Codes seit dem letzten Test verändert oder ergänzt, aber noch nicht getestet wurden, sodass sich die Aufmerksamkeit des Tests genau auf diese Lücken lenken lässt („Test Gap Analysis"). Dadurch eröffnet sich die Möglichkeit, immer nur sehr kleine Anteile umfangreicher Regressionstest-Suiten laufen zu lassen und damit trotzdem den größten Teil der Risiken abzudecken („Test Impact Analysis").

Automatisierung

In den meisten IT-Organisationen unternehmen die Entscheidungsträger Anstrengungen, für erfolgskritische Anwendungen in das Continuous Delivery Regime zu kommen. Dafür werden mit unterschiedlichem Erfolg neben vielen anderen Maßnahmen Automatisierungslösungen für die Unterstützung des gesamten Software Development Life Cycle eingeführt. Dies ist überwiegend bereits der Fall für Versionierung, Umgebungsaufbau und Deployment, was ohne Frage eine wesentliche Voraussetzung für kurze Durchlaufzeiten von neuen Softwareversionen darstellt.

Wie der World Quality Report 2018/2019 ([WQR], S. 26: „Test automation – The single-biggest enabler of maturity in QA and testing") deutlich macht, gibt es aber grundsätzlich noch deutliches Potenzial, was die Automatisierung im Testing angeht. Nach Einschätzung der Umfrageteilnehmer lag zu diesem Zeitpunkt der Automatisierungsgrad in allen abgefragten Feldern (Functional, API, Performance, Security usw.) durchwegs unter 20 Prozent, während ca. zwei Drittel der Teilnehmer in jeglicher Hinsicht Vorteile bei der Testautomatisierung sehen. Hier liegt momentan fraglos noch ein großes Potenzial brach, welches durch entsprechende Überzeugungsarbeit bei den Entscheidungsträgern, die gegebenenfalls die Anfangsinvestition bei der Testautomatisierung scheuen, gehoben werden kann.

Bei der Einführung einer einheitlichen CRM-Lösung bei einem großen Unternehmen in der Automobilindustrie sind wir gegenüber den klassischen Automatisierungsansätzen noch einen Schritt weitergegangen. Während in der Regel neben der statischen Code-Analyse die *Durchführung* von dynamischen Testfällen im Fokus der Automatisierung steht, wurde in diesem Projektkontext eine Lösung zur automatisierten *Erstellung* von Testfällen aufgesetzt. Dadurch war es möglich, in einem hochagilen Umfeld bereits im Stadium noch nicht gefestigter Anforderungen, in dem ein hoher Automatisierungsgrad aufgrund von häufig erforderlichen Testfallanpassungen nicht anzuraten ist, Aufwände und Durchlaufzeiten im Testprozess niedrig zu halten.

Voraussetzung hierfür ist es, Architektur und Design der Use-Cases toolunterstützt zu modellieren und daraus über eine implementierte Logik Testfälle abzuleiten. Im konkreten Fall hat das Team die Modellierung in MagicDraw aufgesetzt. Für die Verwaltung der Testfälle kommt Jira/Xray ins Spiel. Beide Systeme sind so gekoppelt, dass nach einer Überarbeitung eines Use-Cases in MagicDraw auf Knopfdruck angepasste Testfälle in Jira/Xray generiert werden können.

Die Rundreise **(siehe Abbildung 2)**, detailliertere Ausführungen hierzu unter [Nin20], beginnt mit dem Product Owner (PO), der ein neues Epic in Jira erstellt. Anschließend wird das Epic in das als digitaler Zwilling des Systems fungierende Modell in MagicDraw gezogen, wo der Analyst die Use-Cases iden-

Analyse von Test Reports

Functional Area	Nr of tested releases	Nr of reported Defects
FA 1	30	8
FA 2	10	
FA 3	15	
FA 4	30	
FA 5	15	2
FA 6	5	
FA 7	30	5
FA 8	30	2
FA 9	5	2
FA 10	30	
FA 11	25	3
FA 12	25	3
FA 13	1	
FA 14	1	1
FA 15	1	1
FA 16	7	5

Eliminierung von False Positives

Functional Area	Nr of tested releases	Nr of real Defects
FA 1	30	2
FA 2	10	
FA 3	15	
FA 4	30	
FA 5	15	
FA 6	5	
FA 7	30	
FA 8	30	1
FA 9	5	1
FA 10	30	
FA 11	25	2
FA 12	25	1
FA 13	1	
FA 14	1	1
FA 15	1	1
FA 16	7	

Identifizierung der effektivsten Szenarien

Functional Area	Nr of real Defects	Test Scenario IDs
FA 1	2	S17,S45
FA 2		
FA 3		
FA 4		
FA 5		
FA 6		
FA 7		
FA 8	1	S12
FA 9	1	S87
FA 10		
FA 11	2	S24,S55
FA 12	1	S39
FA 13		
FA 14	1	
FA 15	1	
FA 16		

Abb. 1: Reduktion der Testfall-Suite durch Fokussierung auf effektive Test-Szenarien

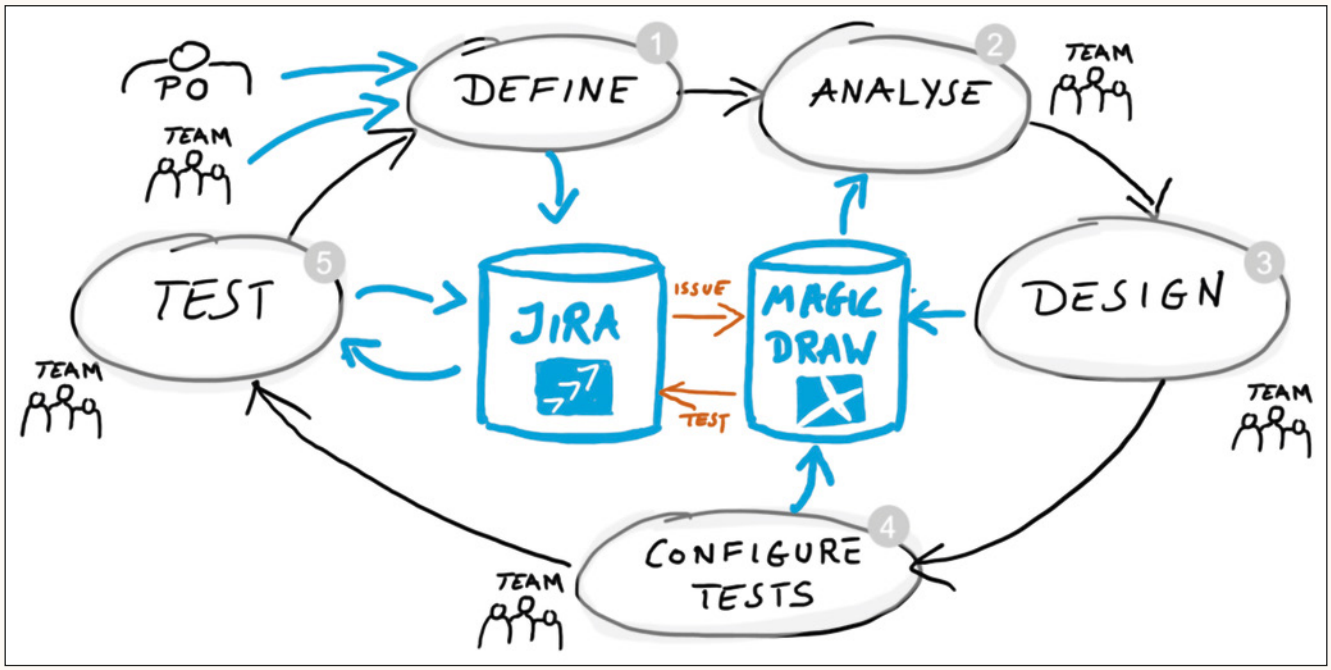


Abb. 2: Rundreise durch den Entwicklungszyklus mit automatisierter Testfallerstellung

tifiziert, die vom Epic betroffen sind, und Abhängigkeiten vermerkt.

Gelangt das Epic in einen Implementierungs-Sprint, prüft der Designer den Ablauf und passt ihn bei Bedarf an. Parallel oder danach prüft der Tester die betroffenen Testkonfigurationen und passt diese ebenfalls bei Bedarf an. Dazu stellt er Wertbelegungen für vordefinierte Testparameter bereit und ordnet Testpfade zu.

Startet man den Testgenerator, dann erzeugt dieser je Kombination aus Testpfad und Testkonfiguration einen instanziierten Testfall. Zudem werden die Testdetails mit Testdaten und erwarteten Ergebnissen pro Testschritt ergänzt. Da generierte Tests automatisch mit ursprünglichen Epics verknüpft werden, lässt sich in Jira auf einfache Weise die Überdeckung von Epics mit Testfällen oder nach Ausführung der Testfälle die Testabdeckung ermitteln. Dadurch kann auch der PO sehr leicht dem Fortschritt folgen. Durch Einsatz dieser Lösung zur Entlastung bei der Testfallpflege konnten die Fachtester ihre Kapazitäten verstärkt anspruchsvollen explorativen Tests widmen.

Automatisierung im Testumfeld ist also nicht nur beschränkt auf das, was man gemeinhin darunter versteht, nämlich die Implementierung einer automatisierten Ausführung von Testfällen. Neben der Ausdehnung der Automatisierung auf die Testfallerstellung sehen wir als weitere Ansatzpunkte in diesem Zusammenhang zum Beispiel den Einsatz von KI zur Automatisierung von Prozessschritten im Defect Management (Automatisches Routing von Defects) oder in der Verwendung von White Box Fuzzing zur Automatisierung von Entwicklertests.

Expansion

Ob ein Softwareprodukt oder ein IT-Service erfolgreich ist, wird letztendlich daran gemessen, wie es/er vom Anwender angenommen wird. Qualität muss daher von Anfang an vom Anwender her gedacht werden. Das bedeutet, dass die unterstellten Nutzenpotenziale und die Erfüllung von Akzeptanzmerkmalen möglichst frühzeitig im Software Development Life Cycle überprüft werden müssen.

In einem kürzlich abgeschlossenen Projekt bei einem globalen Versicherungsunternehmen konnten wir dies erfolgreich mit einem entsprechend adaptierten Schnitt von Teststufen umsetzen. Vor wenigen Jahren hat der Versicherer den erfolgreichen Sprung in die Welt der Lifestyle-Apps gewagt. Es stand fest, dass er es sich nicht leisten kann, den Kunden lange warten zu lassen und anschließend mit einem Produkt zu überraschen, von dem nicht klar ist, ob es beim Kunden positiv ankommt.

Die Herausforderung bestand also darin, einen Weg zu finden, der ausgewählten Kunden einen kontrollierten Zugang zu noch nicht fertigen Produkten gewährt, mit dem Ziel, konstruktives Kundenfeedback zu erhalten, ohne dabei zu viele Informationen an die Konkurrenz preiszugeben oder das positive Image aufgrund von nicht ausgereifter Software zu verlieren.

Zur Lösung dieses Problems hat das Team eine abgesicherte und kontrollierte Spielwiese aufgebaut, zu welcher in zwei Stufen projektfremden Anwendern der Zugang gewährt wurde, nachdem der interne Test durch Entwickler und Tester des Projektteams während der agilen Iterationen abgeschlossen war **(siehe Abbildung 3)**.

In der ersten Stufe unterzogen nach Bestehen der Ausgangskriterien des internen Tests und Deployment des Produkts auf die Spielwiese Mitarbeiter des Unternehmens mit spezifischem Fachwissen das Produkt einem ausgiebigen Test. Im Falle der Umsetzung einer User-Story zur Abrechnungslogik waren dies beispielsweise projektfremde Mitarbeiter aus der Finanzabteilung.

In einer zweiten Stufe durften ausgewählte Kunden testen. Mit Aufnahme in das Pilotprogramm wurde diesen hierzu ein zeitlich begrenzter Zugriff auf die Spielwiese gewährt. Das daraus resultierende Feedback

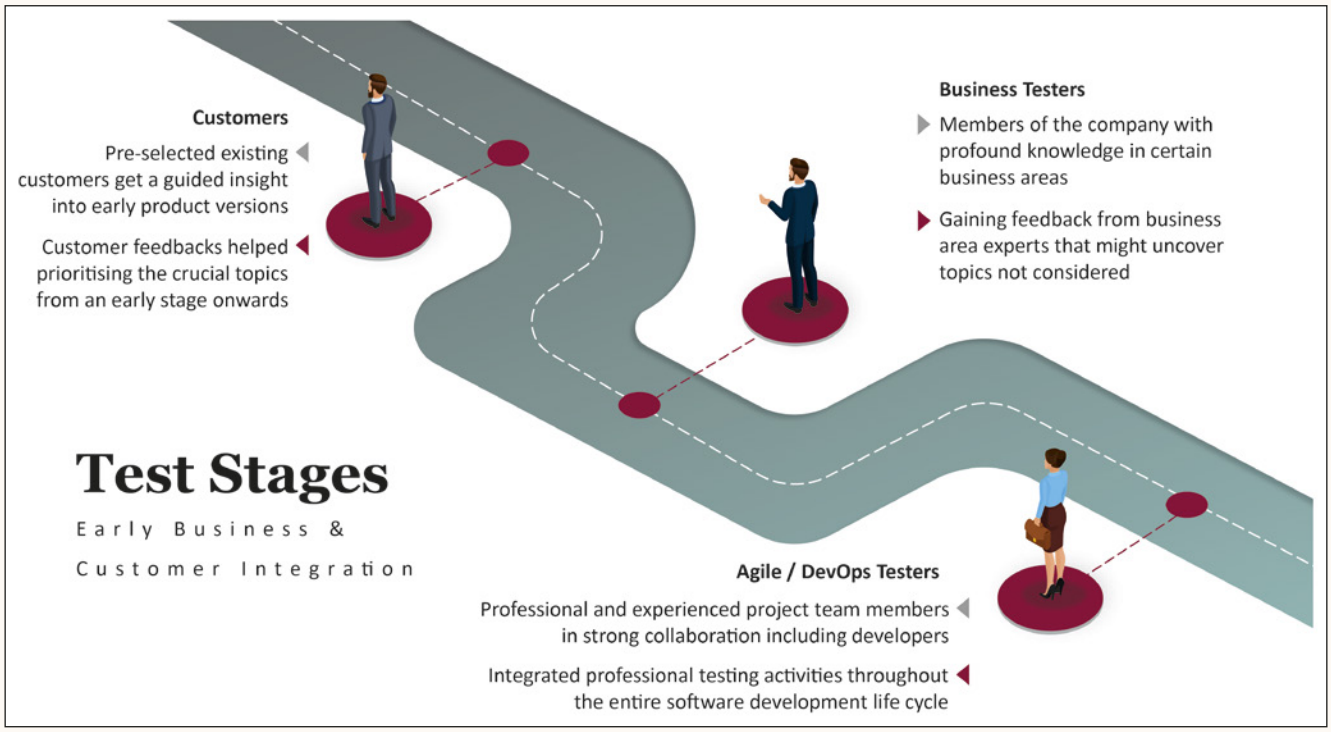


Abb. 3: Teststufen zur frühen Einbeziehung von Business und Kunden in den Test

konnte entwicklungsbegleitend verarbeitet werden, sodass dadurch das Projektteam bei Produktivsetzung zuversichtlich mit einer hohen Nutzerakzeptanz rechnen konnte, weil aufgrund dieser Vorgehensweise

- kritische Issues nicht nur in der App, sondern auch in den Anforderungen entdeckt werden konnten und
- frühzeitig vor Go-Live gezeigt werden konnte, worauf sich Operations und Service vorbereiten müssen.

Der nächste angestrebte Entwicklungsschritt für diese Art des Testings ist die Verwendung von einschlägigen Werkzeugen und Plattformen für die Umsetzung von Crowdtesting. Diese unterstützen die Abwicklung von Tests durch Testkapazitäten, die nicht in eine bestehende Organisation eingebunden sind, besser als klassische Testmanagementwerkzeuge (insbesondere Rekrutierung, Feedback-Verarbeitung, Incentivierung). Darüber hinaus bieten diese die Möglichkeit, nicht erst während des Entwicklungsprozesses, sondern bereits bei der Prüfung von Anforderungen oder noch weitgehender bei der Findung von lohnenswerten neuen Anforderungen das Feedback von Stichproben von (potenziellen) Kunden und Anwendern einzubeziehen.

Fazit

Die richtigen Dinge richtig zu tun, ist auch die Maxime bei der Verschlankung von Test und Quality Management. Zur Effizienzsteigerung Automatisierung, zur Steigerung der Effektivität Fokussierung und Expansion.

Referenzen

- [Nin20] U. Nink, Model-Driven Test Generation with VelociRator, Blog, 31.8.2020, siehe: **https://udonink.de/272/**
- [WQR] Capgemini, Micro Focus, Sogeti: World Quality Report, siehe: **https://www.capgemini.com/research/world-quality-report-2019/**

Dr. Stefan Jobst

stefan.jobst@msg.group

ist Abteilungsleiter im Geschäftsbereich XQT – Test & Quality Management der msg systems ag. Nach Hochschulausbildung und Promotion wechselte er in das Beratungsgeschäft. Für die IT-Tochter einer strategischen Unternehmensberatung verantwortete er eine Dekade lang Projekte für die Entwicklung von Wissensmanagement-Plattformen. In der anschließenden mehrjährigen Arbeit als selbstständiger Berater spezialisierte er sich verstärkt auf Qualitäts- und Testmanagement. Seit vier Jahren leitet er bei der msg systems die Test Consulting Abteilung München.

»Wie integriere ich Qualitätssicherung in die skaliert-agile IT-Systementwicklung«

Wir leben in einer Welt, die zunehmend digitalisiert wird und zu großen Teilen bereits ist. Qualität ist insbesondere bei komplexen, skalierten Systemen stets eine große Herausforderung. Der Einfluss von Lean-Management-Methoden nimmt mit steigender Größe und Komplexität der IT-Systeme stetig zu. „Go-Look-See and adapt for IT" wird nachfolgend als vielversprechender Lösungsansatz für die IT-Systementwicklung vorgestellt.

Der aus dem Lean Management stammende Fokus auf Prozessqualität und Vermeidung von Verschwendung wird in den IT-Kontext überführt. Prozessqualität wird dabei als „Hidden Champion" für die Realisierung von Business Agility identifiziert. Schließlich wird die Bedeutung der Rolle des Quality Coach für die erfolgreiche Implementierung der aufgezeigten Lean-Management-Methoden in der skaliert agilen IT-Systementwicklung herausgearbeitet.

Mik Kersten formuliert die Auswirkungen der steigenden Bedeutung von Software und deren Qualität im Geschäftsleben frei übersetzt wie folgt: Diejenigen Unternehmen, welche es schaffen, große skalierte Softwareprojekte erfolgreich auszuliefern, geben den Takt in der Wirtschaftswelt des 21. Jahrhunderts vor [Ker19]. Mit Blick auf die Börsenwerte und den Einfluss der großen Tec-Giganten wie Google, Microsoft, Apple und Co. scheint sich diese These zu bestätigen.

Agile Vorgehensweisen wie Scrum helfen, viele Aspekte der oben genannten Herausforderungen zumindest teilweise für kleinere IT-Applikationen zu lösen. Doch für größere IT-Verfahren mit mehreren Hundert Mitarbeitern wie bei ERP-Systemen ist dies nicht mehr ausreichend. In diesem Umfeld wird meist mit agilen Skalierungsframeworks wie zum Beispiel SAFe© [Chr17], NEXUS© [Bit18] oder LeSS© [Lar17] gearbeitet, um die Komplexität und Größe beherrschbarer zu machen. Die meisten dieser Skalierungsframeworks setzen auf bewährte Methoden aus dem Lean Management, die mal mehr mal weniger an die spezifischen Besonderheiten der Wissensarbeiter in der Softwareentwicklung angepasst sind.

Diese starke Tendenz zur Anwendung von Lean-Management-Methoden und Prinzipien ist auch auf das allgegenwärtige Ziel, „Business Agility" tatsächlich umzusetzen, ausgelegt. Business Agility und damit verbunden die Reduzierung der Time to Market wird von 56 Prozent der befragten Unternehmen im „Status Quo (Scaled) Agile 2020"-Report der Hochschule Koblenz [Kom20] als Hauptgrund für die Einführung agiler Methoden angegeben. Die Realisierung einer kürzen Time to Market kann dabei auf unterschiedliche Weisen erreicht werden.

Lean-agiles Qualitätsmanagement ist einer der vielversprechendsten und gleichzeitig aktuell noch einer der am wenigsten eingesetzten Wege, um dies zu erreichen. Denn Business Agility ist fundamental von der Fähigkeit abhängig, schnell qualitativ hochwertig zu liefern, schnell auf Änderungen reagieren zu können und dabei stets kundenzentriert zu agieren.

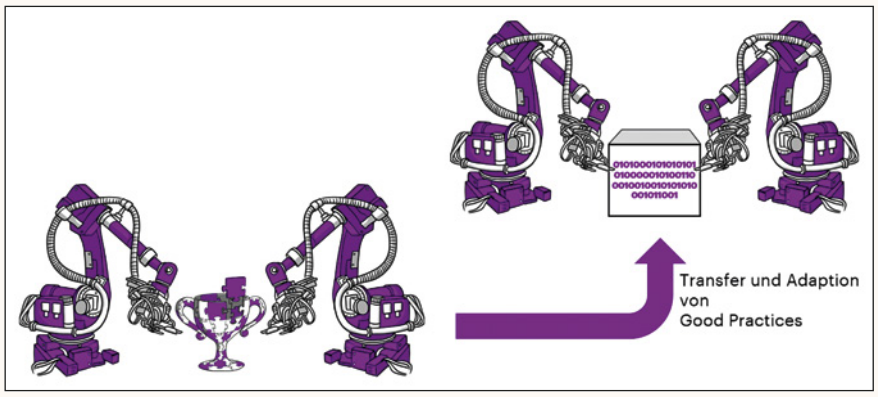


Abb. 1: Transfer und Adaption von Good Practices aus der Fertigungsindustrie in die Softwareentwicklung

Ein agiler Lösungsansatz – Lernen von den Lean-Meistern: „Go-Look-See and adapt for IT"

Im lean-agilen Qualitätsmanagement lassen sich durch die Kombination von agilen, Lean und Six Sigma Methoden herausragende Ergebnisse erzielen. Dabei können wir in der Softwareentwicklung, wie in **Abbildung 1** dargestellt, durch den Transfer und die Adaption von Good Practices aus der Fertigungs- und Automobilindustrie extrem viel lernen. Das Spektrum reicht von einer kundenzentrierten Arbeitsweise auf Basis von Value Streams, über die Visualisierung und die kontinuierliche Verbesserung bis hin zu einem dedizierten Prozessfokus und der Vermeidung von Verschwendung.

Nachfolgend liegt der Fokus auf den Themenbereichen Produkt- und Prozessqualität und der Vermeidung von Verschwendung. Es wird aufgezeigt, welche Praktiken und Methoden der Fertigungsindustrie in den Kontext der Softwareentwicklung überführt werden können. Denn ein einfaches Kopieren ist aufgrund der großen Unterschiede zwischen Massenfertigung am Fließband und kreativer Wissensarbeit an der CI/CD-Pipeline nicht sinnvoll.

Qualitätssicherung (QS) – Das Zusammenspiel von Produkt- und Prozessqualität

In der IT bezieht sich der Begriff Qualität typischerweise darauf, wie gut das entwickelte Softwareprodukt die Kundenanforderungen erfüllt. QS-Maßnahmen wie funktionale und nicht-funktionale Tests zielen darauf ab, die

Umsetzung der Anforderungen und damit die Qualität des Produkts zu validieren. Dabei werden nicht nur die einzelnen Softwarebestandteile, sondern auch ihre Integration zu einem Softwareprodukt betrachtet.

Die *Produktqualität*, wie sie oben beschrieben ist, ist allerdings nur eine Seite der Medaille. Ein Fokus rein auf Produktqualität reicht nicht aus, um die Kunden auf Dauer zufriedenzustellen. Erst wenn die zweite Seite der Medaille, die Prozessqualität, ebenfalls berücksichtigt wird, erschließt sich das volle Potenzial des Qualitätsmanagements im leanagilen Kontext.

Prozessqualität betrachtet, wie gut die gelebten Prozesse zum Erreichen der Unternehmensziele beitragen. Wenn die Qualität der Prozesse gut ist, führen sie stets zu demselben, angestrebten Ziel – sie treffen immer ins Schwarze. Eine unzureichende Prozessqualität wird hingegen sichtbar in Schwankungen der Ergebnisse. Zwei Faktoren spielen hier eine wichtige Rolle: die Präzision eines Prozesses und seine Genauigkeit. Ein genauer Prozess erreicht zwar immer grob sein Ziel, allerdings nicht reproduzierbar mit demselben Ergebnis. Ein präziser Prozess erreicht zwar immer dasselbe Ergebnis, jedoch nicht unbedingt das Gewünschte. Erst wenn ein Prozess sowohl präzise als auch genau ist, also eine Wiederholgenauigkeit gegeben ist, ist die Prozessqualität akzeptabel und die Softwareentwicklung ein Erfolg.

Erst im Zusammenspiel wird der Wert von Produkt- und Prozessqualität deutlich. Um in einem lean-agilen Kontext schnell hoch-qualitative Softwareprodukte liefern zu können, müssen beide Qualitätsaspekte berücksichtigt und in einer holistischen QS-Vorgehensweise zusammengefasst werden. Im Folgenden wird anhand eines Beispiels näher erläutert, wie das Thema Prozessqualität konkret im IT-Kontext zur Anwendung kommen kann.

Der Kreis der Verschwendungen – ein Anwendungsbeispiel

Um die Prozessqualität zu verbessern, können Methoden des Lean Managements angewandt werden, beispielsweise die Verschwendungsanalyse. Verschwendung im Kontext von Lean bezeichnet nicht-wertschöpfende Tätigkeiten, also Tätigkeiten, die nicht direkt dazu beitragen, dem Kunden einen Mehrwert zu liefern. Diese Tätigkeiten sollten identifiziert und dann eliminiert werden. Dies hat signifikanten Auswirkungen auf die Durchflusseffizienz und fördert das Ziel, die „Time to Market" zu reduzieren. Gemäß einer Studie von Poppendieck [Pop03] beträgt die Durchflusseffizienz über verschiedene Industrien hinweg nur 15 Prozent, was wiederum bedeutet, dass 85 Prozent der Arbeit im Softwarelebenszyklus als Verschwendung angesehen werden kann.

Im Kreis der Verschwendungen, der in **Abbildung 2** skizziert ist, werden verschiedene Arten von Verschwendungen kategorisiert. Diese lassen sich auch auf die Softwareentwicklung übertragen, entweder wie von Mary Poppendieck vorgeschlagen auf IT-System-Ebene oder auf Ebene der IT-Organisation. Wie lässt sich Letzteres nun konkret in IT-Organisationen anwenden? Um dies zu beantworten, wird im Folgenden die Verschwendungsanalyse am Beispiel der Testautomatisierung vorgestellt.

① *Überproduktion:* Überproduktion wird auch die Mutter aller Verschwendungen genannt. Im Kontext der Testautomatisierung fallen beispielsweise automatisierte Testfälle, die nur einmal durchgeführt werden, unter diese Verschwendungsart.

② *Bestände:* Als Konsequenz aus der Überproduktion ergeben sich immer größere Bestände, zum Beispiel an automatisierten Testfällen, die zu fehlender Übersicht und größeren Wartungsaufwänden führen, oder auch zu längeren Durchlaufzeiten, weil die großen Bestände das System langsamer machen.

③ *Bewegung:* Diese Verschwendungsart bezieht sich auf die Software-Ingenieure, die

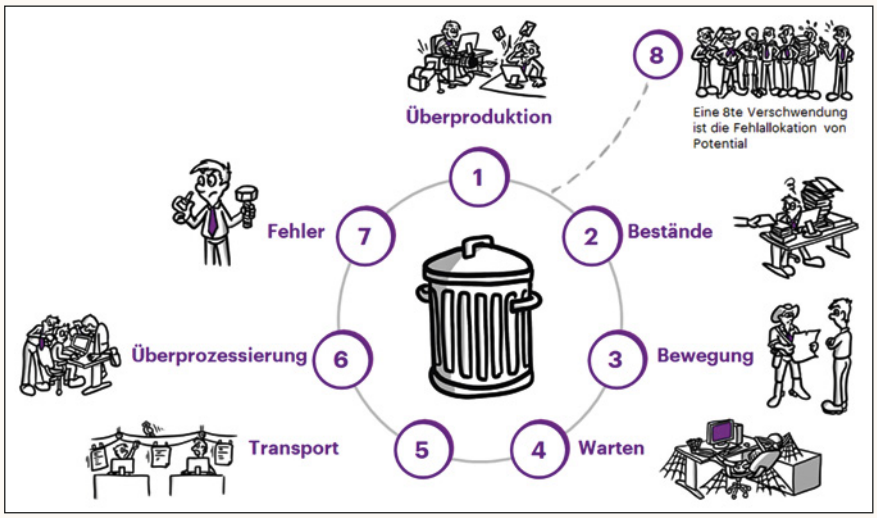


Abb. 2: TIM WOODS oder die 8 Verschwendungsarten

Referenzen

- [Bit18] K. Bittner, P. Kong, D. West, Mit dem Nexus Framework Scrum skalieren – Kontinuierliche Bereitstellung eines integrierten Produkts mit mehreren Scrum-Teams, dpunkt.verlag, 2018
- [Chr17] M. Christoph, SAFe – Das Scaled Agile Framework. Lean und Agile in großen Unternehmen skalieren, dpunkt.verlag, 2017
- [Ker19] M. Kersten, Project to Product, IT Revolution PR, 2019
- [Kom20] A. Komus, M. Kuberg, Status Quo (Scaled) Agile 2020 Report, siehe: **https://www.process-and-project.net/studien/studienunterseiten/status-quo-scaled-agile-2020/**
- [Lar17] C. Larmann, B. Vodde, Large-Scale Scrum – Scrum erfolgreich skalieren mit LeSS, dpunkt.verlag, 2017
- [Lig09] P. Liggesmeyer, Software-Qualität – Testen, Analysieren und Verifizieren von Software, Springer Spektrum, 2009
- [Pop03] M. und T. Poppendieck, Lean Software Development: An Agile Toolkit for Software Development Managers, Addison-Wesley Professional, 2003

sich „in Bewegung" befinden. Hierunter fällt zum Beispiel der allseits bekannte Klassiker, Informationen mühsam hinterherzulaufen, da es keine ganzheitliche Informationsquelle gibt.

④ *Warten:* Auch das Gegenteil der dritten Verschwendungsart, das Warten, ist Verschwendung. Die Zeit, die ein Software-Ingenieur auf Arbeitsprodukte aus vorhergehenden Schritten, auf Informationen oder auf Freigaben wartet, trägt nicht zum Mehrwert für den Kunden bei.

⑤ *Transport:* Diese Verschwendungsart ist der dritten ähnlich, nur dass sich hier das Arbeitsprodukt in Bewegung befindet. In der Testautomatisierung passiert dies beispielsweise, wenn Testfälle oder Schritte von A nach B kopiert werden, oder wenn ein komplexer Freigabeprozess durchlaufen werden muss. Auch der Transport von Testdaten auf die unterschiedlichen Testumgebungen fällt unter diese Verschwendungsart.

⑥ *Überprozessierung:* Diese Verschwendungsart bezieht sich auf die fehlende Balance zwischen Aufwand und Wert eines Prozesses. Dies ist unter anderen der Fall, wenn der Fluss einer CI/CD-Pipeline durch aufwendige manuelle Freigaben unterbrochen wird.

⑦ *Fehler:* Defects bringen dem Kunden eindeutig keinen Mehrwert und sind daher bekanntermaßen Verschwendung. Sie verursachen im Gegenteil sogar exponentiell steigende zusätzliche Kosten in ihrer Behebung je später sie gefunden werden, wie in der „Rule of Ten" [Lig09] beschrieben.

⑧ *Fehlallokation von Potenzial:* Zusätzlich zu den sieben genannten Verschwendungsarten gibt es eine letzte, die besonders im skaliert agilen Kontext relevant ist: die Fehlallokation von Potenzial. Dies bezieht sich sowohl auf die Über- als auch die Unterforderung der Mitarbeiter, beispielsweise, wenn ein funktionaler Experte als Entwickler eingesetzt wird. Oftmals wird bei agilen Transformationen nämlich nicht beachtet, dass Mitarbeiter zunächst in der neuen Methodik und den neuen Werkzeugen geschult und in der neuen Organisation abgeholt werden müssen, damit sie ihr Potenzial voll einbringen können.

Der Quality Coach – die Rolle, die alle QS-Maßnahmen zusammenhält

Um aus dem großen Katalog der lean-agilen Methoden die passenden Methoden für die QS eines Unternehmens herauszusuchen und zu einem sinnvollen Paket zu schnüren, bedarf es tiefer Kenntnisse sowohl in den lean-agilen Prozessen als auch in der QS. Zusätzlich wird Erfahrung in den Bereichen Facilitation und Coaching benötigt.

Weiterhin gilt es zu beachten, dass die Rahmenbedingungen für die QS auf den verschiedenen Ebenen eines Unternehmens unterschiedlich sein können. Das agile Skalierungsframework SAFe® beispielsweise unterscheidet vier Ebenen, von der Teamebene, über Programm- hin zu Large Solutions- und Portfolio-Ebene. Da sich diese Ebenen in ihren Aufgaben zum Teil erheblich unterscheiden, ist es sinnvoll, für jede Ebene die Rolle des Quality Coach einzuführen.

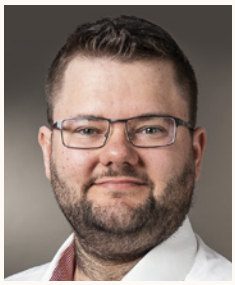
Der Quality Coach ist verantwortlich für die Einführung, Begleitung und ständige Weiterentwicklung der lean-agilen QS-Maßnahmen auf seiner Projektebene, für das Coaching der Mitarbeiter in den Teams sowie der Führungsebene. Sein Ziel ist es, dass die richtigen QS-Maßnahmen zur richtigen Zeit eingesetzt werden und der Qualitätsgedanke jederzeit präsent ist. Der Quality Coach sorgt dafür, dass sämtliche Maßnahmen abgestimmt sind und alle Optimierungen in dieselbe Richtung laufen.

Prozessqualität, der „Hidden Champion" auf dem Weg zu Business Agility

Zusammenfassend bleibt festzuhalten, dass skaliert-agile Projekte oft einhergehen mit komplexen Prozessen und Organisationsstrukturen sowie Hunderten Beteiligten. Speziell die QS sieht sich steigenden Herausforderungen gegenüber, ein übergreifendes, holistisches Qualitätskonzept zu entwickeln, welches sicherstellt, dass die Kunden zum richtigen Zeitpunkt Softwareprodukte erhalten, welche ihren Anforderungen und Wünschen entsprechen.

Lean Quality Management, mit seiner Integration von Methoden aus den Bereichen Lean und Agile, bietet sich für diese Art von Projekten als Lösung an. Speziell der Fokus auf die Verbesserung der Prozessqualität, welcher auf den Erfahrungen der Fertigungsindustrie aufbaut, hilft, die Komplexität der Prozesse zu reduzieren, Verschwendungen zu vermeiden, und somit die Liefergeschwindigkeit und Qualität der entwickelten Softwareprodukte zu steigern und somit Business Agility zu realisieren.

Damit sind die essenziellen Voraussetzungen für eine unternehmensweite Umsetzung von Business Agility erfüllt und die Weichen für eine erfolgreiche Transformation in die digitale Welt gestellt. Lean Agile Quality Management ist also eine entscheidende Fähigkeit, um in der aktuellen und zukünftigen Wirtschaftswelt eine führende Rolle übernehmen zu können.



Thomas Karl

thomas.karl@accenture.com
ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

Bettina Hillringhaus

bettina.hillringhaus@accenture.com
ist Quality Engineer und Coach mit Schwerpunkt auf agilem Qualitätsmanagement und Prozessverbesserungen. Sie hat Erfahrungen bei komplexen Großprojekten in verschiedenen Industrien und mit Fokus auf unterschiedlichen Technologien gesammelt. Als Experte für agile Qualitätsstrategien hat sie mehrere Transformationen begleitet und Konzepte zum Lean Quality Management eingeführt.

»Erste Einblicke in die Umfrage des GTB«

Das German Testing Board (GTB e.V.) hat unter wissenschaftlicher Leitung der TH Köln und der Hochschule Bremerhaven zusammen mit dem Austrian Testing Board (ATB) im Frühsommer 2020 wieder eine der größten Umfragen im Bereich „Test und Qualitätssicherung" im deutschsprachigen Raum durchgeführt. Über 1.250 Teilnehmende beantworteten Fragen zur Qualitätssicherung aus Sicht des Managements, der operativen Umsetzung und der Forschung. Dieser Artikel gibt erste Einblicke in das Profil der Teilnehmenden sowie in einige Antworten der operativ Umsetzenden.

Kennen Sie Aussagen wie diese: *Software wird nur noch offshore getestet! Testautomatisierung lohnt sich nicht! Testmanagement wird nicht mehr benötigt! Softwarequalität hängt nur von den richtigen Entwicklern ab! KI-Systeme werden zukünftig das Testen bestimmen!*

Wir wollten es genau wissen! Dazu haben wir im Frühsommer 2020 zum dritten Mal im 5-Jahres-Abstand eine der größten Umfragen im Bereich des Testens und der Qualitätssicherung (QS) im deutschsprachigen Raum erfolgreich durchgeführt. Die wissenschaftliche Leitung des Umfrage-Projekts liegt bei der Technischen Hochschule Köln und der Hochschule Bremerhaven. Ideeller, personeller (und finanzieller) Förderer ist das German Testing Board (GTB e.V.) zusammen mit dem Austrian Testing Board (ATB). Bei der Bewerbung der Umfrage unterstützten uns der ASQF e.V., bitkom, der dpunkt.verlag, der Fachbereich „Softwaretechnik" der Gesellschaft für Informatik (GI e.V.), die GI-Fach-

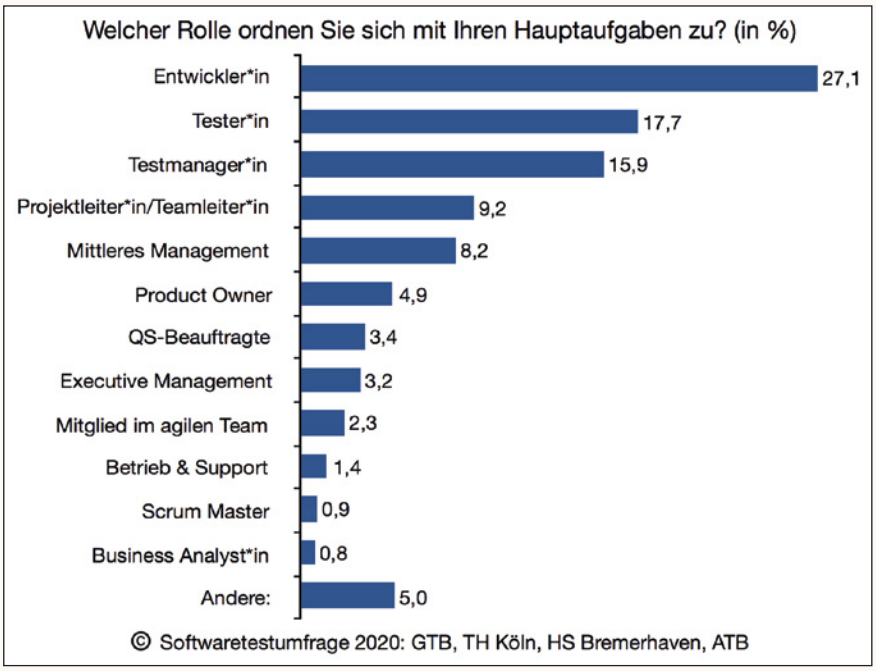


Abb. 1: Rollenzuordnung der Operativen

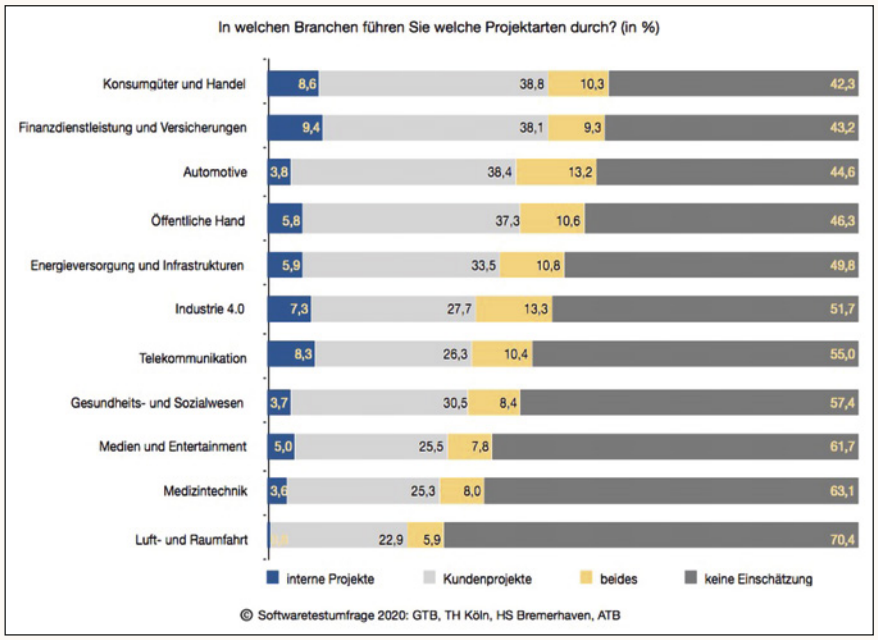


Abb. 2: Branchen und Projektarten

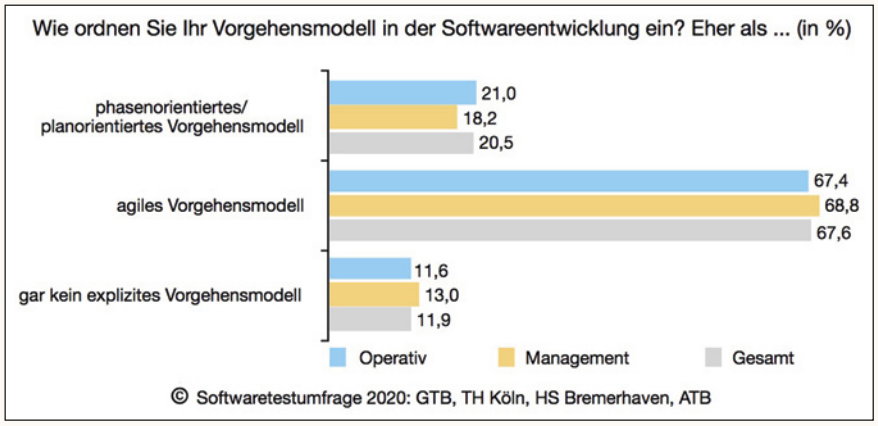


Abb. 3: Vorgehensmodell

gruppe „Test, Analyse und Verifikation von Software (TAV), die Softwareforen Leipzig und das Swiss Testing Board (STB).

Der Fragenkatalog wurde bewusst ähnlich zu den vorherigen Softwaretest-Umfragen 2015/16 und 2011 gehalten, damit unsere aktuellen Analysen die damaligen Einblicke in aktuelle und zukünftige Trends und Herausforderungen rund um die Qualitätssicherung von Software und Systemen fortführen. Im Folgenden kürzen wir der besseren Lesbarkeit halber die Umfrageergebnisse aus 2015/16 mit 2015 ab.

Die Ergebnisse sollen auf der einen Seite einem möglichst breiten Interessentenkreis erlauben, Impulse für die praxis- und forschungsorientierte Ausrichtung der Aus- und Weiterbildung sowie Qualifizierung abzuleiten, sowie auf der anderen Seite den Unternehmen mit den Ergebnissen Grundlagen für ein Benchmarking zur Verfügung zu stellen.

In diesem Artikel skizzieren wir das Profil der Teilnehmenden und stellen einige Ergebnisse aus der operativen Umsetzung vor, um Ihr Interesse an unseren detaillierten Analysen zu wecken.

Wer hat geantwortet?

Die drei Fragebögen waren vom 5. Mai bis zum 30. Juni 2020 online. Mit 1051 (2015: 775, 2011: 1402) Teilnehmenden aus dem Bereich Entwicklung und Test **(siehe Abbildung 1)** sowie 196 (2015: 217, 2011: 221) aus dem Bereich Management erfuhr die Umfrage wieder erfreulichen Zuspruch. Enttäuschend ist dagegen, dass lediglich 60 Teilnehmende zur Forschung antworteten (Fragebogen zur Forschung erst ab 2015, dort: 193).

Interessant ist ein Blick auf die Teilnehmenden des Management-Fragebogens: Ordneten sich 2011 deutlich über 3% der Gruppe des Executive Managements zu, so waren es bei der Umfrage 2015 erfreuliche 5%, jedoch bei der aktuellen Umfrage 2020 nur noch knapp über 3%. Beim mittleren Management sind es aktuell nur noch 8% im Vergleich zu 9% im Jahr 2011 und 14% in 2015. Hat die Relevanz der Themen um Qualitätssicherung und Softwaretest in den Führungsetagen wieder abgenommen? Oder fehlt aufgrund von COVID-19 einfach die Zeit?

Bei den Branchen der Teilnehmenden sieht es ähnlich wie in den beiden früheren Umfragen aus, wobei hier Mehrfach-Antworten erlaubt waren **(siehe Abbildung 2)**. Knapp vorne liegen Konsumgüter und Handel mit 58%, gefolgt von Finanzdienstleistern und Versicherungen mit 57% und Automotive mit 56% sowie der öffentlichen Hand mit 54%.

Agil? Oder doch (noch) nicht?

Wie erwartet sehen wir auch dieses Jahr die weitere Etablierung agiler Vorgehensweisen **(siehe Abbildung 3)**: So geben nur noch knapp 21% der Teilnehmenden an, ein klassisches phasenorientiertes Vorgehensmodell anzuwenden (2015: 47%, 2011: 54%), wohingegen über 67% agile Vorgehensmodelle benutzen (2015: 43%, 2011: 29%).

Befragt nach der eigenen Rolle, dominieren wie in **Abbildung 1** zu erkennen immer noch klar die klassischen Rollenbilder wie Entwickler*in, Testmanager*in und Tester*in. Die agilen Rollen wie Scrum Master, Product Owner und Teammitglied sind noch in der Minderheit. Die Vision von T-Shaped-Skills in agilen Teams wird demnach nicht gelebt.

Bei der Analyse der verwendeten Praktiken im Bereich der Qualitätssicherung **(siehe Abbildung 4)** führen diejenigen das Ranking an, die wenig Vorbedingungen an einen agilen Mindset haben, sondern primär technisch orientiert direkt anwendbar sind: So geben z.B. 73% der Teilnehmenden an, die Clean-

Code-Technik anzuwenden, 79% führen Code-Reviews durch und 74% wenden die Technik des Continuous Integration an. Die Praktiken, bei denen der agile Teamgedanke notwendige Vorbedingung ist, zeigen hier deutlich geringere Werte in Bezug auf ihre Bedeutung für die Qualitätssicherung: So wenden 45% der Teilnehmenden die Collective Code-Ownership an, 49% machen eine gemeinsame Aufwandsschätzung und 44% nutzen Pair-Programming.

Spannend ist darüber hinaus noch, dass bei der agilen Transition viele klassische Methoden im Bereich des Testens scheinbar an Bedeutung verlieren **(siehe Abbildung 5)**. So wenden nur noch 25% die Äquivalenzklassenanalyse regelmäßig an (2015: 38%, 2011: 61%), 12% nutzen Entscheidungstabellen (2015: 18%, 2011: 33%) und 4% greifen zur Erstellung der Testfälle auf das Klassifikationsbaumverfahren zurück (2016: 7%, 2011: 12%).

Schon diese Blitzlichter zeigen, dass die Welt des Testens und der Qualitätssicherung weiter in Bewegung ist! Die Diagramme mit den Antwortverteilungen zu allen Fragen sind bereits im Internet auf der Web-Seite https://www.softwaretest-umfrage.de/.

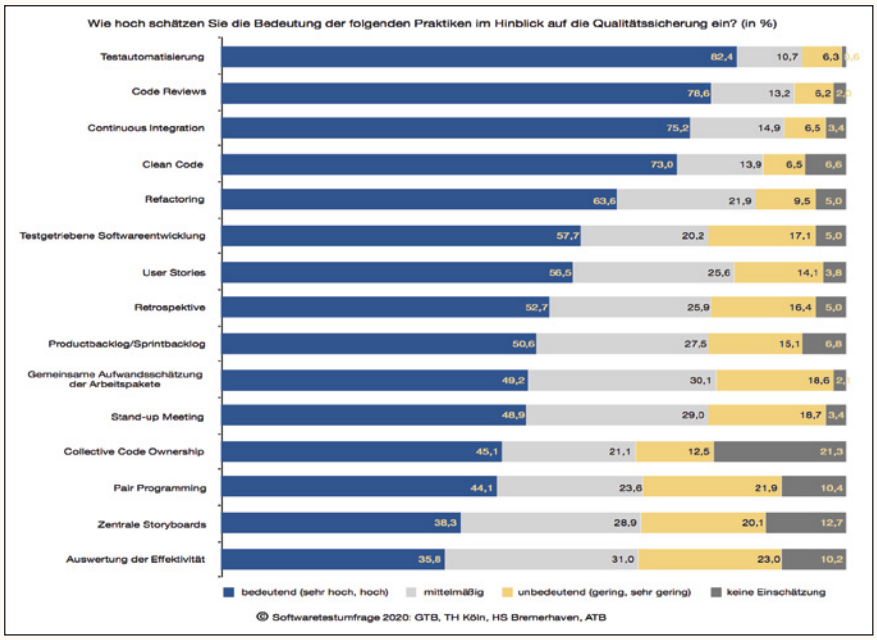


Abb. 4: Bedeutung der Praktiken für die QS

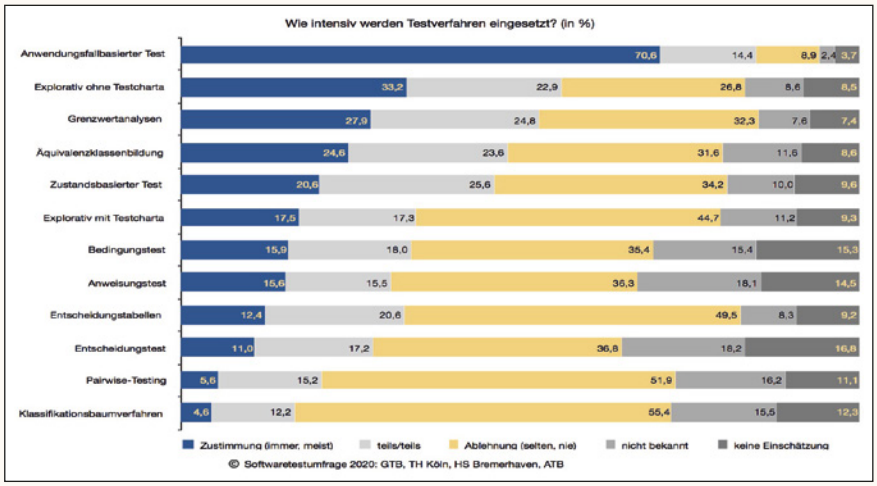


Abb. 5: Einsatz klassischer Testverfahren

- **Prof. Dr. Mario Winter** (mario.winter@th-koeln.de)
  ist Professor am Institut für Informatik der TH Köln, Gründungsmitglied des German Testing Board e.V. und Sprecher des Arbeitskreises „Testen und KI" in GI-Fachgruppe „Test, Analyse und Verifikation von Software".
- **Prof. Dr. Karin Vosseberg** (karin.vosseberg@hs-bremerhaven.de)
  ist Professorin für IT-Systemintegration mit dem besonderen Fokus auf Qualität von Software im Studienbereich Informatik an der Hochschule Bremerhaven und Präsidiumsmitglied im ASQF.
- **Frank Simon** (f.simon@gtb.de)
  arbeitet als Head of IT-Security & IAM in der Zurich Gruppe Deutschland. Im German Testing Board leitet er die Arbeitsgruppen Security.
- **Annette Simon** (a.simon@gtb.de)
  arbeitet als wissenschaftliche Mitarbeiterin bei der Fraunhofer Gesellschaft e.V. im Bereich des integrierten, nationalen und internationalen Projektmanagements. Im German Testing Board leitet sie die Arbeitsgruppe Marketing.
- **Misperi Sakarya** (misperi.sakarya@smail.th-koeln.de)
  studiert Wirtschaftsinformatik an der TH Köln, Campus Gummersbach.

»Quality Engineering geht jeden etwas an«

Software ist zu einem Bestandteil unseres Lebens geworden – und wird in Zukunft noch mehr an Bedeutung gewinnen. Egal ob Smartphone, Fitnessarmband, Navigationssystem, Auto, Fahrkartenautomat, die Kasse im Supermarkt, der Rechner am Arbeitsplatz oder andere alltägliche Dinge: All dies funktioniert nur dann wie gewünscht, wenn die zugehörige Software fehlerfrei arbeitet. Fehler werden nicht mehr einfach so verziehen – Qualität ist die zentrale Anforderung an Software geworden!

Qualität ist eine berechtigte und notwendige Anforderung an jede Art von Softwareapplikation, denn Software ist ein zentraler „Taktgeber" in unserem täglichen Sein. „Quality Engineering" ist die Disziplin, die im gesamten Entwicklungsvorgehen die Fäden zusammenhält, um ein effizientes Qualitäts-Setup zu etablieren. Der wirtschaftliche Nutzen des Quality Engineering ist somit sehr groß: Qualität muss stimmen, da ein Nutzer stets die Möglichkeit hat, bei der Konkurrenz ein vergleichbares Produkt zu finden.

Softwarequalität bezieht sich dabei nicht nur auf die Funktionalität, sprich den fachlich korrekten Ablauf eines Programms. Es gilt auch die Qualitätsanforderungen hinsichtlich der Performance und Kompatibilität ebenso wie die der Wartbarkeit und Bedienbarkeit („Usability") zu berücksichtigen. In der öffentlichen Diskussion ganz weit oben ist vor allem das Thema Sicherheit – sowohl der Anwendung als auch der erfassten Daten.

Wieso Quality Engineering?

Das Erreichen all dieser Qualitätsanforderungen muss daher über den gesamten Softwareentwicklungszyklus hinweg effizient sichergestellt werden. Gleichzeitig wollen und benötigen Nutzer immer schneller und öfter Updates ihrer Programme und Apps – und natürlich auch neue Software.

Softwaretesten ist und bleibt also wichtig und richtig. Dreimonatige Testphasen, wie sie noch vor zehn Jahren in vielen Firmen gang und gäbe waren, sind aber aufgrund der gravierend geänderten Nutzeranforderungen allerdings längst nicht mehr umsetzbar. In der Realität werden dem Kunden die Softwareupdates mittlerweile häufig in sehr hoher Frequenz zur Verfügung gestellt. Daher benötigen wir bei der Softwarequalitätssicherung völlig neue Ideen, innovative Lösungen und

„Die einzigartige Mischung aus Innovation, permanentem Hinterfragen des Status quo, der Zusammenarbeit mit vielen unterschiedlichen Menschen und dem ständigen Streben nach Perfektion macht für mich den Reiz an diesem Beruf aus. Als studierter Betriebswirt in die IT-Beratung einzusteigen, nicht dem Schema F zu folgen und Business-Analyst zu werden, sondern als Tester den Softwareentwicklern Fehler im System aufzuzeigen, war am Anfang ein absoluter Kulturschock für mich.

Nach der ersten Schockstarre dann aber Chancen zu suchen, die Softwareentwicklung zu beschleunigen und die Qualität zu steigern, war und ist fantastisch. Als Quality Engineer auf einem der ersten großen agilen Projekte im deutschsprachigen Raum selbst Methoden und Konzepte für die Qualitätssicherung zu entwickeln, hat meine persönliche Lernkurve bis weit über die Wolkendecke ansteigen lassen. Menschen, Prozesse, Technologien und Firmen weiterzuentwickeln, um Kundenbedürfnisse noch besser zu erfüllen, ist nun zu meiner Passion geworden."

Tim Hansen:
Perfektion über den Wolken

Studiengang: Betriebswirtschaftslehre (Dipl.-Kfm. Univ.)
Erfahrung in der Software-QA: 10+ Jahre

„Die Kombination aus Neugier, Wissensdurst und Kreativität hat mich mein Leben lang begleitet. Gemeinsam mit einem analytischen Denkansatz, den ich mir in meinem Studium angeeignet habe, sind es auch diese Fähigkeiten, die ich als Quality Engineer täglich verwende. Für mich als Quereinsteigerin war die Lernkurve gerade zu Beginn sehr steil, aber dank der Unterstützung meiner Teammitglieder nie unerklimmbar.

Mit jedem Projekt habe ich neue Fähigkeiten hinzugewonnen, beispielsweise in der Testautomatisierung und im Qualitätsmanagement. Immer wieder habe ich den Sprung ins kalte Wasser gewagt und meinen Kunden geholfen, ihr Qualitätsvorgehen zu analysieren und systematisch zu verbessern, indem wir gemeinsam innovative Lösungen ihrer jeweiligen Situation gefunden und umgesetzt haben. Gerade dieser kreative Dialog ist es, der mich als Quality Engineer begeistert.

Carla Duisberg:
Der Sprung ins kalte Wasser

Studiengang: Studiengang: Physik (M. Sc.)
Erfahrung in der Software-QA: 5+ Jahre

vor allem schnelle Hilfe! Wir benötigen Qualitätsingenieure – „Quality Engineers"!

Tester vs. Quality Engineer

Tests in einem klar vorbestimmten Zeitrahmen nach endgültig definierten Anforderungen zu planen und auszuführen, entspricht nicht mehr der Realität im agilen Zeitalter. Vor der Agilität hat ein Tester nach Erreichung eines „Quality Gate" zunächst die Anforderungen und später die fertig entwickelte Software erhalten. Erst dann wurde (vom Entwicklertest abgesehen) die Qualität der Software geprüft. Häufig war es ein Ergebnis dieses Vorgehens, dass Abweichungen von der Qualität spät entdeckt wurden. Die Folge: Die erforderliche Qualität wurde in Form von Fehlerbehebung nachträglich in die Software „hineinentwickelt". Modernes Quality Engineering versteht sich dagegen als integraler Bestandteil des Softwareentwicklungsprozesses (SDLC).

Quality Engineers sind im gesamten Prozess präsent – von der Anforderungsdefinition über die Entwicklung und Test bis hin zur Produktivsetzung – und stellen in den einzelnen Phasen die passenden Qualitätssicherungsmaßnahmen bereit **(siehe Abbildung 1)**. In den schnellen, agilen Releasezyklen stellt das eine besondere Herausforderung dar, da nur wenige Tage zwischen der Definition der Anforderungen, der Entwicklung und der Implementierung der neuen Software liegen.

Die Vorstellungen und Wünsche, welche die Software zu erfüllen hat, muss der Quality Engineer umfassend verstehen und zum richtigen Zeitpunkt, mit den richtigen Methoden, Werkzeugen und Maßnahmen die Umsetzung dieser Anforderungen sicherstellen.

Holistische Strategien entwickeln

Von Beginn des Entwicklungsprozesses an ist es die Aufgabe des Qualitätsmanagers oder Quality-Engineering-Coaches, eine holistische Strategie zu entwickeln, mit der dann die Qualitätsziele des Unternehmens in der Produktentwicklung beschrieben werden können, und wie der Weg aussieht, auf dem diese erreicht werden können. Hierzu wird definiert, zu welchen Zeitpunkten des Entwicklungszyklus welche Maßnahmen zur Qualitätssicherung (QA) eingesetzt werden – und wie diese Maßnahmen durch die Unterstützung geeigneter Werkzeuge, etwa zur Testautomatisierung oder für das Testdatenmanagement, sinnvoll umgesetzt werden können.

Weiterhin beschreibt der Quality Engineer, mit welchen Methoden Tests von Anforderungen abgeleitet werden und wie Testdaten

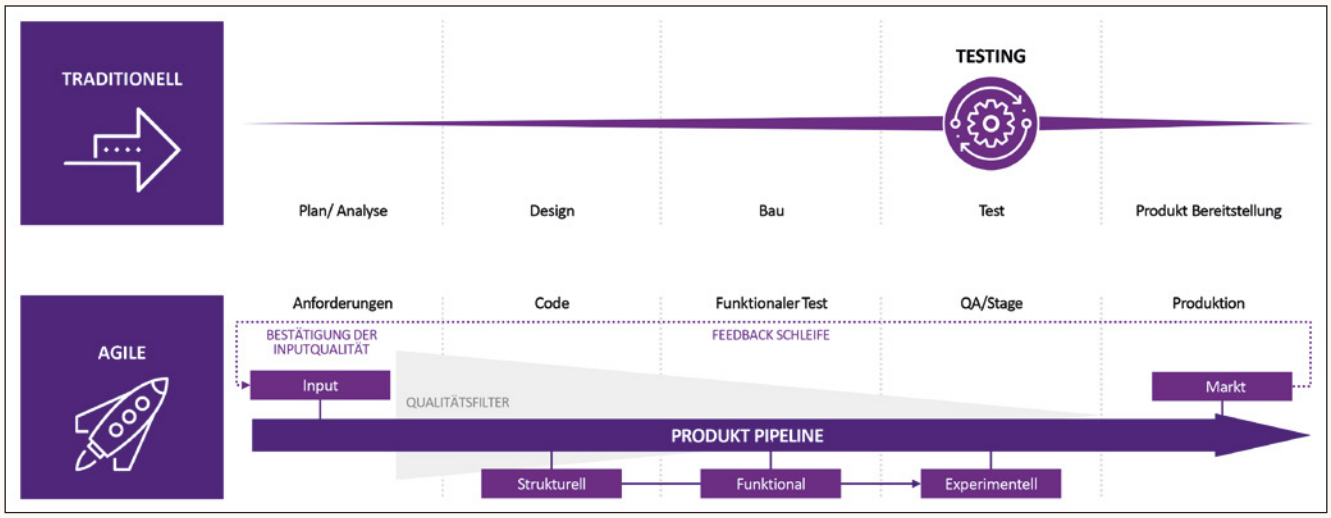


Abb. 1: Quality Engineering in der traditionellen und agilen Softwareentwicklung

„Das Thema Quality Engineering begleitet mich seit mehr als 15 Jahren, von meiner Ausbildung über mein Studium bis zu meiner Arbeit bei Accenture. Hier kann ich als technisch versierter Generalist meine Erfahrungen in allen drei Teilbereichen – Automatisierung, Methodik und Management – gleichermaßen zum Einsatz bringen.

Mein Drang, stets am Puls der Zeit sein zu wollen, und die dazugehörige Passion, Dinge richtig zu machen, gehören hier genauso dazu wie das stetige Interesse, sich und sein Umfeld weiterzuentwickeln. Das ist es, was mich vom Supportmitarbeiter über einen Tester in einem Team dahin getrieben hat, wo ich jetzt bin. Bei Accenture habe ich die Möglichkeit, bei großen nationalen sowie internationalen Unternehmen meine Passion strategisch in die QM-Prozesse einfließen zu lassen und diese mit unseren Teams zum Leben zu erwecken."

Frank Fuchs:
Die stetige Entwicklung zum Quality Engineer

Studiengang: Wirtschaftsinformatik (M. Sc.)
Erfahrung in der QA: 15+ Jahre

„In meiner Tätigkeit als Quality Engineer sind meine Kommunikationsfähigkeiten und meine sprachwissenschaftlichen Kenntnisse täglich stark gefragt. Im Bereich Testkoordination habe ich Organisations- und Führungs-Skills entwickelt. Außerdem habe ich die Möglichkeit bekommen, technische Skills im Bereich der Testautomatisierung zu erwerben und diese in der Praxis einzusetzen.

An meiner Arbeit mag ich vor allem die enge Zusammenarbeit mit den Kollegen, die Abwechselung und die herausfordernden Aufgaben."

Franceska Carlova:
Facettenreiche Entwicklung

Studiengang: Medienwissenschaft & Anglistik (M. A.)
Erfahrung in der Software-QA: 10 Jahre

und Umgebungen für die Tests bereitgestellt werden. Als Automatisierungsexperte kümmert er sich dann sowohl um die fachliche als auch um die technische Einführung der Qualitätsstrategie. Als Architekt verantwortet er die Inbetriebnahme, Wartung und Ablösung der Werkzeuge, die zur Unterstützung der Maßnahmen benötigt werden.

Schließlich ist der Quality Engineer verantwortlich für die Ausführung der vorgesehenen Methoden und Maßnahmen, wobei er kontinuierlich an der Verbesserung der Prozesse beteiligt ist. Er agiert dabei als Vermittler im Team, kommuniziert auf Augenhöhe mit Entwicklern, Fachexperten und Kunden, und hinterfragt Entscheidungen frühzeitig. Das Verständnis der Quality Engineers vom Produkt sowie das schnelle Feedback sind kritisch, damit die Qualität des Produkts und die Release Deadline nicht gefährdet werden. Das Erfolgsgeheimnis liegt dabei in der engen Zusammenarbeit aller Kollegen im Team.

Unterschiedliche Karrierepfade im Quality Engineering

In einem Quality-Engineering-Team sind Kollegen mit technischen Skills und Fachexpertise sowie Kollegen, die organisatorisches Talent und ein Auge fürs Detail mitbringen, gleich geschätzt **(siehe Abbildung 2)**.

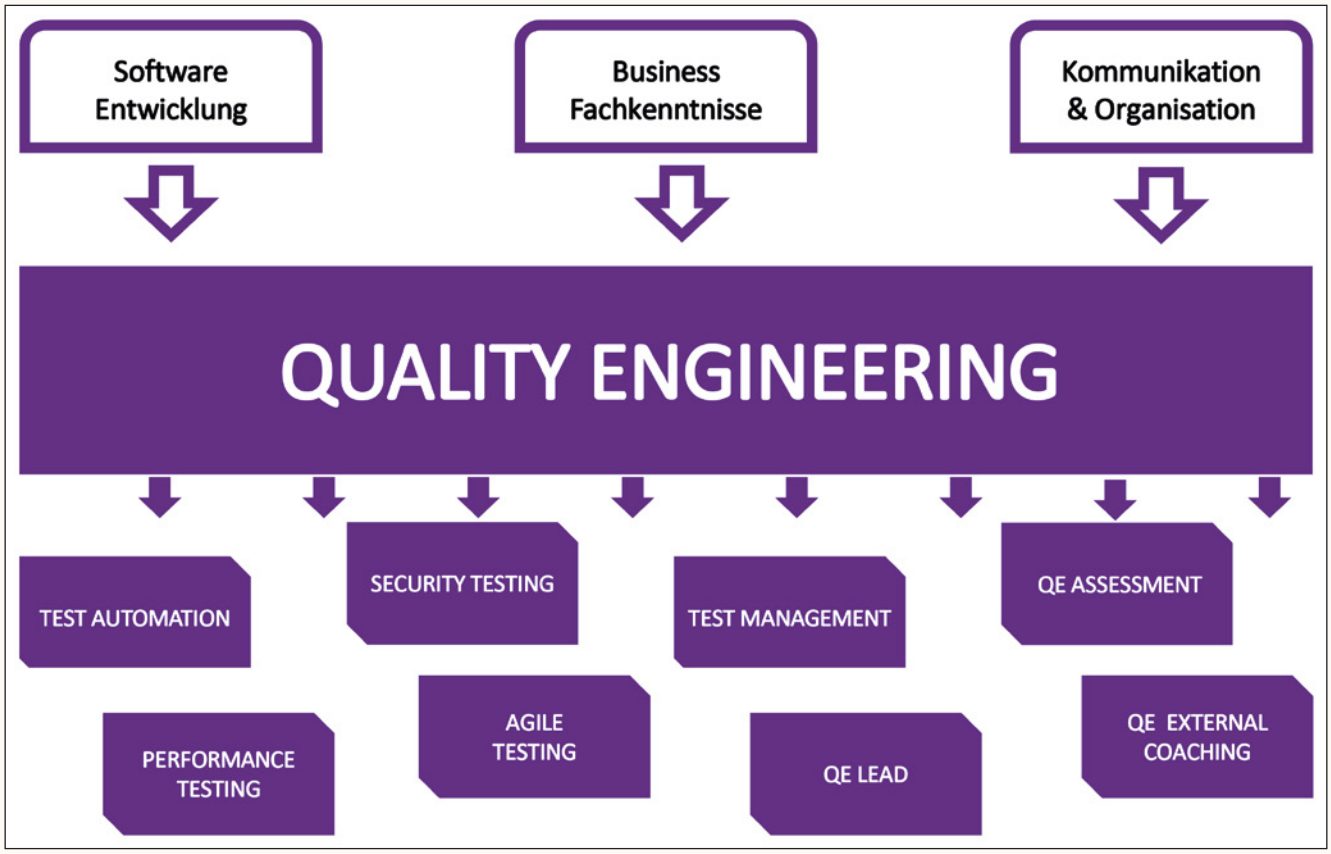


Abb 2: Quality Engineering: Fachkenntnisse und berufliche Ausrichtungen

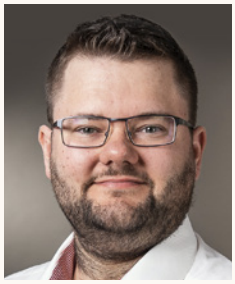
Das Streben nach Qualität ist die Motivation, die alle Quality Engineers antreibt. Ihre Fähigkeiten sind dabei sehr divers. Die technisch Versierteren können einen großen Beitrag in der Testautomatisierung leisten. Die methodisch Orientierten finden sich besser im Testdesign, der Testplanung sowie der Teststrategie zurecht.

Auf jeden Fall werden im Qualitätsmanagement Mitarbeiter mit ausgeprägter Führungsstärke gebraucht. Ein Quality Engineer mag auf die Testautomatisierung spezialisiert sein, wird aber immer wieder mit Testvorgehensweisen und Qualitätsmanagement konfrontiert sein. Somit lernen Kollegen mit unterschiedlichen Stärken voneinander, entwickeln ihre Fähigkeiten gemeinsam weiter und unterstützen sich gegenseitig, um Software in bester Qualität zu liefern. Sie unterstützen das ganze Team mit Schulungen und Beratung zur Qualität.

Die Tätigkeit eines Quality Engineers kann also sehr unterschiedlich sein. Was Mitarbeiter bei Accenture daran spannend finden, machen die vier ganz unterschiedlichen Beispiele in den Infokästen deutlich. Letztlich bedeutet Quality Engineering: umdenken, anpassen und neu erfinden. Organisiert in einer Community, gestalten heute bei Accenture über 400 Quality Engineers im deutschsprachigen Raum die Zukunft von Quality Engineering mit. Über Länder- und Hierarchiegrenzen hinweg werden Informationen und Ideen ausgetauscht, um gemeinsam die Zukunft der Softwareentwicklungsbranche und letztlich unseres Alltags mitzugestalten. Dabei sind Quality Engineers stets am Puls der Zeit und treten jeder Herausforderung zuversichtlich entgegen.

Teodora Petrova
teodora.petrova@accenture.com
ist Quality Engineer und Advisor spezialisiert in Testautomatisierung, Test-Design, agiles Testen, Testkoordination und Defect-Management. Sie hat den Wandel zum Agile Testing auf unterschiedlichen Projekten bei der Einführung und Optimierung des agilen Testprozesses unterstützt. Als Consultant führt sie Analysen durch, die dazu beitragen, die Maßnahmen zur Qualitätssicherung zu verbessern.

Bettina Hillringhaus
bettina.hillringhaus@accenture.com
ist Quality Engineer und Coach mit Schwerpunkt auf agilem Qualitätsmanagement und Prozessverbesserungen. Sie hat Erfahrungen bei komplexen Großprojekten in verschiedenen Industrien und mit Fokus auf unterschiedlichen Technologien gesammelt. Als Experte für agile Qualitätsstrategien hat sie mehrere Transformationen begleitet und Konzepte zum Lean Quality Management eingeführt.

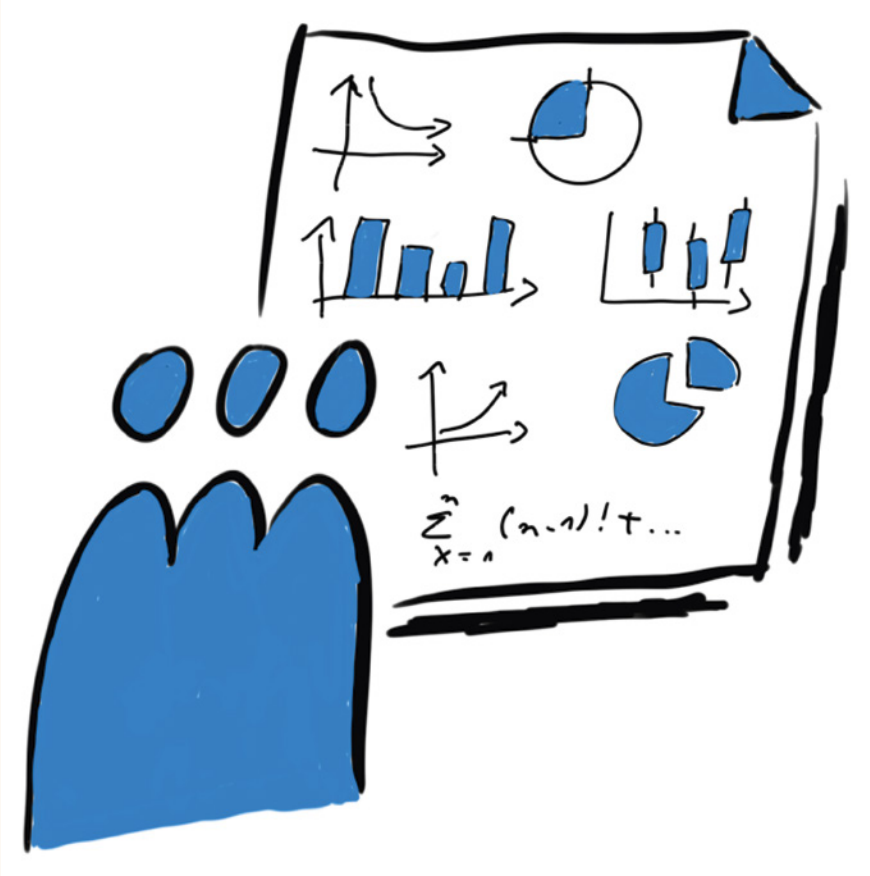


Thomas Karl
thomas.karl@accenture.com
ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

Nico Liedl
nico.liedl@accenture.com
ist Quality Engineer und Advisor mit Schwerpunkt auf agilem Qualitätsmanagement, Testautomatisierung, Prozessverbesserung und organisatorischem Wandel. Er hat Erfahrungen bei verschiedenen komplexen Großprojekten gesammelt. Als Teammitglied, Scrum-Master, Teststratege, Releasemanager und Trainer hat er praktische Erfahrung und theoretisches Fachwissen zu bieten.

»Drei Qualitätsdimensionen einer agilen Organisation«

Agilität ist in der Softwareentwicklung der IT weitestgehend etabliert. Die Geschäftsbereiche entdecken zunehmend die Vorteile der agilen Idee und sind dabei, die agilen Ansätze für sich zu adaptieren. Welche Möglichkeiten bieten sich für die IT, die Transformation der Geschäftsbereiche aktiv zu unterstützen? Können gesammelte Erfahrungen und Gelerntes mit hoher Qualität in die Geschäftsbereiche übertragen werden? Welche Änderungen oder Rückkopplungen bringt der Wandel der Geschäftsbereiche wieder in die IT? Wir zeigen die drei Qualitätsdimensionen einer agilen Organisation. Die intensivere Kopplung der Geschäftsbereiche und der IT im Rahmen der Produktdigitalisierung zeigen wir am Beispiel von mobile Online-Diensten von Fahrzeugen.



In großen Organisationen ist es nicht trivial, spezifisches Wissen über Methoden und Tools systematisch und nachhaltig zu skalieren. Auch agile Modelle sehen zur Wissensbündelung und Verteilung zum Beispiel Gilden wie im Spotify-Modell vor. In der Group IT der Volkswagen AG ist das Competence Center Test & Quality Assurance (TQA) seit Langem etabliert. Das Agile Center of Excellence (ACE) wurde gegründet, um die Group IT systematisch bei der agilen Transition zu unterstützen. Sowohl das ACE als auch das TQA unterstützen systematisch die IT-Entwicklungsbereiche sowie die Fachbereiche, um Kunden hochwertige Produkte und Services bereitzustellen. Dabei ergänzen sich beide Bereiche sowohl im Leistungsportfolio als auch in ihrem Vorgehensmodell.

Synergien aus Qualität und Agilität

Voraussetzung für die Unterstützung ist ein gemeinsames Verständnis zwischen den Bereichen TQA und ACE. Das, was so einleuchtend klingt, war nicht immer selbstverständlich. Während auf der einen Seite agile Evangelisten immer das Mantra „Qualität ist inhärent" vor sich hertrugen und dabei unterschätzten, dass es fachspezifische Anforderungen gibt, erkannten die eingefleischten Qualitätsmanager nicht das enorme Potenzial, welches im agilen Ansatz steckt.

Erst als allen klar war, dass domänenspezifische Anforderungen wie beispielsweise Sicherheitsnormen spezieller Absicherungsstrategien bedürfen und Qualitätsanforderungen bereits in agilen Ansätzen wie der Definition of Done, den Akzeptanzkriterien und den Daily-Stand-up Meetings stecken, konnten Synergien zwischen beiden Gruppen gehoben werden.

Eine zweite Fragestellung, die beide Bereiche betraf, ist die Frage nach der Skalierung des Leistungsportfolios. Sowohl durch die Digitalisierung der Prozesse im Konzern als auch durch das Angebotswachstum digitaler Produkte für Kunden können zentrale Einheiten nicht schnell genug Ressourcen aufbauen und wachsen. Hier mussten auch neue Wege gefunden werden: von zentralen Organisationsformen, die zwangsläufig zum Nadelöhr werden, hin zu hybriden Modellen, die eine fachliche zentrale Steuerung mit lokaler operativer Leistungserbringung verbinden. So unterschiedlich beide Bereiche sind, so ähnlich wurden die gefundenen Antworten.

Eine wichtige Gemeinsamkeit, die eine enge Zusammenarbeit erst ermöglicht hat, war die konsequente Ausrichtung von TQA und ACE hin zu unterstützenden Einheiten mit nur geringem Governance-Umfang.

Diese Zusammenarbeit hat einige interessante Erkenntnisse und Ergebnisse geschaffen. Im Folgenden wollen wir zwei generische Beispiele vorstellen, die in verschiedenen Geschäftsbereichen agiler Organisationen anwendbar sind:

- die Wissensskalierung mittels Self-Service Kits und
- Qualitätsmanagement mit Fokus auf agile Teams.

Self-Service Kit

Mit dem Ansatz *Self-Service Kit* (SSK) wird den Teams Experten-Wissen zu Methoden und deren Anwendung auf qualitativ hohem Niveau bereitgestellt. Die so realisierte Skalierung von Wissen entlastet die agilen Coaches des ACE sowie die Qualitätsspezialisten von TQA.

Ein SSK ist eine Bündelung von verschiedenen Trainings und Lernmethoden mit dem Ziel, Wissen zu vermitteln und zur Anwendung in die Teams zu überführen sowie das nötige Hintergrundwissen bereitzustellen, um den Teams spezifische Adaptionen zu ermöglichen.

Der SSK-Ansatz ist so gestaltet, dass er agile Werte und Prinzipien unterstützt und die Autonomie der Teams fördert. Jeder SSK behandelt ein Thema wie die Erstellung der Levels of Done [Pot-x20, Pot-e120] (Prozess-Fokus), die spezifische Erhebung von Produkt-Qualitätsrisiken [Pot-r20] (Produkt-Fokus) oder die Bestimmung der Team-Arbeitsqualität [Pot-x120] (Team-Fokus). In den Wissensgebieten zu Produkt-, Prozess- und Teamqualität werden verschiedene Schwerpunkte in der Wissensvermittlung bei der SSK-Erstellung gesetzt [Pot-f20] und im Rahmen des Lebenszyklus-Modells des SSK aktualisiert und verbessert. Des Weiteren ermöglicht es der SSK-Ansatz, die autonomen Teams einer Organisation zu Prosumenten zu machen, da alle Experten mit dem SSK zur Erstellung von SSKs befähigt werden, eigenständig einen SSK bereitzustellen.

Team Work Quality

In der klassischen Organisation (Taylorismus) werden Produkte über Prozessen geschaffen und die Mitarbeiter sind prozessschrittausführende Einheiten. Somit sind die im Fokus befindlichen beiden Dimensionen des Qualitätsmanagements die *Produkt- und Prozessqualität*.

In agilen Organisationen ist nicht mehr der Prozess der Mittelpunkt, sondern das Team. Um agilen Teams Autonomie zu gewähren, benötigt es auch Vertrauen seitens der Organisation, dass ein hinreichendes Können für die übertragene Produkt- oder Service-Bereitstellung vorhanden ist und kontinuierlich weiter entwickelt wird, um die Verantwortung für das eigenständige Handeln auch vollumfänglich übernehmen zu können. Die Produkt- und Prozessqualität werden über die klassischen Methoden des Qualitätsmanagements abgedeckt. Im agilen Kontext wird die Produktqualität über die Definition of Done (DoD) und Akzeptanzkriterien sichergestellt, wobei auch Aspekte der Prozessqualität darin abgebildet werden können.

Die zusätzliche neue dritte Dimension einer agilen Organisation, die *Qualität der Teamarbeit*, ist dabei bisher weder systematisch in den klassischen noch den agilen Vorgehensmodellen abgebildet. **Abbildung 1** zeigt die Möglichkeiten der eigenständigen Gestaltung von Prozessen und Produkten bei hinreichender Teamqualität. Scrum und SAFe bieten wenige Indikatoren für die Qualität der Teamarbeit [Pot-e20]. Aus diesem Grund wurde der Ansatz agile Team Work Quality (aTWQ) von uns entwickelt. Er kann als „Self-Assessment" von den Teams über ein

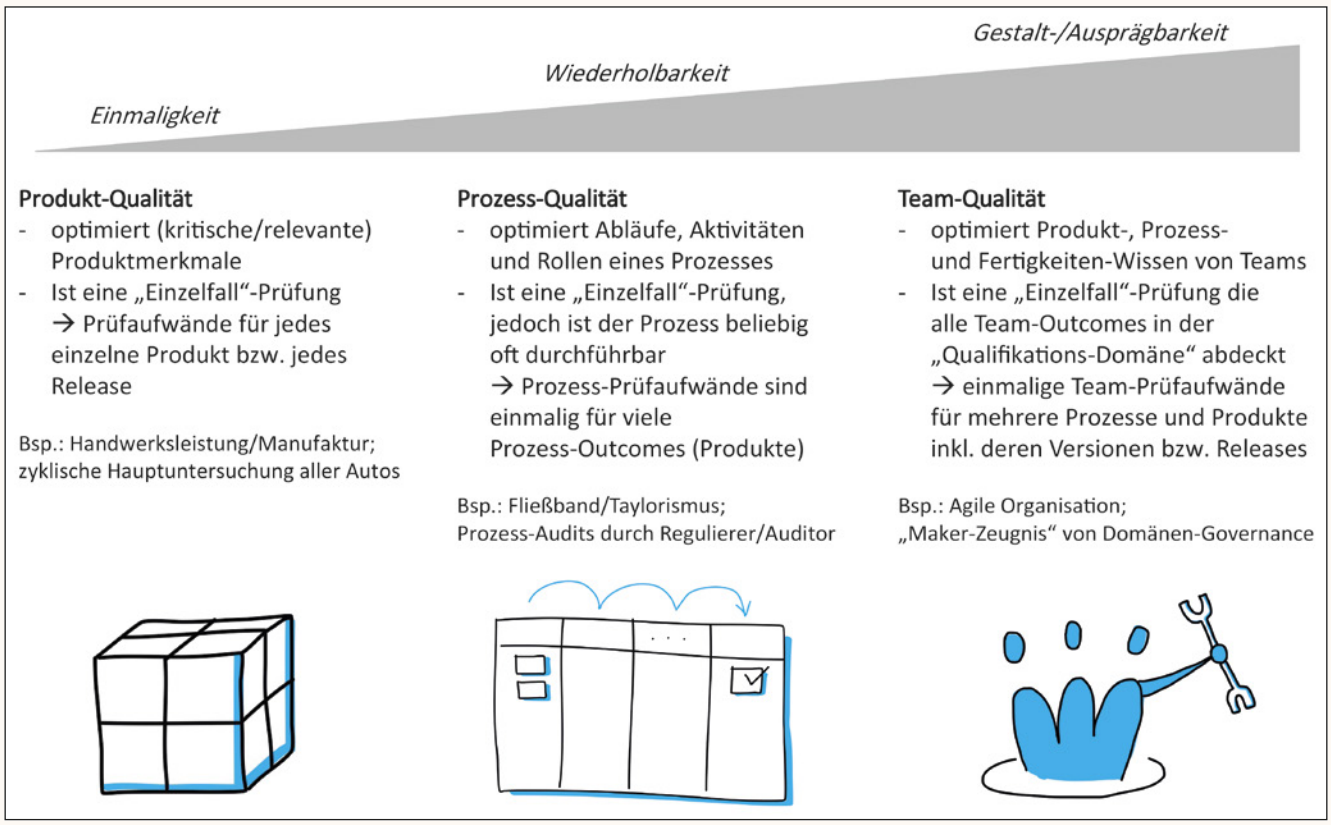


Abb. 1: Qualität für Gestaltungsfreiräume mittels den neuen 3. Dimension: Teamqualität

Referenzen

- [Pot-e120] A. Poth, J, Jacobsen, A. Riel, Systematic Agile Development in Regulated Environments, in: Proc. of Systems, Software and Services Process Improvement. EuroSPI 2020, CCIS 1251, https://doi.org/10.1007/978-3-030-56441-4_14
- [Pot-e20] A. Poth, M. Kottke, A. Riel, Agile Team Work Quality in the Context of Agile Transformations – A Case Study in Large-Scaling Environments, in: Proc. of Systems, Software and Services Process Improvement. EuroSPI 2020, CCIS 1251, https://doi.org/10.1007/978-3-030-56441-4_17
- [Pot-f19] A. Poth, M. Kottke, A. Riel, Scaling agile on large enterprise level, in: Proc. of the 2019 Federated Conf. on Computer Science and Information Systems, ACSIS, Vol. 18, http://dx.doi.org/10.15439/2019F150
- [Pot-f20] A. Poth, M. Kottke, A. Riel, Scaling agile on large enterprise level with self-service kits to support autonomous teams, in: Preproc. of the Federated Conf. on Computer Science and Information Systems, 2020, https://annals-csis.org/proceedings/2020/pliks/186.pdf
- [Pot-r20] A. Poth, A. Riel, Quality requirements elicitation by ideation of product quality risks with design thinking, in: Proc. of the 28th IEEE Int. Requirements Engineering Conference (RE'20), pp. 238-249, 2020; DOI 10.1109/RE48521.2020.00034
- [Pot-x120] A. Poth, M. Kottke, A. Riel, Evaluation of Agile Team Work Quality, in: Proc. of the 21st Int. Conf. on Agile Software Development, LNBIP 396, https://doi.org/10.1007/978-3-030-58858-8_11
- [Pot-x20] A. Poth, J. Jacobsen, A. Riel, A systematic approach to agile development in highly regulated environment, in: Proc. of the 21st Int. Conf. on Agile Software Development, LNBIP 396, https://doi.org/10.1007/978-3-030-58858-8_12

SSK genutzt werden und regt die Teams über Fragen zur Reflexion und Verbesserung an.

Es werden verschiedene Modelle aus der Team- und Organisationsentwicklung in aTWQ zusammengeführt und mit vorhanden Indikatoren der agilen Ansätze wie Scrum und SAFe korreliert. Neben der intrinsischen Teamentwicklung kann aTWQ auch zur strategischen Organisationsentwicklung genutzt werden. In der Group IT ist aTWQ hierfür zum Beispiel in den agilen Team-Review [Pot-f19] integriert.

Governance, Team und Produkt

Die enge Verzahnung von ACE und TQA ermöglicht es, ganzheitliche Ansätze bereitzustellen, die die Qualität und agile Transition gleichermaßen fördern und als Basis für die agile Skalierung dienen – auch in die Geschäftsbereiche hinein. **Abbildung 2** [Pot-f20] zeigt das generische Zusammenspiel von Organisation (Governance), Teams und deren Liefergegenstände (Produkte/Services) am Beispiel der Skalierung von Wissen mit dem SSK-Ansatz. Das ACE und TQA haben neben Liefergegenständen für die Teams und deren Produkte/Service auch Governance-Aufgaben wahrzunehmen. Dies ermöglicht es, immer mit einem Bein im Wertstrom zum Kunden zu stehen und so den Fokus auf Lean Governance zu wahren.

Die Digitalisierung von Produkten und Organisationen führt zur tieferen Integration von IT und Geschäftsbereichen für die operative Leistungserbringung vor Kunde. Ein typisches

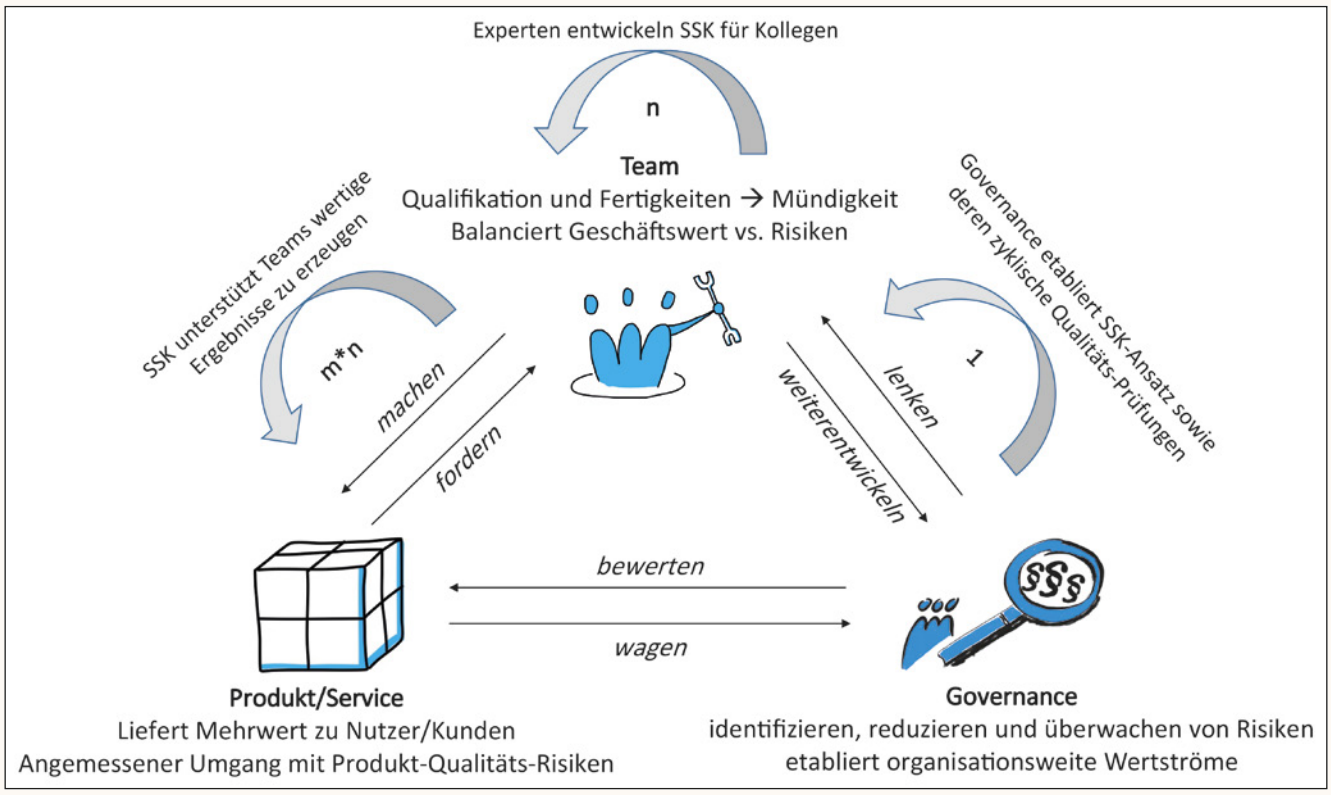


Abb. 2: Interaktion von Governance, Team und Produkt sowie ein Beispiel für Skalierung von SSKs

Beispiel aus der Automobilindustrie ist das Fahrzeug (etabliertes Produkt), welches immer mehr Online-Dienste (IT-Service) als Teil des Kundenerlebnisses integriert. Um dieses Kundenerlebnis bereitzustellen, werden die Fahrzeug-Produkte und IT-basierte Services wie beispielsweise Mehrwertdienste im Fahrzeug oder im Umfeld des Fahrzeugs zusammengeführt. TQA gestaltet mit der Fahrzeugentwicklung und dem IT-Betrieb die Absicherung ausgewählter Services. Hier werden individuelle und spezielle Absicherungsstrategien für die einzelnen Dienste designt und erbracht. Anforderungen aus der Fahrzeugwelt nach robusten Prozessen und stabilen Produkten in hoher Qualität treffen auf den Wunsch des Kunden nach kurzen Entwicklungszyklen und schnellen Feedbackschleifen – klassische Projektentwicklung und -absicherung trifft auf agile Methodik.

Dies muss kein Widerspruch sein, sondern entspricht dem Wunsch, das Beste aus beiden Welten zu vereinen. Schnelles Time-to-market in gewohnt hoher Qualität.

Ein Beispiel an dieser Nahtstelle sind die Fahrzeug-Online-Dienste. Diese haben ihre Benutzerschnittstelle in der Fahrzeugwelt und benötigen eine tiefe Integration in IT-Backend-Systeme für umfängliche Datenbereitstellung und Nutzung im Kontext des Fahrzeugs und der Nutzer. Das Absichern dieser Service-Ketten über verschiedene IT-Systeme hinweg ist komplex. Das ACE gestaltet und etabliert mit der Fahrzeugentwicklung agile Ansätze der Entwicklung von zukünftigen Diensten und deren Entstehungsprozessen mit dem Ziel, einfache und schlanke Services und Systemlandschaften zu etablieren.

Fazit

Diese intensive Zusammenarbeit ist ein Transformationsprozess, der zwar durch die Produkt/Service-Digitalisierung getrieben wird, aber weiteren Rahmenbedingungen unterliegt, wie unterschiedliche Releasefrequenzen zwischen IT-Komponenten und Fahrzeugkomponenten und verschiedenen Entwicklungsmodellen.

Ideen und Erfahrungen aus dem IT-Bereich und der agilen Entwicklung können auch für die Fachbereiche hilfreich sein, da sie vor ähnlichen Herausforderungen stehen. Hier wirken ACE und TQA, um ganzheitliche Lösungen zu gestalten und zu etablieren, die eine angemessene Balance bezüglich Geschwindigkeit und Qualität für ein optimales Kundenerlebnis finden.

Wenn es gelingt, eine ganzheitliche Sicht auf die agile Skalierung zu etablieren, so ist es möglich, Coaching-Aktivitäten auf die Bereiche zu fokussieren, die über Self-Services nicht oder noch nicht abgedeckt werden können.

Dieser Ansatz ermöglicht es, die Geschwindigkeit der Transition im Unternehmen nicht-linear mit der Coaching-Kapazität zu skalieren. In Zukunft wird die Zusammenarbeit von ACE und TQA stärker mit den Geschäftsbereichen erfolgen, um noch höher integrierte und generische Ansätze entwickeln zu können, um Wertströme nach dem agilen Ansatz nachhaltig zu etablieren.

Alexander Poth

alexander.poth@volkswagen.de

wirkt als IT-Qualitätsmanager mit ausgeprägtem agilem Mindset in der Group IT der Volkswagen AG. Seine aktuellen Schwerpunkte liegen in der Rolle Product Owner für das Quality innovation NETwork (QiNET) und Testing as a Service (TaaS) sowie der Unterstützung von Projekten, Programmen, Organisationseinheiten und Produkt/Service-Teams bei qualitätsbezogenen Themen.

Dr. Christian Heimann

christian.heimann@volkswagen.de

verantwortet den Bereich IT-Qualitätsmanagement und Open Source Compliance in der Group IT der Volkswagen AG. Nach seiner Promotion in Informatik und verschiedenen Managementaufgaben im IT-Bereich der Volkswagen AG leitet er den Bereich Test und Quality Assurance. Er ist ausgebildeter agiler Lotse.

Stefan Waschk

stefan.waschk@volkswagen.de

ist verantwortlich für das Agile Center of Excellence (ACE) der Group IT der Volkswagen AG und arbeitet aktiv am Ausbau der „Agile Community" in alle Bereiche des Automobilkonzerns. Mittlerweile erfüllen das ACE und die Agile Community einen offiziellen IT-Governance-Auftrag für mehr Agilität und unterstützen Agile Arbeitsweisen inzwischen im gesamten Unternehmen. Als Mitbegründer der DACH30-Gruppe treibt Stefan den Austausch zu funktionierenden Agilen Praktiken mit Agile Leads namhafter Großunternehmen an – für die nachhaltige Einführung neuer Arbeitsweisen im digitalen Wandel (www.next-level-working.com).

»Qualitätssicherung im Kontext grosser agiler Softwareentwicklungsprojekte«

Die Einbindung von Test und Qualitätssicherung (QS) in agile Frameworks war insbesondere zu Beginn des agilen Entwicklungstrends eine besondere Herausforderung, aufgrund organisatorischer Änderungen, vieler Anpassungen in den Arbeitsprozessen und kürzerer Entwicklungszyklen. Um diese Herausforderungen erfolgreich zu meistern, haben wir das House of Agile Testing (HoaT) als Quality Transformation Framework entwickelt und immer wieder erfolgreich angewendet.

Die Ursprünge des HoaT reichen ins Jahr 2010 zurück, als die „agile Reise" von Accenture bei der Bundesagentur für Arbeit (BA) begann. Ebenso wie sich dieses IT-Verfahren und die meisten IT-Abteilungen der Welt weiterentwickelt haben und neue beziehungsweise komplexere Herausforderungen dazu kamen, hat sich auch das HoaT weiterentwickelt. Dabei berücksichtigt das HoaT V2.0 insbesondere die Herausforderungen der Skalierung, die zunehmende Bedeutung des Lean Management, der kontinuierlichen Verbesserung und die permanente Weiterentwicklung der Mitarbeiter insbesondere auch nach der Transformation.

Im ersten Abschnitt des Artikels wird daher das theoretische Fundament gelegt und die Evolution des HoaT beschrieben, um dann im zweiten Teil den Bezug zum eingangs genannten Praxisprojekt und den dort gesammelten Erfahrungen herzustellen.

Das HoaT V2.0 als theoretisches Fundament

Im Rahmen einer agilen Transformation müssen wie bei jedem größeren Veränderungsprojekt initial einige Fragen beantwortet werden. Dies ist wichtig, um ein einheitliches Verständnis aller Beteiligten bezüglich der Transformation zu schaffen und somit auch die Grundlage für eine erfolgreiche Veränderung zu legen. Erfahrungsgemäß sollten mindestens die folgenden Fragen beantwortet werden:

- Was wollen wir erreichen? – Welches *geschäftspolitische Ziel* wird mit der agilen Transformation verfolgt?
- Welches agile *Framework* ist für unser Unternehmen am besten geeignet? – Damit verbunden ist auch die Frage, welche *Tools* wir benötigen.
- Was benötigen unsere *Mitarbeiter*, um die gesetzten Ziele und Erwartungen erreichen zu können?
- Was benötigt unsere Organisation (als Ganzes betrachtet), um die gesetzten Ziele und Erwartungen erreichen und dauerhaft in die *Organisationsstruktur* und -kultur aufnehmen zu können?

Die Beschreibung einer agilen Transformation aus Sicht des Tests war vor einigen Jahren noch die zentrale Herausforderung, das HoaT lieferte die Antwort darauf. Den klassischen Test gibt es nun großteils nicht

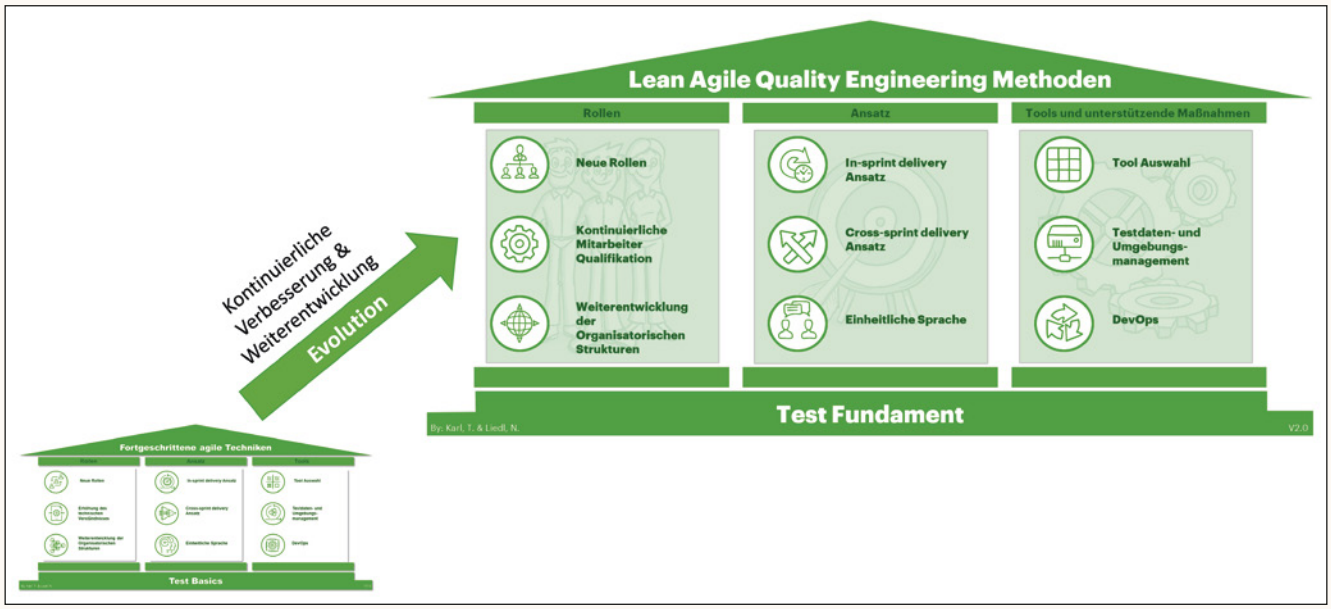


Abb. 1: Evolution des House of agile Testing

mehr, der Test hat sich zu einem proaktiveren Vorgehen, dem Quality Engineering, weiterentwickelt, dabei spielen auch neue Technologien wie KI und der zunehmende Einsatz von Robotic eine immer wichtigere Rolle.

Diese Veränderungen spiegeln sich nun auch im HoaT in der Version 2.0 wider. Diese Trends sowie unsere Praxiserfahrungen aus den letzten Jahren führten zu einer massiven Erweiterung der Inhalte im Dach des HoaT, sodass wir uns hier auch bewusst für ein Renaming entschieden haben. Darüber hinaus haben wir den Inhalt in Säule 1 „Rollen" stark erweitert und den Fokus deutlich stärker auf die kontinuierliche Mitarbeiterweiterentwicklung gelegt. Außerdem haben sich die Inhalte in Säule 3 „Tools" ebenso wie die Tools selbst stark weiterentwickelt. So bietet das HoaT V2.0 **(siehe Abbildung 1)** nun eine Orientierungshilfe, um alle wesentlichen Punkte im Bereich Quality Engineering im Auge zu behalten, insbesondere im Kontext von sehr großen skalierten IT-Entwicklungsvorhaben.

Das Fundament des Hauses bilden die *Test Basics*, also die Grundlagen des Testens und der Qualitätssicherung. Dies beinhaltet sowohl ein fundamentales Verständnis des Testprozesses als auch die methodischen Grundlagen, wie beispielsweise das Wissen über Testfallentwurfsmethoden, Review Techniken und Defekt-Management (vgl. ISTQB CTFL Syllabus).

Die *Säulen* des Hauses definieren die Arbeitsweise/Prozesse des Tests in einem agilen Umfeld. **Abbildung 1** stellt die Inhalte der drei Säulen eine Detailstufe tiefer dar. In weiteren Artikeln gehen wir detailliert auf die einzelnen Säulen ein, diese sind zum Teil erschienen:

- Tools und unterstützende Maßnahmen [KaLi18],
- Rollen [KaLi20],

oder erscheinen zukünftig in weiteren Artikeln:

- Ansatz,
- Lean Quality Engineering.

Nachstehend wird daher nur kurz auf die wesentlichen Inhalte der Säulen eingegangen. In der *1. Säule – „Rollen"* gilt es, Fragen bezüglich der künftig notwendigen Rollen zu beantworten. Dies hängt einerseits von der gewählten agilen Softwareentwicklungsbeziehungsweise Skalierungsmethode ab, andererseits aber auch von der Frage, ob gegebenenfalls Spezialrollen für den Test benötigt werden. Direkt damit verbunden ist die Frage nach dem notwendigen technischen Wissen und Verständnis in den einzelnen Rollen. Insbesondere ist hier auch die Frage zu klären, inwiefern und in welchem Umfang technisches Wissen auf- und ausgebaut werden muss, um das notwendige Skillset, das mit den jeweiligen Rollen verbunden ist, im Unternehmen sicherzustellen. Wenn die beiden vorangegangen Fragen geklärt sind, gilt es schließlich, die organisatorischen Strukturen weiterzuentwickeln und letztlich die neuen beziehungsweise angepassten Rollen im Organigramm der Organisation einzubauen und abzubilden.

Die *2. Säule – „Ansatz"* beinhaltet die Frage nach dem Lieferansatz und der damit direkt verbundenen Frage: „wann, was, getestet werden muss". Insbesondere bei der Einführung von Agilität auf großen Entwicklungsprojekten, die Teil von komplexen IT-Systemlandschaften sind, findet man in der Anfangsphase ein hybrides Vorgehen vor. Die Software wird dann oftmals in Sprints entwickelt, aber erst nach wenigen Monaten im Rahmen eines Releases live gesetzt. Bei dieser Vorgehensweise gilt es zu klären, welche Inhalte innerhalb der Sprints und welche in einem aus- und nachgelagerten Test geprüft werden müssen.

Eine weitere Möglichkeit besteht darin, dem Big-Bang-Vorgehen zu folgen. Im Rahmen einer Implementation Roadmap wird die gesamte Mannschaft im neuen Vorgehen geschult und dann direkt in allen Bereichen in das neue Vorgehen überführt. Es kommt am Ende darauf an, welches Vorgehen zu welchem Projektzeitpunkt sinnvoll ist. Gerade weil wir uns auf den Bereich der QS konzentrieren, stellt die Wahl des richtigen Ansatzes eine besondere Herausforderung dar, denn

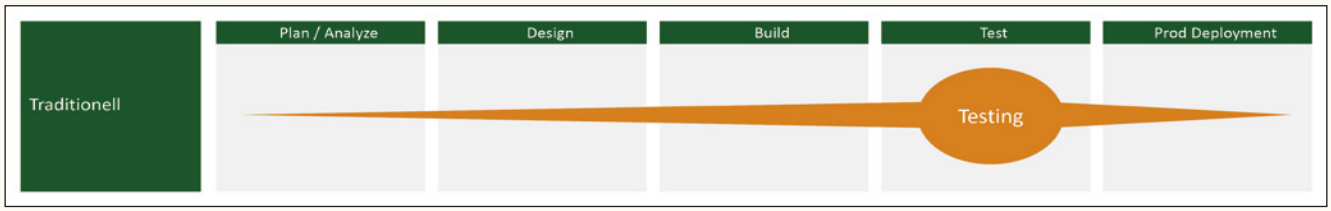


Abb. 2: Traditionelles Testvorgehen

oftmals ist es schwierig, aus diesem Bereich heraus Veränderungen zu motivieren.

Die *3. Säule – „Tools"* beinhaltet schließlich die technischen und prozessualen Fragen der Toolauswahl. Eine frühe Ausrichtung in Richtung DevSecOps kann die nötige Entwicklung in die zukünftige Automatisierung des Projekts fördern. Es gilt zu klären, welche Testautomatisierungstools den gewählten Entwicklungsansatz am besten unterstützen, ob sie in einer zukünftigen Toolchain zusammenpassen und sich immer weiter kombinieren lassen. Genauso gilt es hier, den übergreifende Architectural Runway im Blick zu behalten. Jegliche Weiterentwicklung wirkt sich schließlich auf die Qualitätsmaßnahmen aus, daher sollen diese auch Wirkung auf die zukünftige Ausgestaltung der Architekturbestandteile haben. Darüber hinaus müssen Testumgebungen und der Umgang mit Testdaten frühzeitig in die Planung aufgenommen und schließlich angestrebt werden. Ziel muss sein, die einzelnen Tools und Umgebungen in eine End-to-End-Tool-Pipeline einzubetten.

Das *Dach* des Hauses bilden die *Lean Quality Engineering*-Methoden. Hier sind schließlich die komplexeren Methoden und Techniken anzusiedeln, die sich als Teil oder rund um die agile Softwareentwicklung herausgebildet haben. Dazu zählen beispielsweise fortgeschrittene Testmethoden wie das Behaviour-Driven Development [Fer14], proaktive Ansätze des Quality Engineering, zum Beispiel Performance Engineering statt nur Performance Test, aber auch der gesamte Bereich des Lean Quality Engineering.

Im lean-agilen Qualitätsmanagement lassen sich durch die Kombination von agilen, lean und Six Sigma Methoden beeindruckende Resultate erzielen. Dabei können wir in der Softwareentwicklung durch den Transfer und die Adaption von Good Practices aus der Fertigungsindustrie extrem viel lernen. Das Spektrum reicht von einer kundenzentrierten Arbeitsweise auf Basis von Value Streams, über die Visualisierung und die kontinuierliche Verbesserung bis hin zu einem dedizierten Prozessfokus und der Vermeidung von Verschwendung. Insbesondere die Themenbereiche Prozessqualität und Vermeidung von Verschwendung sind im skalierten Umfeld von besonderer Bedeutung. Dabei gilt es zu berücksichtigen, dass ein simples Kopieren der Methoden aufgrund der großen Unterschiede zwischen Massenfertigung am Fließband und kreativer Wissensarbeit an der CI/CD-Pipeline nicht sinnvoll ist.

Bei der BA werden im Projekt virtueller Arbeitsmarkt (VAM) zwei Anwendungen entwickelt. Auf der einen Seite die *Jobbörse*, ein externes Portal, über das Arbeitgeber und Arbeitssuchende Profile einstellen und verwalten können. Auf dieser kann man Suchen ausführen und selbstständig nach Arbeitsstellen oder Arbeitskräften suchen. Auf der anderen Seite steht die Anwendung *VERBIS*. Diese interne Software wird in den Jobcentern und Arbeitsagenturen verwendet, um Arbeitssuchende und Arbeitgeber zu vermitteln und zu beraten. Über 100.000 Mitarbeiter in den Arbeitsagenturen erzeugt so ca. 40 Mio. Seitenaufrufe pro Tag.

Derzeit werden die Anwendungen in 16 agilen Teams entwickelt, die seit 2011 schrittweise aufgebaut wurden. Eine der größten Herausforderungen ist es, dass der VAM im Umfeld der IT-Systemlandschaft mit 19 internen sowie externen Schnittstellensystemen im Verbund steht. Diese werden zu großen Teilen im Wasserfall entwickelt. Das führt dazu, dass die Entwicklung im hybriden Umfeld stattfinden muss, welches mit drei Releases im Jahr keine reine Agilität zulässt.

Kasten 1: Das Umfeld virtueller Arbeitsmarkt (VAM)

Neben der testspezifischen Betrachtung der Inhalte stellen ein auf die agile Transformation angepasstes Veränderungsmanagement und eine gezielte Organisationsentwicklung einen weiteren Erfolgsfaktor dar, beides sind integrale Bestandteile des HoaT V2.0 [Kai17].

Praxisprojekt „virtueller Arbeitsmarkt"

Nachfolgend wird nun die Umsetzung des beschriebenen Konzepts bei der Bundesagentur für Arbeit (BA) beschrieben.

Seit dem Jahr 2010 begleitet Accenture gemeinsam mit dem Kunden die agile Reise. Man sammelte viele Erfahrungen, die im weiteren Verlauf des Artikels näher betrachtet werden sollen. Die vorgestellte Theorie wird anhand von Beispielen aus dem Praxisprojekt vorgestellt.

Im Rahmen der bekannten agilen Skalierungsframeworks wie SAFe© [Chr17], NEXUS© [Bit18] oder LeSS© [Lar17] werden viele der im Folgenden beschriebenen Best Practices eingesetzt. Zum Start der Transformation in der damaligen Wasserfallwelt hatte der Test seinen etablierten Stand. Der Change-Prozess hin zum agilen Vorgehen schürte bei den Testern Ängste und Ungewissheiten. Unklare Rollen, Angst vor Auflösung des Testteams, auch die Zusammenarbeit zwischen Entwicklung und Test und der damit verbundenen Auflösung der Silos bargen die Herausforderung, Tester dazu zu motivieren, in die agile Welt zu transferieren. Das Skillprofil des Testers wandelt sich seit den Anfängen der Agilisierung stark in Richtung eines agilen Quality Engineer. Der damit verbundene Mindchange, der im ersten Teil theoretisch beschrieben ist, stellt in der Praxis die größte Herausforderung dar. **Abbildung 2** zeigt den traditionellen Ansatz des Testings sinnbildlich. Die Testaktivitäten konzentrieren sich auf die Testphase im Softwareentwicklungszyklus.

Referenzen

- [Bit18] K. Bittner, P. Kong, D. West, Mit dem Nexus Framework Scrum skalieren – Kontinuierliche Bereitstellung eines integrierten Produkts mit mehreren Scrum-Teams, dpunkt.verlag, 2018
- [Chr17] M. Christoph, SAFe – Das Scaled Agile Framework. Lean und Agile in großen Unternehmen skalieren, dpunkt.verlag, 2017
- [Coh09] M. Cohn, Succeeding with Agile, Software Development Using Scrum, Pearson Education, 2009
- [Fer14] J. Ferguson, BDD in Action, Manning, 2014
- [Kai17] S. Kainzbauer, W. Brandhuber, Eine Einführung in die Agile Organisationsentwicklung, Sigs Datacom eBook, 2017, siehe: https://www.sigs-datacom.de/wissen/ebooks/wissen-titel/agile-organisationsentwicklung.html
- [KaLi18] Th. Karl, N. Liedl, Qualitätssicherung im Kontext von großen agilen Softwareentwicklungsprojekten, in: German Testing Magazin, 2/2018, siehe: https://www.accenture.com/t00010101T000000Z__w__/de-de/_acnmedia/PDF-94/Accenture-Qualitatssicherung.pdf#zoom=50
- [KaLi20] Th. Karl, N. Liedl, House of Agile Testing – Säule 1: Rollen, in: German Testing Magazin, 2/2020
- [Lar17] C. Larmann, B. Vodde, Large-Scale Scrum – Scrum erfolgreich skalieren mit LeSS, dpunkt.verlag, 2017

Im Folgenden werden exemplarische Praxisbeispiele für die 3 Säulen des HoaT aus dem Beispiel bei der BA beschrieben.

Rollen

Säule 1 im HoaT stellt eine große Herausforderung dar, weil es sich erstens um einen sehr langwierigen Prozess handelt, Rollen zu definieren und abzustimmen, und sich zweitens die Rahmenbedingungen durch immer wiederkehrende Teamveränderungen ändern. Eine Maßnahme, die uns bei dem Beispielprojekt längerfristig begleitet und einen gewissen Rahmen für die Weiterentwicklung verschafft hat, ist das *Role Model Canvas* (RMC). Im Rahmen dessen werden die Verantwortlichkeiten auf den verschiedensten Ebenen abgebildet und für die jeweiligen Personengruppen organisiert. Sei es innerhalb der Teams, über Entwicklungsteams oder zuliefernde Teams hinweg oder auch vom Management in die Teams. Das RMC stellt das aktuell vorherrschende Rollenverständnis und im Ergebnis dann das zukünftige Zielbild dar. Durch die klare Kommunikation über die gesamte Mannschaft kann man schließlich alle weiteren organisatorischen Weiterentwicklungen daraufhin ausrichten.

Für die Ausgestaltung der neu gewonnenen geteilten Verantwortungen in den Testrollen wurde eine *Test Community of Practice* (Test-CoP) eingeführt. Diese entwickelte sich zu einem der zentralen Bestandteile, die zum nachhaltigen Erfolg geführt hat. Im Rahmen dieser CoP werden die Teams aktiv in die Weiterentwicklung des Testvorgehens und Methodiken eingebunden und damit Anregungen und Ideen teamübergreifend diskutiert, verfeinert und eingeführt. Damit erreicht man, dass gute Ansätze aus den Teams auf dem gesamten Projekt Anwendung finden und ein übergreifendes Qualitätsverständnis und Qualitätsniveau gehalten werden kann.

Ansatz

Als Beispiel aus der Säule „Ansatz" ist die *Strukturierung der organisatorischen Teamstruktur und deren Aufgabenbereiche* zu nennen. Diese Fragestellungen gilt es, im Laufe der Organisationsentwicklung zu klären. Über die Jahre hinweg wurde das klassische Testteam immer weiter in die agilen Teams überführt. Der Effekt war der, dass Testaktivitäten relativ schlecht abgestimmt einerseits in den agilen Teams (ET) und andererseits im Testteam (TEST) durchgeführt wurden. In einem weiteren Schritt galt es, klare Strukturen vorzugeben, wo und wann welche Testaktivitäten durchgeführt werden. Dem Team muss bewusst werden, dass es allein für die Qualität seiner Anwendungsbereiche verantwortlich ist und kein weiteres Team ein „Sicherheitsnetz aufspannen wird".

Unter diesen Voraussetzungen entstand ein selbstverantwortetes Qualitätsverständnis

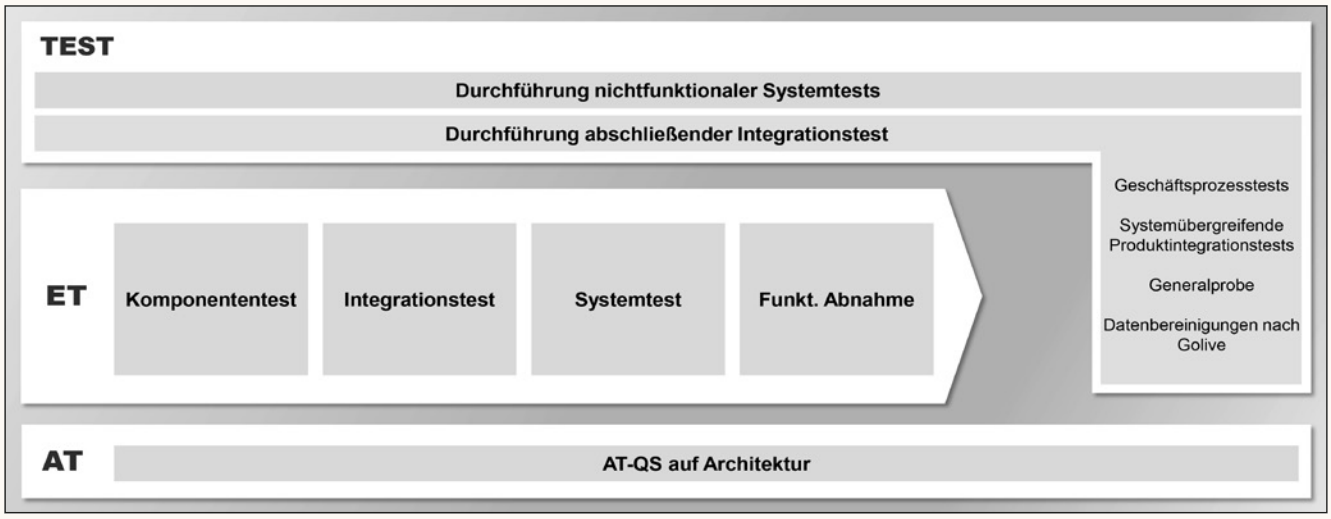


Abb. 3: Organisatorische Teamstruktur und Aufgabenbereiche

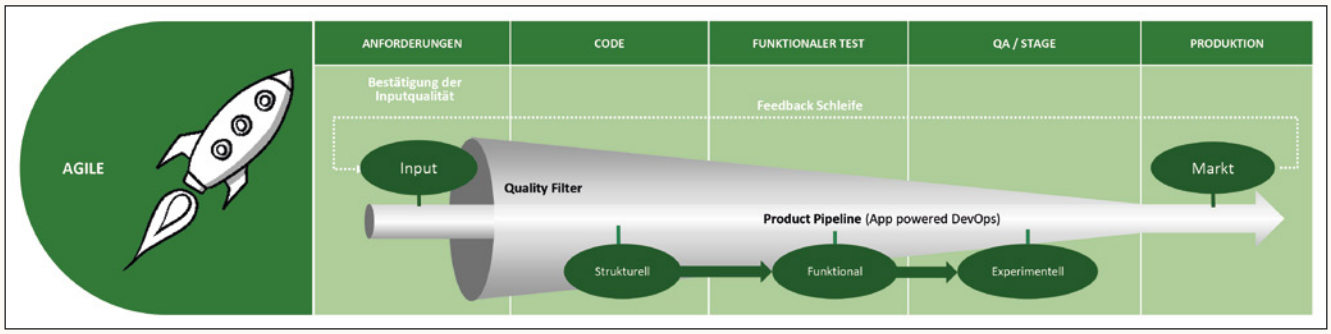


Abb. 4: Zielbild „shift left"

der Teams, welches die Identifikation mit dem Produkt stärkt. Das Testteam ist weiterhin als wichtiges QS-Bindeglied zum hybriden Vorgehen der BA nötig. Es stellt die Verbindung zu den Schnittstellenpartnern her und kümmert sich um die Release abschließende QS. Dazu werden abschließende QS-Maßnahmen durchgeführt, die erst kurz vor dem Go-Live relevant werden. Die Teams können sich dadurch besser auf die Weiterentwicklung des Folgereleases konzentrieren. **Abbildung 3** stellt die Aufteilung zwischen den Entwicklungsteams (ET) und dem Testteam und ihren zeitlichen Ablauf dar.

Tools

Ein Beispiel der Säule Tools aus dem HoaT ist die Umsetzung der aus der Literatur bekannten *Testautomatisierungs-Pyramide* [Coh09] und der damit verbundenen Umsetzung vollumfänglicher Testautomatisierung. Wie soll es möglich sein, alle Testaktivitäten aus der „alten Welt" in nur einem Sprint für das „potentially shippable product" unterzubekommen? Viele Projekte kennen sowohl das Problem als auch die Pyramide. Die zentrale Herausforderung ist jedoch die Umsetzung.

Das Prinzip des „shift left" **(siehe Abb. 4)** gibt klare Strukturen vor, wie Testaktivitäten sinnvoll aneinandergereiht werden, um mit einer vorgegebenen Sprintlänge alle nötigen Testaktivitäten umsetzen zu können. Bezug nehmend auf **Abbildung 2** werden damit die Testaktivitäten zum frühestmöglichen Zeitpunkt durchgeführt. Umsetzbar ist das durch ein klares Engagement zur Testautomatisierung. Auf dem Beispielprojekt wurde so über die Jahre eine Testautomatisierungsquote von 95 Prozent über alle Teststufen erreicht.

Fazit

Die genannten Beispiele geben einen Einblick, welche Bestandteile in den vielen Jahren in die Qualität eingezahlt haben. Aus Sicht von Accenture ist der VAM eines der am weitesten entwickelten agilen Projekte. Der nächste logische Schritt ist, eine konsequente Umsetzung des projektübergreifenden *Lean Agile Quality Management*-Ansatzes zu verfolgen. Dafür benötigt man eine übergreifende Vision und eine damit einhergehende stetige Weiterentwicklung der Vision und der daraus abgeleiteten Mission. Die genannten Werkzeuge werden auch weiterhin dazu beitragen, wobei die Marschrichtung regelmäßig überarbeitet werden muss.

Genauso muss das HoaT-Konzept stetig kritisch betrachtet und weiterentwickelt werden. Verschiedenste Projekte ermöglichen es uns, am „Puls der Zeit" diese Weiterentwicklungen in der Praxis stetig voranzutreiben und so das HoaT kontinuierlich weiterzuentwickeln, um neuste Trends einzubauen.

Nico Liedl

nico.liedl@accenture.com

ist Quality Engineer und Advisor mit Schwerpunkt auf agilem Qualitätsmanagement, Testautomatisierung, Prozessverbesserung und organisatorischem Wandel. Er hat Erfahrungen bei verschiedenen komplexen Großprojekten gesammelt. Als Teammitglied, Scrum Master, Teststratege, Releasemanager und Trainer hat er praktische Erfahrung und theoretisches Fachwissen zu bieten und möchte sie mit einem breiten Publikum teilen.

Thomas Karl

thomas.karl@accenture.com

ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

»Ein arbeitsorganisatorischer Ansatz in der VUKA-Welt«

Seit die agile Transformation in den meisten Softwareentwicklungsprojekten angekommen ist, haben sich Firmen darauf fokussiert, agile Prinzipien mithilfe von etablierten Softwareentwicklungsmethoden wie Scrum, SAFe oder Kanban zu implementieren. All diese Frameworks konzentrieren sich auf die Zusammenarbeit der Menschen untereinander, den Arbeitsprozess und den Mindchange in der Organisation. Auf das Thema Qualität wird meist nur implizit Bezug genommen. Genau dieser Bereich der ganzheitlichen Qualitätsoptimierung und speziell die notwendige Neuorientierung in Bezug auf „Menschen und Rollen" soll in diesem Artikel genauer betrachtet werden.

Um in einem agilen Umfeld grundsätzliche Leitlinien zu etablieren, haben wir das in **Abbildung 1** dargestellte „House of agile Testing" (im Folgenden HoaT) entwickelt. Es setzt sich aus fünf Teilbereichen zusammen. Die Basis bilden die *Test Basics*. Ohne diese methodischen Ansätze und die damit verbundenen Arbeitsweisen können in der Qualitätssicherung nicht die gewünschten Ergebnisse erzielt werden. Darauf bauen die drei Säulen *Menschen und Rollen*, *Ansatz* und *Tools* auf. Wenn diese drei Säulen aufgebaut sind und eine gewisse Reife aufweisen, wird das Haus mit *Lean Agile Quality Engineering*-Methoden vervollständigt, um auch komplexere Methoden und Techniken einzuführen.

Zuletzt sind wir in unserem Artikel „Qualitätssicherung im Kontext von großen agilen Softwareentwicklungsprojekten" [KaLi18] und in unseren zahlreichen Messevorträgen wie beispielsweise dem German Testing Day und der OOP auf das HoaT eingegangen. In einem weiteren Artikel haben wir uns speziell auf den effizienten Einsatz von Tools und der Testautomatisierung konzentriert [KaLi20]. Diese erfolgreiche Reihe möchten wir hier fortsetzen und uns mit den Herausforderungen und Belangen der ersten Säule beschäftigen.

Überblick über die drei Säulen des HoaT

Der Schwerpunkt unseres Artikels liegt damit auf der *Säule 1* „Menschen und Rollen". In dieser Säule gilt es, Fragen bezüglich der künftig notwendigen Rollen zu beantworten. Die Ausgestaltung hängt von der gewählten agilen Softwareentwicklungs- beziehungsweise Skalierungsmethode ab und auch davon, ob Spezialrollen für den Test benötigt werden. Direkt damit verbunden ist die Frage nach dem notwendigen technischen Wissen und Verständnis in den einzelnen agilen Rollen. Insbesondere ist zu klären, in welchem Umfang und welcher Tiefe technisches Wissen auf- und ausgebaut werden muss. Erst nach dieser Klärung werden die organisatorischen Strukturen weiterentwickelt und letztlich die neuen oder angepassten Rollen in der Organisation etabliert.

Die *Säule 2* „Ansatz" beinhaltet die Frage nach dem Testansatz, der sich aus dem gewählten Liefervorgehen ableitet. Insbesondere bei der Einführung von Agilität in Großprojekten, die Teil von komplexen

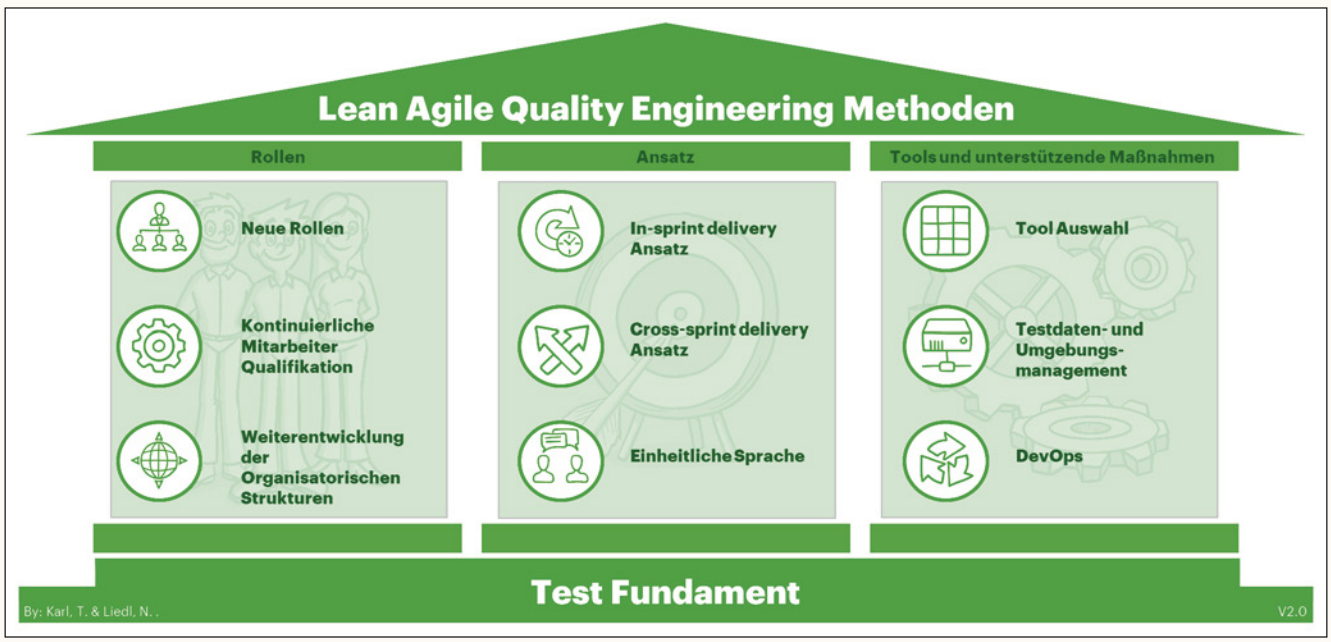


Abb. 1: „House of Agile Testing"

IT-Systemlandschaften sind, wird zu Beginn einer agilen Transformation ein hybrider Ansatz verfolgt. Die Software wird dann oftmals in Sprints entwickelt, aber erst nach mehreren Monaten im Rahmen eines „normalen" Release live gesetzt. Bei solchen Vorgehensweisen ist zu klären, welche Inhalte innerhalb der Sprints getestet werden können und welche Teile der Software nachgelagert geprüft werden müssen.

Die letzte *Säule 3* beinhaltet die technischen und prozessualen Fragen der Toolauswahl. Es ist zu klären, welche Testautomatisierungstools am besten den gewählten Entwicklungsansatz unterstützen und ob bestehende eigenentwickelte Testautomatisierungstools weiterverwendet werden können. Gleichzeitig muss betrachtet werden, welche und wie viele Testumgebungen benötigt und wie diese mit Testdaten versorgt werden. Für ein effizientes Liefer- und Testvorgehen ist dann insbesondere die Einbettung der einzelnen Tools und Umgebungen in eine End-to-End-Tool-Pipeline zu beachten.

Säule 1 „Menschen und Rollen" – Ein arbeitsorganisatorischer Ansatz in der VUKA-Welt

Wir leben in einer Welt, die sich ständig verändert sowohl im Kleinen als auch im Großen, für den Bereich der Softwareentwicklung trifft dies noch mehr zu als für viele andere. In der Literatur hat sich der Begriff VUKA-Welt dafür etabliert. VUKA steht für *V*olatilität, *U*nsicherheit, *K*omplexität und *A*mbiguität. Agile Arbeitsweisen werden oftmals als pauschale Musterlösung für die Herausforderungen der heutigen VUKA-Welt genannt. Aber auch hier gilt, „One fits all"-Lösungen passen nicht in diese Zeit, wie sie aber auch noch nie für die Arbeit mit Menschen gepasst haben. Auch im „Agilen Modus" ist es entscheidend, befriedigende Lösungen hinsichtlich der Strukturierung der Belegschaft, der Motivation der Mitarbeiter, der langfristigen Personalentwicklung und in Zeiten des Fachkräftemangels nicht zuletzt der Mitarbeiterbindung zu finden.

Themen, die klassischerweise bei der Personalabteilung (HR) angesiedelt sind, müssen im Kontext einer agilen Transformation zwingend mitberücksichtigt werden. Eine enge Kooperation zwischen der Personalabteilung und beispielsweise dem Testmanager mit Personalverantwortung ist also nötig. Daher ist die Säule 1 ein ganz zentraler Erfolgsfaktor der agilen Transformation, denn diese Art von Änderungen in der Organisation sowie im Skill- und Mindset der Mitarbeiter bedarf eines organisatorischen Change.

Wir sehen innerhalb der Säule 1 dabei drei wesentliche Bereiche, erstens das neue Rollenmodell, zweitens die kontinuierliche Mitarbeiterqualifikation und drittens die zugehörige Weiterentwicklung der organisatorischen Strukturen. Diese drei Kernbereiche werden nachfolgend näher betrachtet.

Neue Rollen

Die Definition der Rollen über das gesamte Projekt hinweg wird maßgeblich dadurch entschieden, welches Framework (beispielsweise SAFe, SCRUM, Nexus, …) eingesetzt wird. Diese geben ein gewisses Setup vor und definieren im Vornherein, welche Rollen empfohlen werden und in Teilen auch wie diese ausgestaltet sind. In den wenigsten Fällen geht es darum, ein gesamtes neues Team oder gar die ganze Mannschaft neu aufzubauen. Meist werden die Mitarbeiter initial aus vorhandenen Teams zusammengestellt und teilweise in das agile Setup überführt. Dabei ist es wichtig von der ersten Kommunikation an, die Leute möglichst aktiv mit einzubeziehen. Gegebenenfalls ist es sinnvoll, konkrete Wünsche einzufordern, um den Leuten klar zu machen, dass sie gefragt sind, wo sie ihre Stärken und Schwächen sehen und wohin sie sich entwickeln möchten.

Die Besetzung der Teams, aber auch die Schaffung von neuen Rollen ist im Folgenden einer der wichtigen Erfolgsfaktoren. Die Teams sich selbst zu überlassen und ihre Qualitätsstrukturen vollständig allein zu strukturieren, mag sich in der Anfangszeit besonders „agile" anhören. Nur leider kommt es spätestens bei der Skalierung dieses Ansatzes im Rahmen der späten Integration des Gesamtsystems dazu, dass der Bedarf besteht, eine übergreifende Struktur zu haben. Nur eine gemeinsame Vision mit gemeinsamen Zielen kann die gesamte Organisation auf einen Nenner bringen. Die Rolle eines *Lead Quality Engineer*, der als Architekt der Gesamtstruktur gesehen wird, kann dieses Ziel anstreben und die Mannschaft dahingehend coachen.

Innerhalb der Teams ist es genauso nötig, eine Position zu definieren, die den Qualitätsmaßnahmen im Team verpflichtet ist und das Team in dieser Hinsicht challenged. Wir nennen diese Rolle *Quality Engineers*, welche eher technisch, als SDET (Software developing Engineer in Test) oder eher fachlich, als In-Sprint Quality Spezialist oder UAT-Spezialist definiert werden. Es ist sinnvoll, eine Person mit Spezialwissen im Team zu haben, die optimalerweise in beiden Bereichen Stärken hat. Im Rahmen des T-shaped-Modells lässt sich über die Projektlaufzeit ein Fähigkeitsprofil über das Team hinweg darstellen. Mit diesem Wissen kann sich die Teamentwicklung immer weiter an das individuelle Ideal annähern.

Zusätzlich müssen alle Beteiligten regelmäßig abgeholt und darüber am Veränderungsprozess beteiligt werden. Nur so werden neue Rollen zum Leben erweckt und nachhaltig weiterentwickelt. Wichtig ist auch, dass man eine solche Struktur nicht an Tag 0 einführt. Auch hier ist es nötig, iterativ vorzugehen und schrittweise eine Maßnahme nach der anderen intensiv anzugehen. Begleitet von einem zielführenden Changemanagement-Prozess wird über einen längeren Zeitraum hinweg eine Qualitätsstruktur entstehen, die sich mit den entsprechenden Spezialisten selbst trägt und weiterentwickelt.

Abbildung 2 zeigt beispielhaft verschiedenste Rollen, die innerhalb sowie außer-

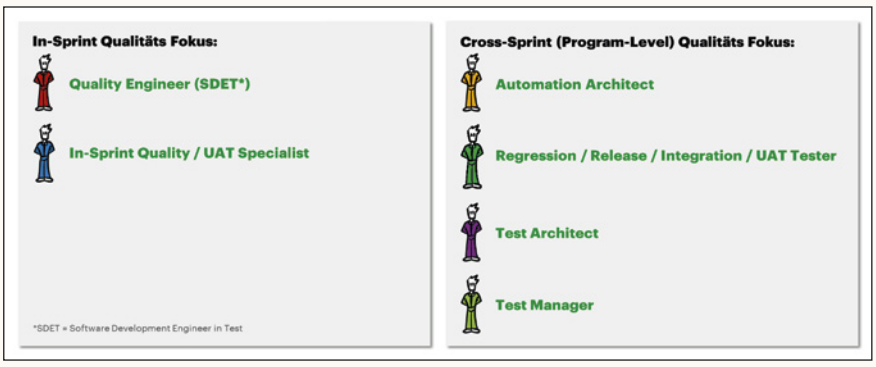


Abb. 2: Unterschiedliche Rollen im Umfeld von Quality Engineering

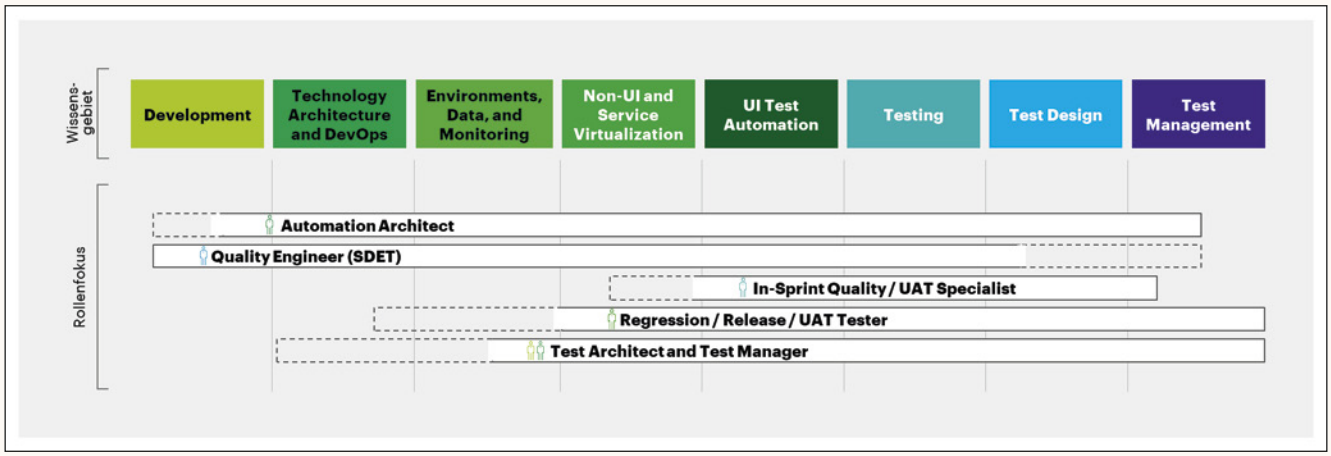


Abb. 3: Verschiedenste Wissensgebiete als Teil des T-shaped-Modells und der dazugehörige Fokus der beschriebenen Rollen

halb des Sprints Qualitäts-Ownership übernehmen müssen und unterschiedlichste Fähigkeiten mitbringen. Diese müssen nicht zwingend von einzelnen Personen oder Personengruppen ausgefüllt werden. Es ist genauso möglich, dass mehrere Rollen von einzelnen Personen ausgefüllt werden. Die Rollen mit „In-Sprint Qualitäts-Fokus" haben wir bereits eingangs beleuchtet. Je nach Ausbildung, Projektsetup und Relevanz empfehlen wir, zusätzlich zu betrachten, welche Rollen mit „Cross-Sprint Qualitäts-Fokus" nötig sind oder mit der Zeit nötig werden. Dabei geht es darum, Spezialrollen oder übergreifende Fähigkeiten auszulagern, um eine gewisse Ausrichtung über das Gesamtprojekt zu erreichen. Dieser Bereich wird im Abschnitt „Weiterentwicklung der organisatorischen Strukturen" genauer betrachtet.

Abbildung 3 zeigt verschiedene klassische Wissensgebiete und den jeweiligen Fokus, den die Rollen ausfüllen können. Damit wird auch sichtbar, welche Themen bei den QS-Rollen liegen können. Der Idee des T-shaped-Skill-Modells folgend, sollte entsprechend der Abbildung ein Basiswissen in den abgebildeten Kategorien vorliegen und je nach Rolle ein dazu passendes Expertenwissen im Kernaufgabengebiet.

Kontinuierliche Mitarbeiter-Qualifikation

Die kontinuierliche Weiterentwicklung der Mitarbeiter ist ein zentraler Erfolgsfaktor für Unternehmen in der heutigen Zeit und gewinnt insbesondere in der Wissensarbeit weiter stark an Bedeutung. Es gibt eine Vielzahl von unterschiedlichsten Modellen, Vorgehensweisen und good Practices in diesem Bereich.

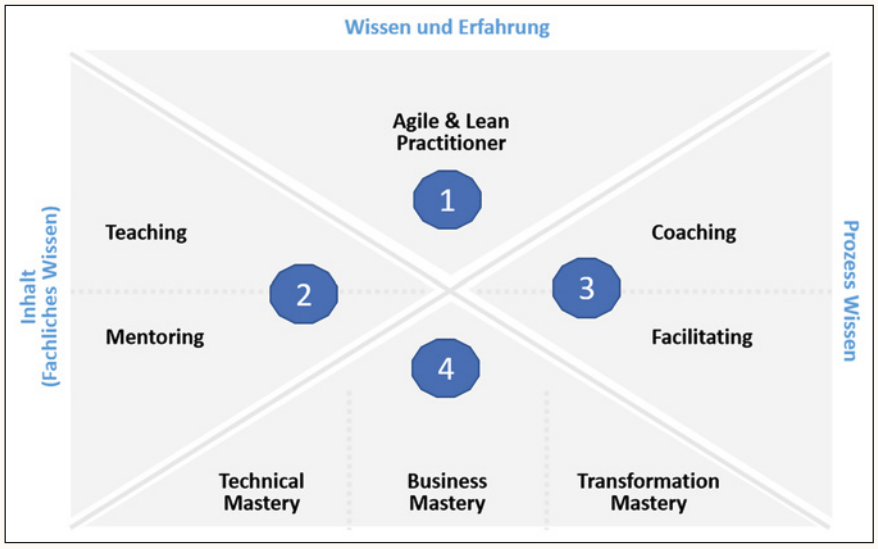


Abb. 4: Agile Coaching Competency Framework [ACI]

Nachfolgend wird exemplarisch ein Ansatz vorgestellt, den wir für pragmatisch und relativ unkompliziert in der Anwendung halten und den wir bereits mehrfach erfolgreich in Transformationsprojekten eingesetzt haben. Dabei handelt es sich um das „Agile Coaching Competency Framework" des Agile Coaching Institute [ACI]. Einen vergleichbaren Ansatz stellt die „Skill- bzw. Teamkompetenzmatrix" aus der Management 3.0-Methoden-Werkzeugkiste dar, die wir ebenfalls bereits mehrfach erfolgreich eingesetzt haben.

Es sei noch daraufhin gewiesen, dass sich beide Modelle ausschließlich auf die Entwicklung von Wissen und Fähigkeiten fokussieren, die Karriereentwicklung wird hier nicht betrachtet. Eine enge Kooperation mit der Personalabteilung, um die unterschiedlichen Fähigkeiten und Rollen auf das Karrieremodell und Incentive System zu mappen, ist zwingend notwendig.

Das *Agile Coaching Competency Framework*, welches in **Abbildung 4** dargestellt ist, ist originär dazu gedacht, die Kernfähigkeiten von gutem agilem Coaching darzustellen und zu clustern. Dabei wird explizit darauf hingewiesen, dass ein Coach nicht in allen Bereichen über Expertenwissen verfügen kann. Innerhalb eines agilen Teams sollten aber alle Bereiche abgedeckt sein. Das Framework ist in vier Sektoren aufgeteilt, um Wissen und Erfahrungen zu clustern:

- Sektor 1: „Agile & Lean Practitioner" repräsentiert das Wissen und Erfahrung hinsichtlich lean-agiler Methoden aller Art.

- Sektor 2 ist unterteilt in die beiden Blöcke „Teaching" und „Mentoring". Beides sind Methoden, um Inhalte – sprich fachliches Wissen – zu vermitteln.
- Auf der gegenüberliegenden Seite ist im Sektor 3 das Prozesswissen abgebildet mit der Unterteilung in „Coaching" und „Facilitating".
- In Sektor 4 werden schließlich die drei Bereiche „Technical Mastery", „Business Mastery" und „Transformation Mastery" unterschieden.

Dabei sind unter anderem folgende Fragen relevant:

- *Technical Mastery:* Verfügt eine Person oder das Team über ausreichend technisches Wissen, beispielsweise Front- und Backend-Programmierkenntnisse oder Performanz- und Securitytesting-Wissen?
- *Business Mastery:* Ist das fachliche Wissen über die Geschäftsabläufe und das fachliche Produktwissen in ausreichendem Maße vorhanden?
- *Transformation Mastery:* Wie ausgeprägt ist das Wissen hinsichtlich Change Management, Kommunikation usw.

Basierend auf unserer Praxiserfahrung hat es sich bewährt, das Wissensniveau in jedem der Bereiche von Abbildung 4 zwischen 1 und 10 zu bewerten[1]. Die Bewertung wird in regelmäßigen Abständen wiederholt, die Ziele des Mitarbeiters und der Organisation jeweils dagegengehalten. So kann man schnell einen einfachen Weiterentwicklungsplan erstellen und die Fortschritte sichtbar machen, denn Fortbildungsmaßnahmen unterschiedlichster Art lassen sich auch hervorragend den jeweiligen Bereichen zuordnen.

Das „Agile Coaching Competency Framework" wird also in zweierlei Hinsicht verwendet. Zum einen, um die verschiedenen Fähigkeitsniveaus der Teammitglieder in den unterschiedlichen Bereichen zu erfassen und dieses einerseits als Ausgangspunkt für die individuelle Weiterentwicklung der Mitarbeiter zu verwenden. Andererseits aber auch, um den Fähigkeitenpool des Teams als Gesamtes weiterzuentwickeln und dadurch sicherzustellen, dass das Team über Expertise auf dem jeweils benötigten Niveau verfügt. **Abbildung 5** gibt einen Eindruck davon, wie wir im Rahmen eines Teamworkshops zu Beginn einer Agilen Transformation dieses Konzept im Rahmen einer Teamaufstellung anwenden.

Zum anderen dient das Framework aber natürlich auch dazu, die Fähigkeiten der Coaches zu erfassen und so ein optimales fachliches Mapping zwischen Person/Team und Coach sicherzustellen. Natürlich muss zwingend darauf geachtet werden, dass der Faktor Mensch, also die zwischenmenschliche Verbindung zwischen Coach und Person/Team, auch passt.

Wichtig ist anzumerken, dass die kontinuierliche Mitarbeiterweiterentwicklung immer maßgeschneidert auf die jeweilige Unternehmenssituation und Personalausstattung, im engen Schulterschluss zwischen der Personal- und den Fachabteilungen stattfindet.

Weiterentwicklung der organisatorischen Strukturen

Je größer die Organisation ist und je mehr Teams oder gar Programmteile in einem Projekt in Verbindung stehen, umso wichtiger wird es auch, die organisatorischen Strukturen aktiv aus der QS-Brille zu entwickeln. Das ist immer abhängig von der jeweiligen Projektsituation.

Folgende Aspekte können hier Einfluss nehmen:

- Verteilung der Teams (inhouse, nearshore, offshore),
- Fähigkeiten in den Teams/Komplexität von Tools,
- Anzahl verwendeter Architekturen (Web, Rich Client, Microservices, …),
- Lieferart (CI/CD, Release, Hybride Modelle, …).

Die Frage ist immer, was möchte man wie erreichen oder abbilden. Normalerweise muss man das auch erst im Laufe der Zeit lernen und sich dann aktiv in den Bereich entwickeln, der am besten zur Gesamtsituation passt. Eine regelmäßige Überprüfung des eingeschlagenen Wegs ist hier zu empfehlen.

Das Modell „Role Model Canvas" (RMC) kann zusätzlichen Input generieren, der einen besseren Einblick in die Gesamtorganisation ermöglicht. Mithilfe dessen wird die Verteilung von Verantwortlichkeiten und Aufgaben über das Projekt sichtbar. In unserem Verständnis sollte man dieses Modell um eine detaillierte

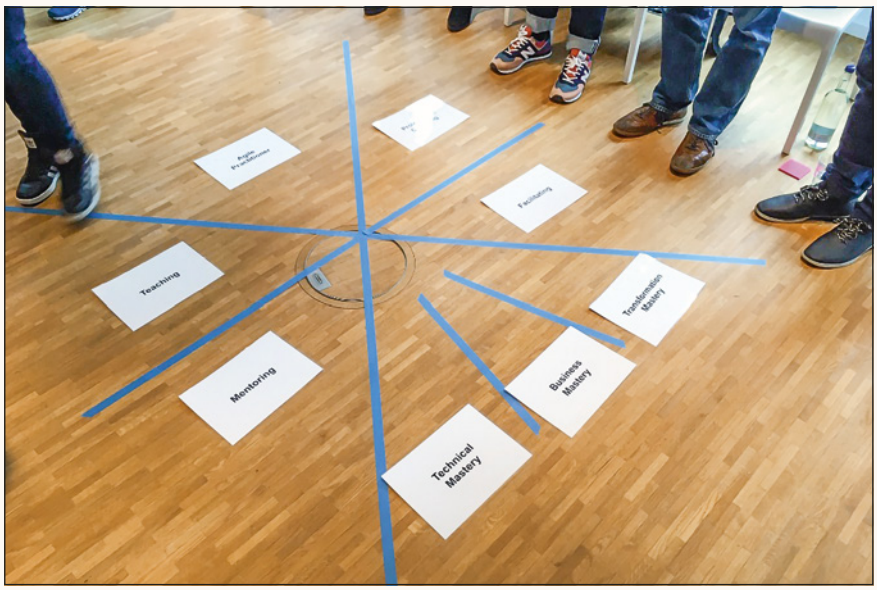


Abb. 5: Self-Assessment im Team mithilfe des Agile Coaching Competency Framework als Ausgangspunkt für die Weiterentwicklung

[1]) Natürlich kann anstatt der Einteilung 1 bis 10 alternativ bspw. auch das Shu – Ha – Ri Model (Lehrling, Geselle, Meister) angewendet werden. Im Sinne einer engeren Verzahnung mit einem strukturierten Fortbildungsplan ist erfahrungsgemäß die Abstufung 1 bis 10 besser geeignet.

Referenzen

- [ACI] Agile Coaching Competency Framework, siehe: https://agilecoachinginstitute.com/agile-coaching-resources/
- [Coh09] M. Cohn, Succeeding with Agile, Software Development Using Scrum, Pearson Education, 2009
- [KaLi18] Th. Karl, N. Liedl, Qualitätssicherung im Kontext von großen agilen Softwareentwicklungsprojekten, in: German Testing Magazin, 2/2018, siehe: https://www.accenture.com/t00010101T000000Z__w__/de-de/_acnmedia/PDF-94/Accenture-Qualitatssicherung.pdf#zoom=50
- [KaLi20] Th. Karl, N. Liedl, Testautomatisierung 4.0 mithilfe des House of agile Testing, in: JavaSPEKTRUM, 3/2020

Betrachtung der Qualitätsaktivitäten über das gesamte Projekt erweitern. Gerade in Bezug auf die Weitergabe und Verteilung von Kompetenzen vom Management auf die Teams über Teams hinweg oder gar der Eigenwahrnehmung von einzelnen Mitarbeitern ist das RMC eine sinnvolle Methode. In Verbindung mit dem „Delegation Poker" wird plastisch sichtbar, wie unterschiedlich meist die Sicht auf Zuständigkeiten ist oder gar, ob Verantwortungslücken entstanden sind. Es bietet spielerisch ein Analysewerkzeug, die Problemfelder sichtbar zu machen und im gleichen Zug zu behandeln. Eine regelmäßige Betrachtung des aktuellen Stands mit allen Beteiligten, oder in Kleingruppen, hilft, die Organisation in Bezug auf die neuen Rollen und Verantwortlichkeiten stetig zu fordern und weiterzuentwickeln.

Als Ergebnis einer derartigen Analyse kann sichtbar werden, inwiefern Rollen aus dem „Cross-Sprint Qualitäts-Fokus" **(siehe Abbildung 2)** nötig sind. Aufgaben können beispielsweise ineffizient innerhalb der Teams durchgeführt werden, weil sich mehrere Rollen oder Teams für Aufgaben verantwortlich fühlen **(siehe Abbildung 3)**. Rollen, wie Automation-Architekt, Regression/Release/Integration/UAT-Tester, Test-Architekt oder Test-Manager können dann ausgelagert werden, um im Projekt einheitliche Strukturen zu fördern und die Gesamtqualität final aus einem Guss sicherzustellen.

Ein weiterer Erfolgsfaktor für eine gute Qualitätssicht über das gesamte Projekt ist, auch ein Qualitäts-Ownership in den Rollen zu wecken, die initial nicht direkt mit Test und Qualitätssicherung zu tun hatten. Das geht über den Product Owner, der bei der Beschreibung der Anforderung wachsam ist, über den Entwickler, der das Shift-left-Vorgehen durch frühe Tests fördert, bis hin zu den Projektverantwortlichen, die übergreifend ein Mindset fördern, dass eine gute Qualität vor der reinen Ablieferung von Features steht.

Fazit

Zusammenfassend bleibt festzuhalten, dass alle Maßnahmen und Aktivitäten innerhalb der Säule 1 (Rollen) im höchsten Maße in Abhängigkeit zu den Entscheidungen innerhalb aller Bestandteile des HoaT (Ansatz, Tools und unterstützende Maßnahmen) sind. Darüber hinaus müssen alle Maßnahmen aus Säule 1 zwingend im engen Schulterschluss mit den Personalverantwortlichen, der HR-Abteilung und der Geschäftsleitung getroffen werden. Alle diese Maßnahmen zahlen unmittelbar auf die Motivation, Qualifizierung und Zufriedenheit der Mitarbeiter ein. Mitarbeiter stellen dabei in unserer heutigen VUKA-Welt insbesondere im Bereich der Wissensarbeit das wertvollste Gut dar!

Festzuhalten ist, dass Qualität zukünftig neu erfunden werden muss. Nur mit einem ganzheitlichen Denken und einer Umgestaltung der Arbeitsabläufe in der Softwareentwicklung ist das möglich. Aus Softwaretestern von gestern müssen innovationsorientierte, neugierige Qualitätsingenieure werden, die Freude daran haben, die Grenzen automatisierter Testabläufe mit Analytics und KI laufend weiter zu stecken.

Thomas Karl

thomas.karl@accenture.com
ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

Nico Liedl

nico.liedl@accenture.com
ist Quality Engineer und Advisor mit Schwerpunkt auf agilem Qualitätsmanagement, Testautomatisierung, Prozessverbesserung und organisatorischem Wandel. Er hat Erfahrungen bei verschiedenen komplexen Großprojekten gesammelt. Als Teammitglied, Scrum Master, Teststratege, Releasemanager und Trainer hat er praktische Erfahrung und theoretisches Fachwissen zu bieten und möchte sie mit einem breiten Publikum teilen.

»Distributed Agile Testing? Ja, es funktioniert!«

Heutzutage ist das Thema Digitalisierung allgegenwärtig. Es gibt zahlreiche Berichte in den Nachrichten, die sich mit diesem Thema auseinandersetzen. Der Fokus liegt nicht mehr nur auf IT-Unternehmen, sondern betrifft fast alle Branchen und Unternehmen. Die Unternehmen haben erkannt, dass die Digitalisierung nicht nur die IT-Abteilungen, sondern das gesamte Unternehmen betrifft. Eine große Rolle in der Digitalisierung spielen agile Entwicklungsmethoden, mit denen man nicht nur schnell und flexibel auf Veränderungen reagieren, sondern auch schnell Ergebnisse erzielen kann.

Ein Kernkonzept der agilen Methoden beruht auf der Co-Location, um die Abstimmungswege und die Zusammenarbeit zu erleichtern. Hier stellt sich die Frage, ob in Zeiten der Digitalisierung, der Revolution des Homeoffice und der zunehmenden Online-Kommunikation, bei der der Gesprächspartner meist nur einen Klick entfernt ist, eine Co-Location notwendig ist. Es gibt zahlreiche globale Roll-out-Projekte, bei denen Teams auf der gesamten Welt verteilt arbeiten und den Projekterfolg genauso gut meistern wie Teams in der gleichen Umgebung. Manchmal übertreffen diese verteilten Projekte sogar die Projekte, bei denen alle Mitarbeiter an einem Ort sind, dies lässt sich vermutlich auf die große Diversity der Teilnehmer zurückführen.

Es kommt häufig vor, dass der Qualitätsaspekt in agilen Entwicklungsmethoden vernachlässigt wird. Jedoch ist es gerade bei agilen Entwicklungsmethoden von großer Bedeutung, die Qualität von Beginn an zu berücksichtigen. Hier spricht man vom Shift-left-Ansatz. In einfachen Worten steht hinter Shift-left die Idee, die Qualität der Software durch ein Verschieben der Aufgaben im Lebenszyklus so weit wie möglich nach vorne (links) zu verlagern. So sollen die Technical Debts und die Cycle-Time reduziert werden. Der Fokus rückt vom Auffinden von Fehlern hin zur Prävention von Fehlern. Was passiert, wenn man sich entscheidet, die Qualitätssicherung auszulagern? Hierzu betrachten wir ein Beispiel, welches in ähnlicher Weise erfolgreich umgesetzt wurde.

Beispiel-Testansatz für Distributed Agile Testing

Wie **Abbildung 1** zeigt, wurde gemeinsam mit dem Kunden ein Testansatz gewählt, der genau auf die Anforderungen des Kunden im agilen Set-up abgestimmt ist.

Die wichtigen Unit-, Service-Layer- und Performanz-Tests sind weiterhin fester Bestandteil der Entwicklung und werden im In-Sprint Scope ausgeführt. Für dieses Modell wurden die Aufgaben des funktionalen Tests, die Testautomatisierung, der Regressionstest und das Defektmanagement an das Offshore-Team übergeben, welches diese Aktivitäten sowohl In-Sprint als auch Cross-Sprint ausführt. So erstellt das Offshore-Team im

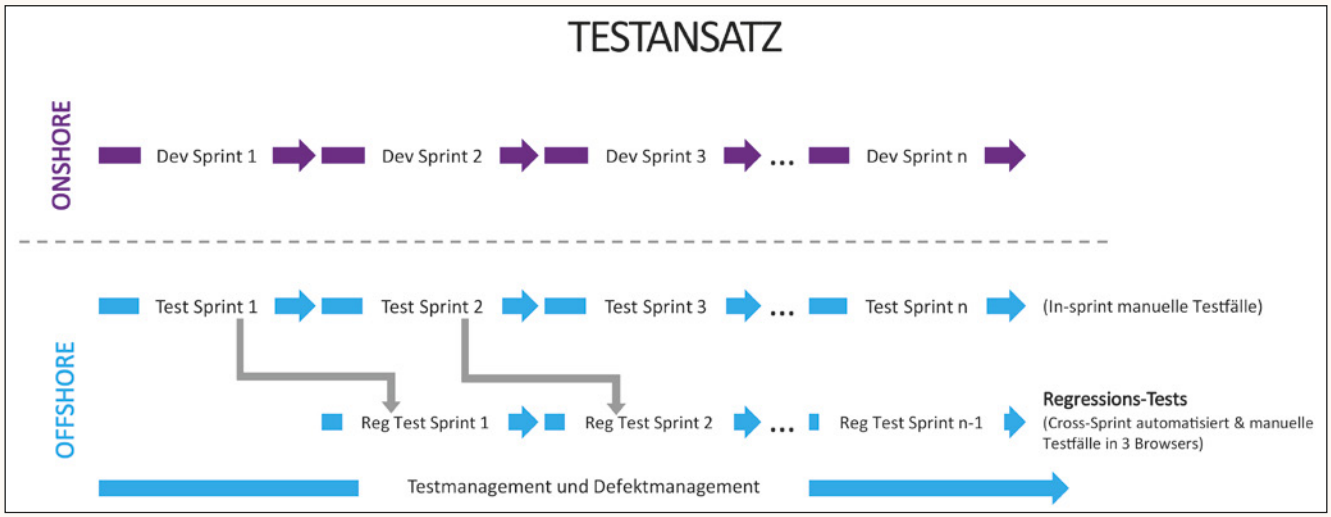


Abb. 1: Beispiel-Testansatz

Sprint die funktionalen Testfälle und führt diese manuell aus. Bei erfolgreichem Durchlauf werden diese in die Automatisierung überführt und falls notwendig direkt in den Cross-Sprint-Regressionstest aufgenommen. Werden Fehler im Prozess festgestellt, so werden diese in Jira dokumentiert und dem entsprechenden Entwicklungsteam zugewiesen.

Um die Organisation über die Teams zu vereinfachen, wurde ein Workflow zwischen dem Kunden, dem Onshore- und Offshore-Team abgestimmt und festgelegt **(siehe Abbildung 2)**. Hierdurch konnte Transparenz über den Ablauf und das Vorgehen geschaffen werden, was die Akzeptanz des Entwicklungsteams gesteigert und allen Beteiligten die erfolgreiche Zusammenarbeit ermöglicht hat.

Anwendungsformen für Distributed Agile Testing

Im agilen Kontext findet man viele Variationen und Umsetzungen der bekannten Vorgehensmodelle und Frameworks wie Scrum oder SAFE. Oft adaptieren Projekte nur zum Teil die Mechanismen, Strukturen und Abläufe dieser Modelle, um, zusammen mit den ganz individuellen Projektbedingungen und internen Vorgaben, ein passendes Arbeitsmodell umzusetzen. Die Vielzahl an Modellen stellt die Qualitätssicherung immer wieder vor sehr spannende Herausforderungen.

Die angesprochene Vielfalt spiegelt sich auch in den möglichen Anwendungsszenarien im Bereich Distributed Agile Testing wider. In der Praxis sind die folgenden beiden Modelle besonders häufig anzutreffen:

- Distributed Testing und
- Distributed Delivery.

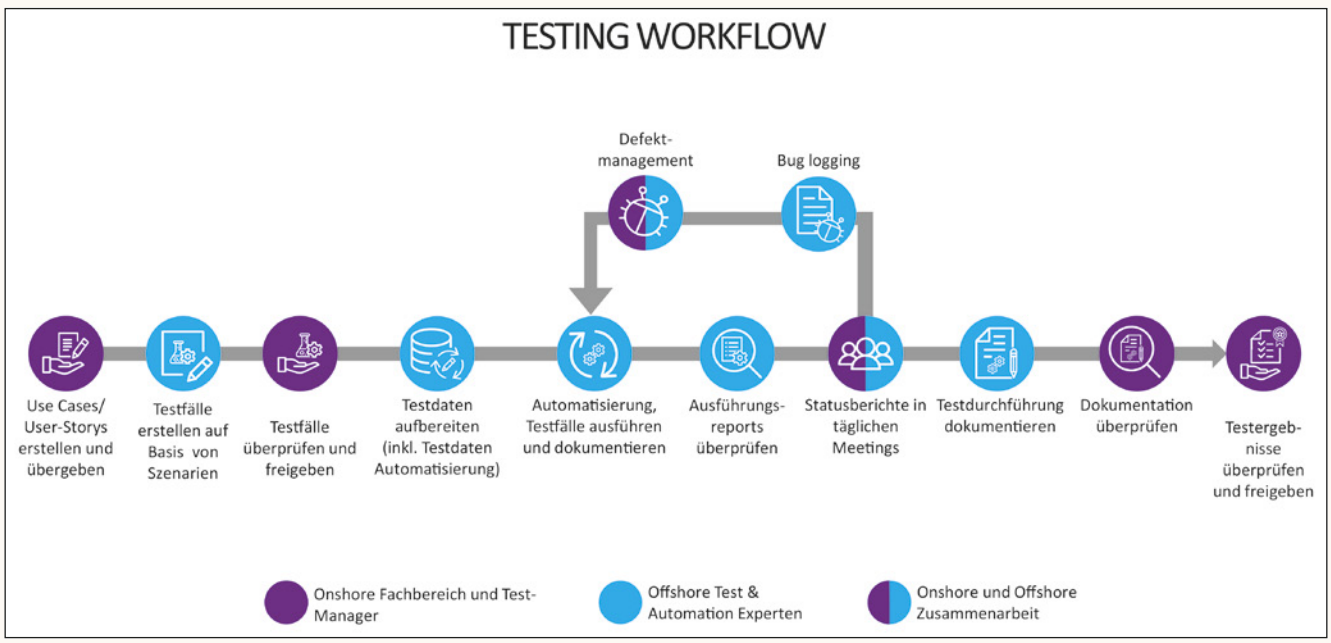


Abb. 2: Beispiel-Workflow

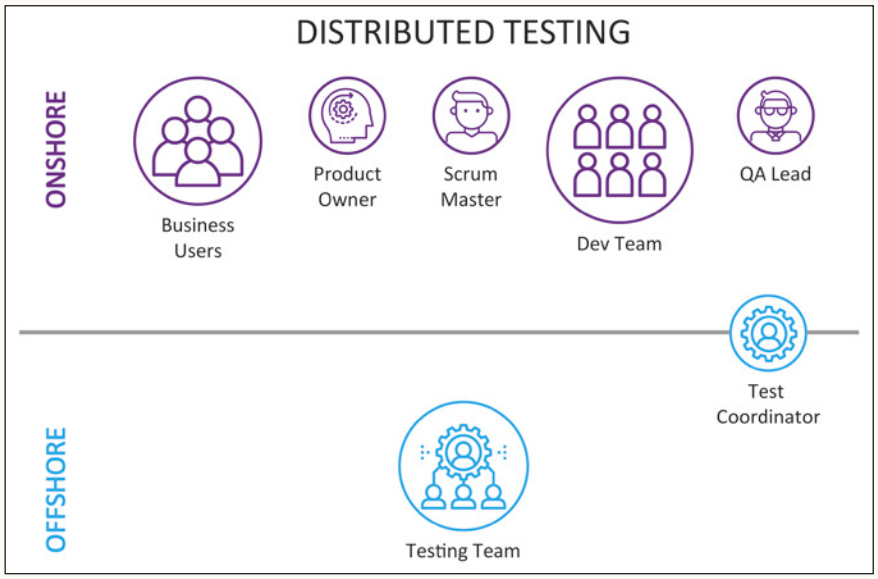


Abb. 3: Distributed Testing

Distributed Agile Testing

Der Ansatz des Distributed Agile Testing beinhaltet die Verlagerung der Qualitätssicherung im verteilten Modus und wird in **Abbildung 3** schematisch dargestellt.

Scrum Master, Product Owner sowie das Development-Team und die fachliche Expertise verbleibt in einer Co-Location. Es wird jedoch ein dediziertes Test-Team gebildet, welches die Testaktivitäten Offshore oder auch Nearshore übernimmt. Dieses Team kann in einem skalierten Ansatz mit mehreren Teams ähnlich eines Test-Factory-Ansatzes die Testaktivitäten als Service übernehmen. Es ist aber ebenso möglich, dass die Tester jeweils einem agilen Team zugeordnet sind und für dieses exklusiv die Qualitätssicherung übernehmen. In diesem zweiten Ansatz kann die Weiterentwicklung des im Test-Team vorhandenen Know-hows durch einen regelmäßigen Team-Wechsel der einzelnen Teammitglieder zusätzlich gefördert werden.

Zur Koordination der Testaktivitäten, insbesondere bei skalierten Projekten mit mehreren agilen Teams, sollte, wie bereits angesprochen, ein Test-Koordinator eingesetzt werden. Dieser übernimmt zum einen die Koordination, zum Beispiel teamübergreifender Testaktivitäten wie Integrations- und End-2-End-Tests, zum anderen übernimmt er eine Brückenfunktion bezüglich der räumlichen Trennung der Teammitglieder im verteilten Projektansatz, um den notwendigen Kommunikationsfluss sicherzustellen.

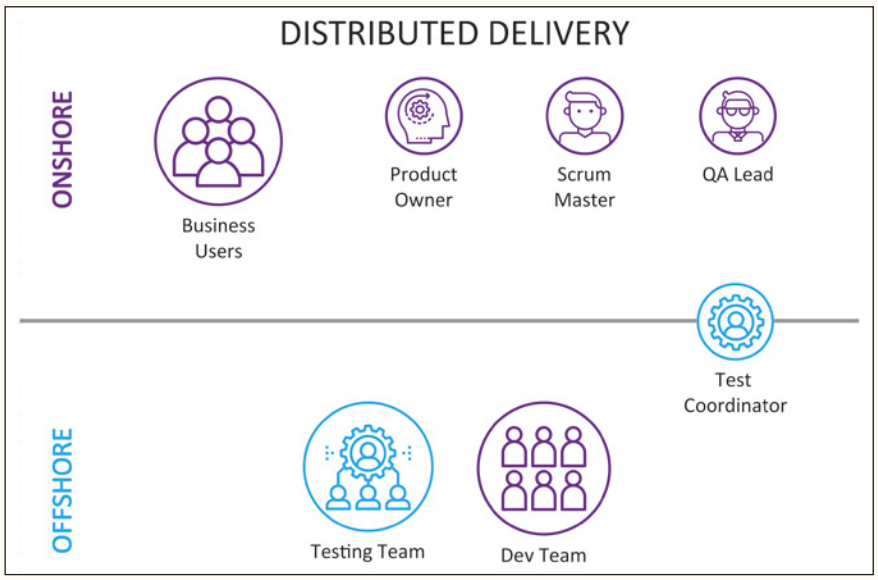


Abb. 4: Distributed Delivery

Um die Qualitätssicherung zu fördern, kann es sinnvoll sein, im Onshore-Umfeld des Projekts einen QA-Lead zu benennen oder dessen Rolle onshore zu vergeben. Der QA-Lead ist unter anderem für die gewählte Teststrategie und die Strategie zur Testautomatisierung zuständig. Weiterhin übernimmt er vor Ort das teamübergreifende Status-Reporting an die Projektleitung. Er ist aber, ähnlich zu der Rolle des Scrum Masters, ebenso verantwortlich dafür, Projekthemmnisse und Verbesserungspotenzial im Bereich Quality-Engineering zu identifizieren und die testrelevanten Prozesse zu optimieren. Hierbei sollte er insbesondere den Fokus auf die im Team notwendigen Skills, den Einsatz von geeigneten Tools, aber auch den Informationsfluss und die Motivation des Teams legen. Trotz des verteilten Ansatzes ist es von großer Wichtigkeit, dass gemäß dem agilen Vorgehen die Verantwortung für Qualität nicht allein beim dedizierten Test-Team, sondern in der Verantwortung aller Beteiligten liegt.

Distributed Agile Delivery

Im Gegensatz zum Ansatz des Distributed Agile Testing wird bei Distributed Delivery **(siehe Abbildung 4)** nicht nur die Aktivität des Testens, sondern auch die Entwicklungstätigkeit in ein Delivery Center ausgelagert.

Fachbereich, Product Owner und Scrum Master bleiben weiterhin in einer Co-Location. Auch der im Distributed Agile Testing bereits vorgestellte QA-Lead ist in der Co-Location verortet. Je nach Aufbau und Größe des Projekts, zum Beispiel bei einem skalierten Ansatz mit mehreren Agilen Teams, sind hier neben dem beschriebenen Test-Koordinator auch Rollen für die Koordination der Entwicklungsaktivitäten denkbar und sinnvoll.

Bei einer Auslagerung von Entwicklung und Testen in Offshore-Abteilungen lassen sich Synergieeffekte nutzen, insbesondere bei einem hohen Grad an Testautomatisierung sowie der Verwendung von TDD (Test-Driven Development) oder ATDD (Acceptance Test-Driven Development).

Möglichkeiten und Herausforderungen

Beide Ansätze des verteilten Testens bieten die Vorteile der Ausnutzung von Offshore/Nearshore-Ansätzen (Kostenersparnis, Nutzung von lokal nur begrenzt vorhandenem Expertenwissen, Skalierung der Kapazitäten passend zum jeweiligen Projektbedarf). Ebenso ist eine mögliche Zeitverschiebung zwischen on- und offshore ein nicht zu unterschätzender Faktor, da der Arbeitstag virtuell um die entsprechenden Stunden verlängert wird. Natürlich ist eine Verteilung der Aktivitäten aber auch mit den im Vorfeld beschriebenen Herausforderungen verbunden.

Die starke räumliche Trennung zwischen fachlichen Stakeholdern sowie Entwicklern und Testern scheint dem Gedanken der täglichen persönlichen Zusammenkunft zum Beispiel zum Daily Stand-up sowie der so gewinnbringenden engen Zusammenarbeit entgegenzustehen. Ebenso wird durch die organisatorische Trennung in Entwickler- und Test-Teams der positive Aspekt von interdisziplinären und selbstorganisierenden agilen Teams erschwert, da vermehrt auf koordinative Rollen zurückgegriffen werden muss. Ein solcher Ansatz kommt daher in der Regel eher bei großen, komplexen und entsprechend skalierten Projekten zum Einsatz.

In einem Projekteinsatz wurde ein entsprechendes Modell zum Beispiel in einer Konstellation mit 27 agilen Teams über einen Zeitraum von mehreren Jahren umgesetzt. Der Herausforderung bezüglich Informationsfluss, Zusammenarbeit und gemeinsamer Verantwortung wurden zum einen durch den Einsatz von täglichen Videokonferenzen und einer ständig offenen Videoschaltung als virtuelles Fenster zum offshore arbeitenden Büro begegnet. Zum anderen wurde aber auch kein reiner On-/Offshore-Ansatz gewählt, sondern ein Teil der Entwickler- und Test-Teams war stets vor Ort. Die Kollegen wurden also, obwohl eigentlich offshore verortet, immer für einen begrenzten Zeitraum auch in der Co-Location eingesetzt und regelmäßig durch Kollegen aus dem Offshore-Bereich Entwicklung und Test ausgetauscht. Durch diesen regelmäßigen Austausch wurde der Teamgedanke durch persönlichen Kontakt erheblich gefördert und die fachlichen Kenntnisse konnten vor Ort sehr viel schneller und tiefer ausgebaut werden.

Um weitere Synergieeffekte zu erzeugen, wurden sowohl Entwickler als auch Tester nicht nur zwischen on- und offshore, sondern auch durch die verschiedenen agilen Teams rotiert. Dies führte zusätzlich zu einem breiten Verständnis der teamübergreifenden Zusammenhänge und integrativen Funktionen des gesamten Projekts.

Fazit

Diejenigen, die bereits im agilen Umfeld Erfahrungen sammeln durften, werden zustimmen, dass die Sicherung der Qualität in agilen Projekten besondere Herausforderungen bereithält. Der verteilte Ansatz kann viele Vorteile bieten, führt aber auch zu zusätzlichen Problemen, die beachtet werden müssen. Beispiele hierfür sind bereits im einfachen menschlichen Miteinander durch die kulturellen Unterschiede zu finden, mitunter kommt es gar zu „Wir gegen die"-Gedanken, da im agilen Zusammenarbeiten Fehler und Schwächen sehr viel offensiver angesprochen und adressiert werden, als dies normalerweise im traditionellen Ansatz mit dedizierten Teams und geschlossenen Verantwortlichkeiten der Fall ist. Aber auch der direkte Kontakt zueinander, das Treffen in der Kaffeeküche, der Smalltalk beim Gang in der Kantine sind im verteilten Ansatz nicht mehr alltäglich, was zudem den Zusammenhalt des Teams und dadurch oft auch die Produktivität beeinflussen kann.

Wie im Beispiel dargestellt, sollte ein Prozessflow festgelegt werden, welcher die Kanäle und Strukturen für die notwendigen Kommunikationen regelt. Virtuelle Kanban Boards, multimediale Meetingräume mit 360°-Kamera, Microsoft Teams, permanente Video-Sessions und digitale Whiteboards helfen bei der direkten Kommunikation.

Ebenso kann der Zusammenhalt des Teams durch regelmäßige persönliche Treffen am Projekt- und Offshore-Standort, durch die übergreifenden Austauschprogramme und durch teambildende Maßnahmen und Events (z. B. Virtuelle Lunch-Termine) erfolgreich gefördert werden.

Martin Flockenhagen

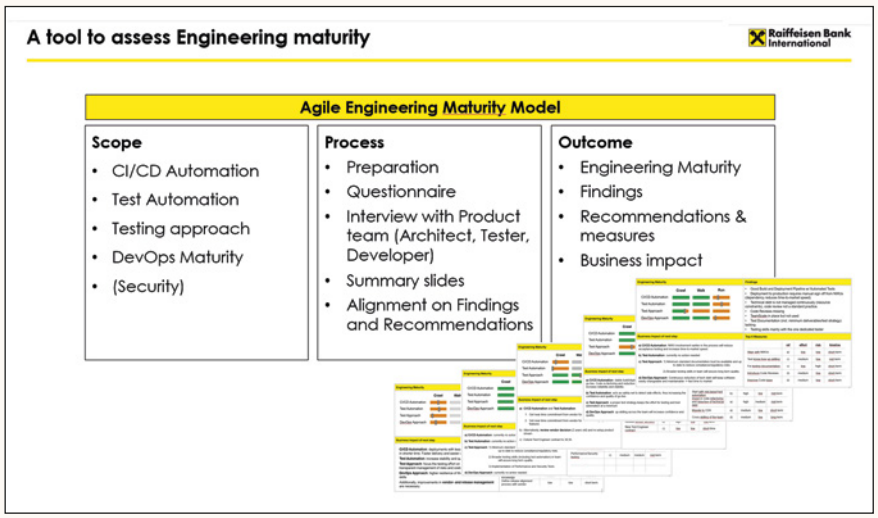
martin.flockenhagen@accenture.com
ist Test-Manager und Advisor mit Schwerpunkt auf Qualitätsmanagement und Prozessverbesserung in komplexen skalierten agilen Projekten. In seiner Tätigkeit befasst er sich mit agilen Methoden, Lean Six Sigma, SAFe, Scrum und Kanban, sowie deren Einsatz zur Qualitätssicherung.

Dennis Dangmann

dennis.dangmann@accenture.com
ist Quality-Engineer und Advisor mit Schwerpunkt auf agilem Qualitätsmanagement, Testautomatisierung, Prozessverbesserung und distributed agile Testing. Er hat Erfahrungen bei verschiedenen komplexen verteilten Großprojekten gesammelt. Als Teammitglied, Scrum-Master, Teststratege, Testkoordinator und Test-Lead hat er praktische Erfahrung gesammelt und vertraut in die Diversity eines verteilten Teams.

»Ein Test-Evaluierungsmodell als Lösung zur Weiterentwicklung für agile Teams?«

Die Raiffeisen Bank International AG (RBI) begann 2014 mit der Umstellung der Projektabwicklungen von einem klassischen Ansatz, mehrheitlich nach dem Wasserfallmodell, in einen agilen Ansatz. Dies ergab nicht nur Änderungen für Projektteams, sondern auch konkret für das Berufsbild der Tester im Unternehmen. Aus unabhängigen Testteams wurden agile Teams, welche aus Entwicklern, Testern und Fachbereichs-Mitarbeitern bestehen. Diese Ausrichtung trägt dem Whole-Team-Approach Rechnung. Unterstützung fanden die Teams dabei durch Agile Engineering Coaches, welche durch ein selbst entwickeltes Test-Reifegradmodell (Agile Engineering Maturity Modell, kurz AEMM) den Teams eine Evaluierung und Verbesserung ihrer Engineering-Fähigkeiten im Bereich Test ermöglichen.



Diese Neuausrichtung wurde 2019 komplett umgesetzt und brachte die Herausforderung, Testen gemeinsam wahrzunehmen und sich als (neues) Team weiterentwickeln zu wollen. Die Evaluierung wurde mittlerweile auf das gesamte RBI-Headoffice ausgeweitet. Bisher nahmen 50 Prozent aller Teams im Headoffice an der Befragung teil. Zurzeit bereiten die Agile Engineering Coaches die Ausweitung auf den RBI-Gesamtkonzern in CEE (Central and Eastern Europe) vor.

Dieser Artikel gibt einen Einblick, wie das Reifegradmodell entwickelt und angewendet wird, und zeigt den Nutzen für die Teams auf.

Hintergrund für den Bedarf des AEMM

Schon vor dem Start der Transformation der Projektabwicklung zum agilen Ansatz war absehbar, dass die Engineering-Skills in den jeweiligen Teams stark abweichen. Das sollte näher beleuchtet werden. Ferner wollte das Management unter dem Titel „Agile Investment" die agilen Teams auch mit Geld- und Zeit-Budget unterstützen, um neue Technologien, Tools und Methoden einzuführen („Upskilling").

Ziel waren vor allem sehr alte („legacy") und historisch gewachsene Applikationen und die zugehörigen Teams. Die übergeordneten Ziele sind ein Mindeststandard für die vier Bereiche CI/CD, DevOps, Test und Testautomation und weiters die Möglichkeit, die Umsetzung von Verbesserungen über die Zeit bewerten zu können.

Entwicklung des Modells AEMM, Version 1

Basis für die Bereiche „Testing" und „Test Automation" des AEMM waren die Modelle TPI NEXT® (Sogeti, [TPI]) und TMMi (Test Maturity Model integration, [TMMi]). Nach einigen Testläufen mit Pilotteams wurde eine adaptierte Variante des TPI Next namens Agile TPI® (Sogeti) ausgewählt.

Für die Bereiche DevOps und CI/CD wurde das Reifegradmodell der Standard-Bank als Basis verwendet. Für alle vier Teilbereiche wurde schlussendlich ein einfacher gemeinsamer Bewertungsansatz in drei Reifegradstufen (Crawl-Walk-Run) analog des DevOps Maturity Model der Firma Adidas [GitH] gewählt **(siehe Tabelle 1)**.

Im Zuge einer mehrmonatigen Pilotphase wurden Erfahrungen gesammelt und schlussendlich die *Fragen* für die Version 1 so adaptiert, dass sie möglichst *allgemeine Gültigkeit* haben. Eigen- und Fremdentwicklungen werden gleichermaßen unterstützt. Auch unterschiedliche Technologien müssen vergleichbare Ergebnisse erzielen.

Erhebung des Test-Status-quo

In AEMM, Version 1 werden in einem zweistündigen Interview ca. 150 Fragen in 27 detaillierten Kategorien beleuchtet. Enthalten sind auch Fragen zu CI/CD und DevOps, der Artikel geht aber nur auf Test-relevante Fragen ein. Diese decken Themen wie Test-Design, Testautomation, Soft Skills usw. ab. In der Auswertung wird jeder Fragenkategorie ein Reifegrad auf Basis der Einzelfragen pro Kategorie zugewiesen: „Crawl" (Must be), „Walk" (Should be) oder „Run" (Could be).

Ziel ist es, den Teams einen Überblick über den Status quo zu geben sowie *Verbesserungspotenziale* aufzuzeigen. Die Auswertung des Fragebogens erfolgt in einem einseitigen *Executive Summary* durch Agile Engineering Coaches. Die Ergebnisse der einzelnen Kategorien werden aggregiert dargestellt und es werden Empfehlungen zur Verbesserung ausgesprochen **(siehe Abbildung 1)**. Betrachtet werden auch potenzielle Auswirkungen bestimmter Maßnahmen auf das Business.

Category	Maturity	
Test Analysis & Design	C	Test specifications are based on test strategy
Test Analysis & Design	C	Test specifications are sufficiently described and stored in HPQC or Adaptavist
Test Analysis & Design	C	Test specifications are created by testers based on (implicit & explicit) acceptance criteria
Test Analysis & Design	W	Test specifications are designed as part of the iteration (sprint) and design is finished once development finishes
Test Analysis & Design	W	Test specifications for security and performance requirements are available
Test Analysis & Design	R	Test specifications are created during story elaboration by the team
Test Analysis & Design	R	Example based test design methods like BDD or SBE are used to design tests
Test Analysis & Design	R	Test specifications for NON-functional requirements are in place
Test Analysis & Design	R	Exploratory Testing charters are used
Test Analysis & Design	R	The different types of test specifications are constantly evaluated, improved and brought back into the testing guild in order to improve all teams regarding testing

Tabelle 1: Beispielkategorie on AEMM, Version 1 „Test Analysis & Design" mit Fragen und zugehörigen Reifegradstufen (*C*rawl, *W*alk, *R*un)

Alle Vorschläge werden in einem zweiten Gespräch mit den verantwortlichen Teammitgliedern abgestimmt. Bei Bedarf werden dann Maßnahmen ergänzt oder neu bewertet. Die Umsetzung erfolgt grundsätzlich in Eigenverantwortung durch die jeweiligen Teams. Falls gewünscht, arbeitet ein Agile Engineering Coach einige Zeit im Team mit.

Herausforderungen bei der Evaluierung

Erhebungen jeglicher Art sind für die befragten Teams oft mit Vorbehalten behaftet, weil ein Vergleich mit anderen Teams befürchtet wird. Eine übertrieben positive Darstellung der Ist-Situation ist oft die Folge. Deswegen wird mehrfach die stark *eingeschränkte Vergleichbarkeit* der Ergebnisse zwischen den Teams betont. Zusätzlich werden die Fragen bereits im Vorfeld versendet und es wird die Auswertungslogik erklärt. Die Ergebnisse (Executive Summary) der Teambefragungen werden für Auswertungen anonymisiert und aggregiert.

Erkenntnisse für Teams, Coaches und Management

Eine der wohl spannendsten Erkenntnisse mit Version 1 war, dass viele Teams von sich aus an einem *Agile Engineering Maturity Model* Interesse zeigen. Das Interesse war bereits in der Pilotphase absehbar – unter anderem an den Diskussionen im Zuge der Befragung.

Teams erkennen den Nutzen, wenn sie sich in ihrem „eigenen Test-Status-quo" wiederfinden und die mit ihnen erarbeiteten Verbesserungsvorschläge in ihr Team-Backlog übernehmen können. Für viele Teams entstehen durch die umfassende Abdeckung der Test-Thematik im AEMM wichtige *neue Ideen* zur Verbesserung. Ein Mehrwert für ein Team kann beispielsweise ein Aufgreifen von agilen Testmethoden sein oder die Optimierung der Qualität der Testdaten. Für Coaches gibt es durch die Interviews laufend Rückmeldung zum Modell selbst, sodass dieses weiter verbessert werden kann.

Für das Management werden auf aggregierter Ebene **(siehe Abbildung 2)** in den jeweiligen Kategorien nach einer größeren Anzahl an Interviews *ähnliche Muster* oder Trends erkennbar, beispielsweise bei der durchgängigen Umsetzung des Whole-Team-Approachs in der Testautomation. Auf Basis dieser Muster kann das Management Team-übergreifende Maßnahmen beschließen.

Teams erhalten mit der Executive Summary eine *Basis* für Verhandlungen mit dem Management oder Product Owner, um für die Verbesserung ihrer Testaktivitäten die benötigte Priorität im Team-/Product-Backlog zu bekommen. Ein Beispiel wäre die Integration von Testautomationsaktivitäten in eine CI/CD-Pipeline.

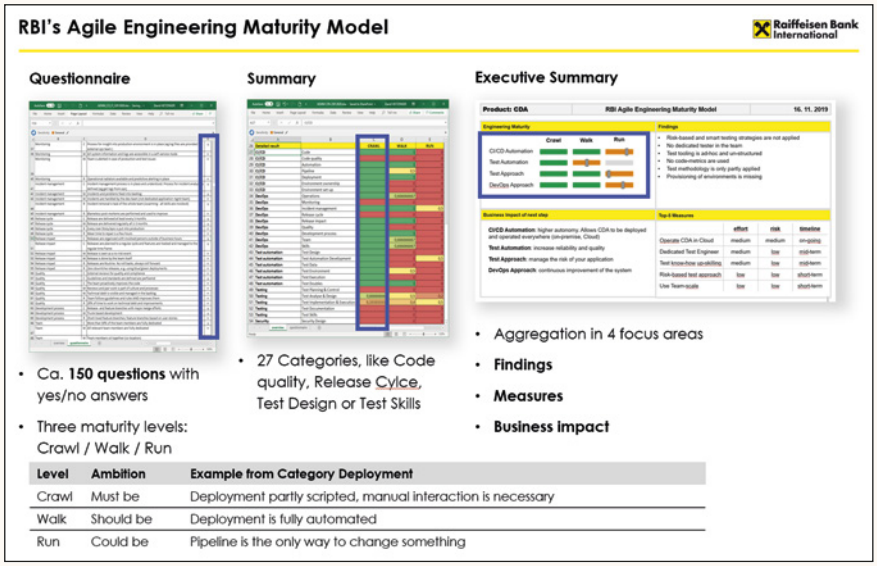


Abb. 1: Gesamtüberblick über die drei Stufen eines Interviews mit einem Product Team – vom Fragebogen über die Summary zum Fragebogen bis zur Executive Summary

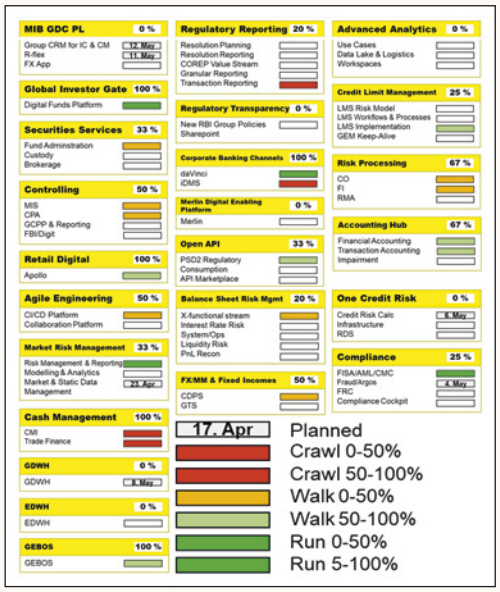


Abb. 2: Aggregation der AEMM-Ergebnisse über eine Vielzahl an Product Teams

Ergebnisse der Evaluierung

Nach der Durchführung von ca. 30 Interviews mit AEMM, Version 1 konnten die folgenden Kernthemen ermittelt werden:

Im Bereich Testansatz

Abbildung 3 fasst die Ergebnisse zusammen. Wichtige *Test-Methoden* wurden nur teilweise oder nicht angewendet, zum Beispiel die Durchführung nicht-funktionaler Tests, explorativer Tests oder risikobasierte Testdurchführung. *Test-Dokumentation* war veraltet und entsprach nicht den firmeninternen Mindeststandards, zum Beispiel waren Test-Strategien nicht mit Produkt-Risiken abgestimmt oder Testfälle wurden nicht im firmeninternen Testmanagement-Tool verwaltet. Auch *Test-Skills* waren nach der agilen Transformation noch nicht in allen Abteilungen ausreichend aufgebaut, der Whole-Team-Approach konnte nicht überall umgesetzt werden. Auf sich alleingestellte Test-Experten, Teams ohne Test-Automations-Know-how sowie unzureichende *Testdaten-Qualität* sind konkrete Folgen davon.

Im Bereich Testautomation

Die *Testfalldurchführung* war in einigen Teams *nur teilweise automatisiert* – nur Tests für einzelne Komponenten (z. B. Backend, Frontend) automatisiert, Regressionstests nur teilweise automatisiert **(siehe Abbildung 4)**. Der *Whole-Team-Approach* war manchmal nur zum Teil umgesetzt – ein hoch spezialisiertes Testframework wird von einem einzigen Teammitglied „bedient", das Testfalldesign und die Testfallentwicklung werden ausschließlich von Testern durchgeführt, jedoch nicht auch vom Fachbereich oder von Entwicklern. In drei Teams gab es *keine Testautomation*. In anderen Teams wurden *alle Tests automatisiert*, jedoch noch nicht in die CI-Pipeline eingebunden. Hinsichtlich *Testautomations-Methodik* waren in einem Team die Aufwände der Testautomation nicht dokumentiert, eine Kosten/Nutzen-Analyse für zu automatisierende Testfälle fehlte.

Anpassung des Evaluierungsvorgehens mit AEMM, Version 2

Nach den ersten 30 Interviews mit Product Teams wurden in einer Retrospektive der Agile Engineering Coaches mehrere Problembereiche betreffend Inhalt und Ablauf genannt.

Das wichtigste Problem war, dass der geplante Zeitrahmen von zwei Stunden in 90 Prozent der Interviews überzogen wurde. Die längsten Interviews dauerten in Summe bis zu 5 Stunden. Grund war meist verlorene Zeit durch Erklärungen des Modells an sich und Detaildiskussionen zur Auslegung einzelner Fragen. Da die Zeit für die Interviews nicht verlängert werden sollte, war eine Strukturänderung notwendig. Nun stehen immer drei Fragen – je eine für jede Reifegradstufe – nebeneinander in einer Zeile **(siehe Tabelle 2)**.

Das zweite Hauptproblem waren widersprüchliche Endergebnisse. Durch die Möglichkeit, Fragen unabhängig voneinander mit „JA" oder „NEIN" zu beantworten, entstanden mehrfach Situationen, wo Fragen für höhere Reifegradstufen positiv beantwortet wurden, verwandte Fragen für niedrigere Reifegradstufen hingegen negativ. Deswegen wurden die Einzelfragen von Fragengruppen besser abgestimmt, zusätzlich wurden viele Fragen überarbeitet. Die Verbesserungen sind nun mit der Version 2.0 des AEMMs umgesetzt.

In ersten Probeläufen wurde eine Reduktion der benötigten Zeit auf 40 bis 60 Prozent gemessen und die Ergebnisse scheinen auf den ersten Blick gleich gut zu bleiben. Es besteht aber die Befürchtung, dass die neue Struktur mittelfristig zu einer Tendenz der Einordnung in der Mitte – dem Reifegrad Walk – führt und damit die Ergebnisse verfälscht.

Abgesehen davon gibt es weitere kleinerer Probleme. Interviews waren manchmal wegen falscher Teilnehmer wenig aussagekräftig. Andere Teams betreiben offensichtliches Overselling, was insofern ein Problem darstellt, weil das Team entweder eine falsche Selbsteinschätzung hat oder aber das AEMM als unerwünschte Einmischung betrachtet. In beiden Fällen ist eine selbstständige Verbesserung des Teams fraglich.

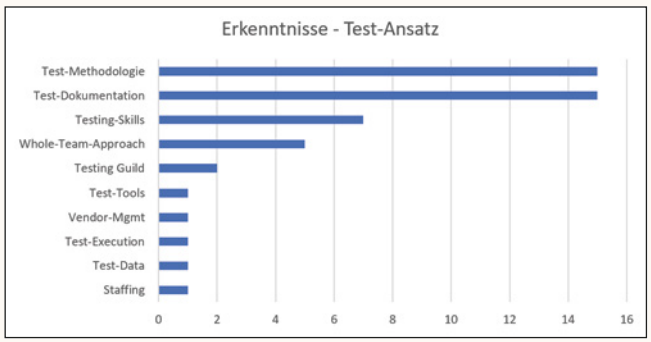


Abb. 3: Häufigste gefundene Kernthemen/Problembereiche im Bereich Testing/Testansatz nach 30 Interviews

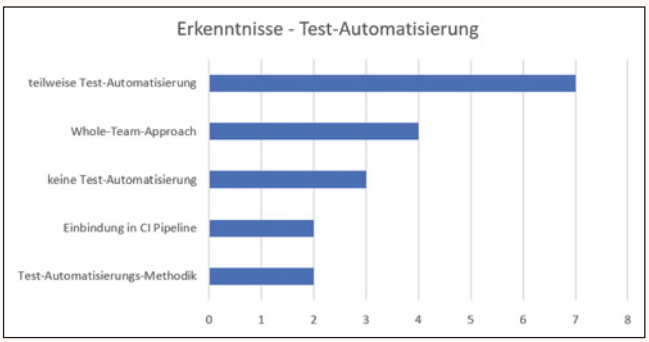


Abb. 4: Häufigste gefundene Kernthemen/Problembereiche im Bereich Testautomation nach 30 Interviews

Category	CRAWL	WALK	RUN	Self assessment	Comment
Test Analysis & Design	Test specifications are derived from test strategy	Test specifications for non-functional requirements (security and performance) are available	Test specifications for extended non-functional requirements (interoperability, ...) are available		
Test Analysis & Design	Test specifications are created by testers based on (implicit & explicit) acceptance criteria	Exploratory Testing charters are used to get insight in the system and use this insight for designing new tests	The different types of test specifications are constantly evaluated, improved and brought back into the testing guild in order to improve all teams regarding testing		
Test Analysis & Design	Test specifications are sufficiently described and stored in HPQC and Adaptavist	Test specifications are designed as part of the iteration (sprint) and design is finished once development finishes	Test specifications are created during story elaboration before the sprint starts by the team		

Tabelle 2: Beispielkategorie in der neuen Struktur, Version 2.0

Um auch diese Probleme zu entschärfen, wird zukünftig die Auswahl der Teilnehmer aus den Product Teams genauer hinterfragt. Das Thema „overselling" soll durch kürzere Wiederholungsintervalle für neuerliche AEMM-Interviews adressiert werden.

Abschließend war noch die lange Durchlaufzeit für die Nachbearbeitung der Interviews ein Thema. Dadurch geriet für alle Beteiligten der Interview-Termin in Vergessenheit, was die Diskussion über Findings und Verbesserungsmaßnahmen erschwerte. Die Agile Engineering Coaches haben sich nun eine Frist von 1 bis 2 Wochen für die Nacharbeit gesetzt.

Fazit

Das AEMM ist ein Beispiel, wie mit einem Test-Reifegradmodell ein Mehrwert für Teams, Agile Engineering Coaches und auch das Management beziehungsweise das Unternehmen selbst geschaffen werden kann. In diesem Fall hat sich der Aufwand für das an das Unternehmen angepasste Modell gelohnt! Wenn die agilen Teams dadurch Optimierungen umsetzen, steigen insgesamt die Effizienz und Effektivität.

Referenzen

- [GitH] https://github.com/adidas/adidas-devops-maturity-framework
- [TMMi] https://www.tmmi.org/tmmi-model/
- [TPI] Sogeti, TPI, https://www.tmap.net/building-blocks/test-process-improvement-tpi

Christa Gegendorfer

christa.gegendorfer@rbinternational.com
arbeitet als Agile Engineering Coach bei der Raiffeisen Bank International AG. Sie beschäftigt sich seit mehr als 15 Jahren mit Blackbox-Testing im klassischen und agilen Umfeld und ist Topic-Lead für das Thema Testdaten-Management in der RBI Testing Guild.

Martin Rohr

martin.rohr@rbinternational.com
ist Agile Engineering Coach bei der Raiffeisen Bank International AG und unterstützt dort Feature-Teams in den Themen Test und Testautomation. Er beschäftigt sich seit ca. 20 Jahren mit dem Thema Test, Test-Management und Testautomation in den unterschiedlichsten Projekten im Banken- und Beratungsumfeld.

»Qualität in vier Stufen «

Um von der Wasserfallmethode zur agilen Welt mit Scrum zu gelangen, muss man Shift Left denken. Agil zu entwickeln bedeutet, den Test in frühen Entwicklungsphasen einzuplanen, um später umfangreichere und zeitaufwendigere Fehlerbehebungen zu vermeiden. Regulatorische Anforderungen im Testbereich können eine der vielen Herausforderungen bei der Transition von Wasserfall zu Scrum darstellen. Bei einem Projekt mit einer Großbank ist jene Herausforderung erfolgreich mit einer Teststrategie im Vier-Stufen-Modell bewältigt worden.

Der erste Teil des Artikels befasst sich mit der Teststrategie. Hier wird erläutert, wie das Shift-left-Verfahren beim Testing gezielt in vier Teststufen eingesetzt werden kann. Im zweiten Teil wird kurz über die Erfahrung und die Ergebnisse dieser Teststrategie bei dem Bankkunden berichtet.

Um den Test in diesem agilen Modell kurz vorzustellen, vorab ein paar FAQs:

- *Was ist Shift-left Testing?* Dies ist ein Vorgehen, welches verschiedene Testansätze in früheren Stadien eines Softwarezyklus nutzt, um die Testabdeckung zu erhöhen und die Qualität der Tests zu verbessern. Dauerhaft führt dies zu einer Reduktion der Kosten, da früh identifizierte Testabweichungen weniger Aufwand für die Entwicklung sind als in einer späteren Phase des Softwarezyklus.
- *Welche Fähigkeiten braucht ein Tester?* Er muss im Stande sein, fachliche und auch technische Anforderungen in Testfallspezifikationen (hier: Workflows und Testszenarien) zu übersetzen. Im besten Fall ist der Tester auch Testautomatisierer, um den ständig wiederholten Bedarf an Regressions- sowie Last- und Performanztests zu decken.
- *Welche Voraussetzungen müssen für erfolgreiches Shift-left Testing in vier Teststufen geschaffen sein?* Ein gut strukturiertes Testumgebungsmanagement spielt hierbei eine zentrale Rolle. Vier Teststufen bedeuten nämlich auch, dass für jede Stufe eine eigene Testumgebung mit entsprechenden Testdaten bereitstehen muss. Ein geeignetes Testdatenmanagement, das die unterschiedlichen Testdaten, je nach Bedarf und Nutzungsgenehmigung, auf den jeweiligen Umgebungen bereitstellt, ist ebenfalls wichtiger Bestandteil der Teststrategie. Aufgrund der Datenschutzgrundverordnung (DSGVO), welche seit Mai 2018 anzuwenden ist, gewinnt dieser Bereich zudem immer mehr an Bedeutung. Verfahren zur Pseudonymisierung, Anonymisierung oder Anlage von synthetischen Testdaten sind die möglichen Lösungen, um sich von den produktionsgleichen Daten zu distanzieren.

Testen in vier Stufen

Vorbereitung durch Test-Driven Requirement Analysis

Das Vier-Stufen-Modell im Shift-left Testing **(siehe Abbildung 1)** beginnt schon vor der ersten Stufe.

In Vorbereitung auf die Entwicklung beginnt die Arbeit für das Testteam schon in der Anforderungsanalyse: Test-Driven Requirement Analysis. Bevor die fachlichen Anforderungen und Features in User-Storys runtergebrochen werden, werden vom Testteam Main-Workflows und alternative Workflows sowie dazugehörige Testdatenkonstellationen ausgearbeitet. Anhand dieser Vorbereitung werden dann die User-Storys und Akzeptanzkriterien geklärt und definiert.

Stufe 1 – Entwicklertests, doppelte Code-Reviews und statische Codeanalyse

Über JIRA erhalten Entwickler bei jedem Sprintbeginn die User-Storys. Auf JIRA wird der Entwicklungsprozess in den einzelnen Entwicklungsstufen dokumentiert. Bei jedem Entwicklungsabschluss einer User-Story erfolgt durch den Entwickler selbst ein Komponententest auf der Entwicklungsumgebung. Die Entwicklung wird dann zudem von einem weiteren Entwickler validiert, sodass hier ein doppelter Code-Review stattfindet. Die Entwicklertests werden während eines Sprints durchgeführt, bevor die User-Storys an das In-Sprint-Testing-Team für einen fachlichen Test übergeben werden.

Abb. 1: Vier-Stufen-Modell

Stufe 2 – In-Sprint-Tests (fachlich)

Die Übergabe der User-Storys für den Sprint erfolgt gleichzeitig an Entwickler und Tester. Während die Softwareentwicklung startet, beginnt für das Testteam das Wettrennen um die Erstellung der Testfälle für die einzelnen User-Storys. Die Testfallerstellung muss zum Zeitpunkt des Entwicklungsabschlusses ebenfalls abgeschlossen und vor In-Sprint-Testbeginn durch den Product Owner (PO) validiert werden. Nach Entwicklungsabschluss findet auf JIRA die Übergabe der User-Storys an den entsprechenden Tester statt (es gibt einen manuellen Tester pro Scrum-Team). Auch für Tests findet hier auf JIRA eine genaue Dokumentation der Testfallausführung statt, sodass nach Testabschluss eine Entscheidung zur Schließung oder erneuten Entwicklung der User-Story getroffen werden kann oder gar eine Anpassung der Anforderungen erfolgen muss.

Zum Shift-left-Ansatz zählt auch der regelmäßig, während eines Sprints, automatisiert ausgeführte Regressionstest **(siehe Abbildung 2)**. In der zweiten Teststufe wird mit

Abb. 2: Testautomatisierung im Vier-Stufen-Modell

Abb. 3: Dritte Stufe – Sprintreviews, Experten- und Usability-Tests

der Testautomatisierung begonnen. Während das manuelle Testteam sich mit den neuen User-Storys des aktuellen Sprints befasst, wenden sich die Testautomatisierer (ebenfalls einer pro Scrum-Team) den regressionsrelevanten User-Storys aus dem vorangegangenen Sprint zu und legen diese Testfälle automatisiert in der TOSCA-Testsuite an. Das Regressionstestportfolio wird auf diesem Weg Sprint für Sprint erweitert, um den letzten Sprintabschluss immer wieder erneut zu validieren.

Die Testabdeckung liegt im manuellen Test bei 100 Prozent, das heißt, es werden alle testrelevanten User-Storys und die dazugehörigen Akzeptanzkriterien vertestet. Für die Testautomatisierung liegt das Testabdeckungsziel bei 80 Prozent, hier werden nur

Abb. 4: UAT-Aufwand wird verringert

die regressionsrelevanten Akzeptanzkriterien in Betracht gezogen.

Stufe 3 – Feedback durch das Fachteam sowie Experten- und Usability-Tests

Um während der Entwicklungssprints den Product Owner über den Entwicklungsfortschritt abzuholen, wird der Sprintreview genutzt. Die Vorstellung der erfolgreich abgeschlossenen User-Storys erfolgt mit dem PO, sodass ein frühes Feedback eingeholt werden kann. Um jedoch Bank-interne Experten und auch zukünftige Anwender abzuholen und eine Akzeptanz der Applikation sicherzustellen, werden diese in Expertentests eingebunden **(siehe Abbildung 3)**.

Bei einem durch die Product Owner geführten Expertentest werden die bank-internen Experten und zukünftige Anwender vor verschiedene Aufgaben gestellt, die sie in der Anwendung ausführen sollen. Durch diesen geführten sowie zum Teil auch explorativen Test erhält der Experte ein Gefühl für die Anwendung und kann zudem Features für die nächsten Sprints entsprechend der Testergebnisse erweitern oder spezifizieren. Diese Stufe ist zudem eine Vorbereitung auf den User Acceptance Test, da so die Testfallerstellung für Stufe 4 durch das Fachteam mit etwas Vorwissen der Anwendung erfolgt.

Stufe 4 – User Acceptance Test

Bei der Transition von einem Wasserfallmodell zu einer agilen Softwareentwicklung bleibt häufig der User Acceptance Test (UAT) übrig, da dieser bei einer Abnahme von der Revision zur Betrachtung herangezogen wird. Nach Abschluss aller Entwicklungssprints findet daher noch mal ein finaler Test des Releases durch den PO sowie ein Last- und Performanztest statt. Der Last- und Performanztest kommt bei dieser Teststufe ins Spiel, da ab dem Zeitpunkt des UAT die Entwicklungen abgeschlossen sind und die Applikation bereits auf einer höheren und daher auch produktionsnahen Testumgebung deployt worden ist. Die Testergebnisse aus dieser Teststufe sind dann ausschlaggebend für die Entscheidung über die Freigabe der Applikation.

Erfahrungen aus der Praxis

Die Teststrategie über ein Vier-Stufen-Modell entstand in den Anfängen der Entwicklung des ersten Projektreleases bei dem Bankkunden und wurde über die Dauer von 2 Jahren sowie 4 weiteren Releases immer weiter ausgebaut.

Im ersten Release waren die Testautomatisierung der fachlichen Tests sowie ein Last- und Performanztest noch ein Zielbild für die kommenden Releases. Mit der Weiterentwicklung der Applikation durch mehr und komplexere Anforderungen sowie der Erweiterung der bisherigen Anforderungen wuchs auch der Bedarf an Regressionstests, der durch den manuellen Test nicht mehr abzudecken war. Daraufhin wurde die Testautomatisierung des Regressionstests mit der TOSCA-Testsuite erfolgreich in das Projekt integriert. Von der Einführung der Testautomatisierung profitiert auch der PO, da auch während des UAT-Zeitraums Regressionstests automatisiert ausgeführt werden können **(siehe Abbildung 4)**.

Das dritte und vierte Release war flächendeckend, sodass die Anwendung auf Last- und Performanz geprüft werden musste. Für den Last- und Performanztest wurde Jmeter genutzt, um die technischen Anforderungen zu testen.

Die guten Testergebnisse mit geringeren Defectmeldungen in den Abnahmetests sowie die erfolgreichen Releases spiegeln den Erfolg der Teststrategie wider. Es ist gelungen, mit dem Vier-Stufen-Modell Defects rechtzeitig zu identifizieren, zu beheben und die Qualität des Endprodukts zu sichern.

Ein Messwert, um die Entwicklungsqualität der Applikation und die Testergebnisse zu vergleichen, ist die Defect Density. Diese beschreibt die Anzahl der Defects pro Umgebung oder Teststufe auf 1000 Entwicklerstunden, sodass eine niedrige Defect Density bei 80 bis 100 Prozent Testabdeckung eine gute Entwicklungsleistung darstellt. In diesem Fall wird die Defect Density ab der zweiten Teststufe gemessen. Die Messwerte für den UAT (2,2 – 6,7 Defects/1000 Entwicklerstunden) liegen stets unter dem Durchschnitt von weiteren Accenture-Projekten gemeldeten Werten (< 30 Defects/1000 Entwicklerstunden).

Die Umsetzung der Shift-left-Teststrategie erfolgte stets in enger Zusammenarbeit des Testmanagements mit dem Delivery Manager, Scrum Master und Product Ownern des Projektteams. Das Vier-Stufen-Modell wurde in wiederkehrender Absprache nach Bedarf angepasst oder erweitert, sodass jede Partei in die Gestaltung der Testprozesse einbezogen wurde.

Fazit

In der Praxis hat diese Teststrategie hohe Anerkennung durch den Kunden erhalten und führte stets zu erfolgreichen Releases. Es erfolgten nach Go-Live wenige bis gar keine Incidentmeldungen, sodass auch hier eine Akzeptanz bei den Bankmitarbeitern geschaffen wurde. Die Teststrategie findet nun auch bei anderen agilen Projekten des Kunden Anwendung.

Thanh-Mai Le

thanh-mai.le@accenture.com
ist Quality Engineer und Advisor spezialisiert in agilem Testing, Testkoordination und Defect-Management sowie Testdaten und -umgebungsmanagement. Ihre Erfahrungen sammelte sie in verschiedenen Projekten bei Großkunden im Bereich Financial Services. Als Test Specialist hat sie mehrere Projekte begleitet und hierfür agile Testprozesse aufgesetzt. Aufgrund ihrer zusätzlichen bankkaufmännischen Ausbildung bringt sie zudem praktische Erfahrung sowie Fachkenntnisse im Anwendungsgebiet mit.

Alexander Fabritius

alexander.fabritius@accenture.com
ist ein Transformation Advisor für IT Delivery und organisatorischen Wandel. Seine Erfahrung reicht von der Entwicklung einer strategischen Neuausrichtung eines Unternehmens bis hin zur IT-Umsetzung digitaler Produkte. Er verfügt über fundierte Kenntnisse in den Bereichen IT-Governance und Prozessoptimierung sowie im Programm-Management.

»Verlaufen im Qualitätsdschungel?«

Der heutige Markt erfordert eine zeitnahe Anpassung an sich schnell ändernde Umstände. Für ein Softwareprodukt bedeutet dies eine schnelle Lieferung in höchster Qualität. Daher müssen bisherige Prozesse überdacht werden, sei es in Form von Strategien, Methodiken oder Tools. Der Qualitätsbegriff sollte dabei über Projekte und Prozesse hinaus aus ganzheitlicher Sicht betrachtet werden. Eine Einführung von Testautomatisierung beispielsweise ermöglicht zwar eine Kapazitätsfreigabe, stimmt jedoch das Kooperationsmodell zwischen Fachbereichen, Entwicklung und Test nicht, reicht der Testfokus nicht aus, um für Qualität zu sorgen. Daher ist es notwendig, die Wechselwirkung zwischen den Prozessen und Projekten aus übergreifender Sicht zu betrachten, um das Qualitätsniveau zu heben und die einzelnen Maßnahmen effektiver einzusetzen.


Photo by Jamie Street on Unsplash

Die Qualitätskriterien der Prozesse und Methoden sollten von der Umsetzung bis hin zur gesamten unternehmerischen Wertschöpfungskette durchgängig verknüpft sein, um die Effizienz der einzelnen Schritte immer in Verbindung zu deren Qualität sichtbar zu machen. Dies führt typischerweise zu folgenden Fragestellungen:

- Wie kann ich die übergreifende Qualität aller Maßnahmen bewerten?
- Welche verschwenderischen Aktivitäten können eliminiert werden?
- Welchen Erfolg hat die einzelne Maßnahme? Hat sich die Qualität verbessert?

Folgende Aussage des US-amerikanischen Ökonomen Peter Drucker:

- *„If you can't measure it, you can't manage it"*

erscheint auf den ersten Blick wie eine Binsenweisheit, findet aber in der praktischen Welt noch zu selten Anwendung.

Um ein Lean Quality Management System (QMS) einzuführen und die obigen drei Fragen beantworten zu können, werden Daten aus Bereichen vor allem ineffizienter Prozesse benötigt. Diese lassen sich zu Informationen zusammenfassen. KPIs (Key-Performance-Indicator) sind ein bewährtes Mittel, um Informationen transparent darzustellen. Doch welche Aspekte sollten wir betrachten?

Wir haben ein Softwareunternehmen begleitet, welches sich entschieden hat, das bestehende Produktportfolio anzupassen mit den daraus folgenden Geschäftszielen:

- Ziel 1: Adäquate Reaktionen auf Marktveränderungen
- Ziel 2: Kundenorientierte Produkte höchster Qualität

Zusammen haben wir basierend auf diesen Geschäftszielen folgende KPIs definiert:

- KPI 1: Produktivitätssteigerung
- KPI 2: Release Quality Index
- KPI 3: Time-to-Market

Abbildung 1 zeigt in einem KPI-Dashboard das Zusammenspiel von Geschäftszielen, KPIs und Metriken. Sie sind über eine Ampel-Darstellung visualisiert.

Basierend auf den KPIs wurden folgende operative Lösungsstrategien erarbeitet und bei unserem Kunden eingeführt, um die Geschäftsziele zu erreichen:

- Methode 1: Risikobasierte Testmethodik
- Methode 2: Agile Entwicklung
- Methode 3: Testautomatisierung

Das Besondere dieser drei Methoden ist, dass diese die ganzheitliche Qualität und übergreifenden Unternehmensziele im Kern der Systematik immanent haben und aus Lean-Management-Perspektive den Fokus auf effiziente Priorisierung richten, um Verschwendung zu vermeiden.

Die Umsetzung der drei Methoden ermöglicht somit anhand der gemessenen Metriken eine Steuerung, die sicherstellt, dass KPIs und damit die Geschäftsziele erreicht werden. Wird eine Einführung der KPIs ohne die Umsetzung der Methoden angestrebt, ist ein Erfolg fraglich.

Im Folgenden zeigen wir die Zusammenhänge zwischen Methoden und Metriken auf und begründen, warum diese entscheidend für eine erfolgreiche Umsetzung sind.

Methode 1: Risikobasiertes Testen

Risikobasierten Testen wird von uns als ganzheitlicher Ansatz genutzt, der das Testing über alle Prozesse im SDLC (Software Development Life Cycle) mit den Unternehmensrisiken verbindet und gleichzeitig den Testaufwand von komplexen Systemen konstant hält, da der Fokus zuerst auf risikobehaftete Bereiche gelegt wird. Dadurch werden schwerwiegende Fehler vermieden und so wird die Qualität weiter verbessert.

Um eine risikobasierte Testmethodik zu implementieren, sind folgende drei Schritte notwendig:

- Identifizierung der Produktrisiken,

- Risikoanalyse anhand von Eintrittswahrscheinlichkeit und Schadensausmaß zur Erstellung einer Risikopriorisierung,
- Planung, Design und Ausführung aller Qualitätsaktivitäten gemäß Priorisierung inklusive laufender Neuevaluierung des Risikos.

Gerade der letzte Schritt ist ungemein wichtig, damit wir nicht, als ein Extrem, ein überbordendes Quality Management haben, welches als einzige Aufgabe die Dokumentation und Nachverfolgung von Risiken hat, jedoch auch keine lange Liste von einmal definierten Risiken, die in der täglichen Umsetzung keine Rolle mehr spielen.

Aus diesem Grund ist der Einsatz folgender Metriken nützlich:

- prozentualer Anteil der zurückgewiesenen Failures,
- prozentualer Anteil an kritischen Failures (in Produktion),
- risikobasierte (Regressions-) Testabdeckung.

Methode 2: Agile Entwicklung

Agilität wird oft gleichgesetzt mit Geschwindigkeit. Jedoch bedeutet agiles Arbeiten unter anderem, auch das QA-Team schon von Beginn der Anforderungserfassung in den Prozess zu integrieren und zudem die Zusammenarbeit zwischen Entwicklung und Test zu stärken. Ebenso bilden ein kompetentes Team, offene Kommunikationskanäle und ein unterstützendes Umfeld die Kernpfeiler von agilem Arbeiten. Sie sind der fruchtbare Boden, auf dem die Performanz gesamthaft wächst.

Der Einsatz von Agilität in der Softwareentwicklung greift den ganzheitlichen Ansatz von oben auf, indem die risikobasierte Betrachtung des Testens zusätzlich auch die Betrachtung der Softwareentwicklung einbezieht. Agilität beherrscht die Handhabung von komplexen Systemen gut (minimum viable product und inkrementelles Vorgehen), integriert neue technologische Herausforderungen leichter und führt so auch zu einem Shift-Left der Gesamtaufwände – ganz im Sinne von Lean. Wieso einen Fehler beheben, wenn man diesen durch frühere Maßnahmen verhindern kann?

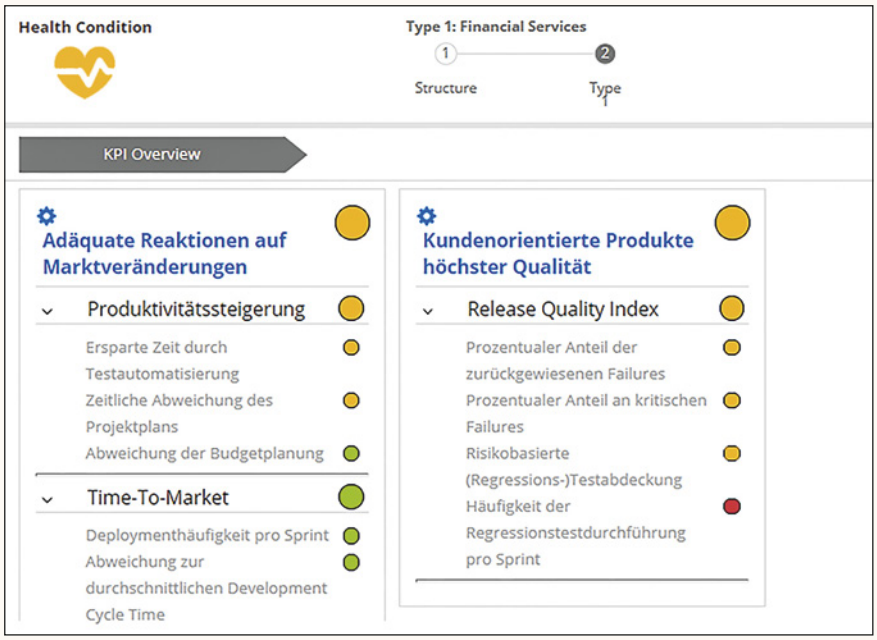


Abb. 1: Beispielhafter Ampelstatus in QACube. Die Daten werden je nach individuell definierter Gewichtung zum KPI aggregiert. Über eine Drill-Down-Funktion lassen sich jedoch auch die Metriken im Detail analysieren (Trends, Gleitende Mittelwerte usw.)

Wir möchten auf drei essenzielle Aspekte hinweisen, die bei uns in einem Agilen Framework mit Hinblick auf die Einführung von sinnvollen KPIs im Fokus stehen:

- iteratives und inkrementelles Vorgehen,
- effiziente Kommunikation,
- schnelle Anpassungsmöglichkeit und Feedback.

Damit ist der Einsatz folgender Metriken nützlich:

- Deploymenthäufigkeit pro Sprint,
- Abweichung der Budgetplanung, zeitliche Abweichung des Projektplans, Abweichung der Testaufwandsschätzung,
- Abweichung zur durchschnittlichen Development Cycle Time.

Methode 3: Testautomatisierung

Wendet man Lean-Prinzipien auf Methoden und Prozesse in der Testautomatisierung an, so lassen sich übergreifende Qualitätszuwächse durch Optimierung und Fokussierung erreichen.

Daher strukturieren immer mehr Unternehmen (wie auch Spotify, siehe: https://engineering.atspotify.com/2018/01/11/testing-of-microservices/) ihre Tests als Testing-Diamant **(siehe Abbildung 2)**. Der Fokus wird dabei auf ein Regressionstestset auf Integrationsebene gelegt. Dies vereinfacht die Pflege von Tests erheblich und verifiziert zudem die Interaktion mit anderen Modulen, denn dort, wo die Integration stattfindet, finden sich die meisten Probleme und das höchste Risiko. Auch in unserer eigenen Produktentwicklung für die QACube-Plattform befolgen wir dieses Prinzip und konnten eine Reduktion der Fehler im Integrationstest und verschwenderischen Wartungsaufwand auf Unittest-Ebene bei gleichzeitiger Steigerung der übergreifenden Qualität feststellen. Testautomatisierung dient in diesem Fall als langfristige Lösung, um zusätzlich die Effizienz im Testing zu erhöhen und damit weitere Kapazität zu schöpfen.

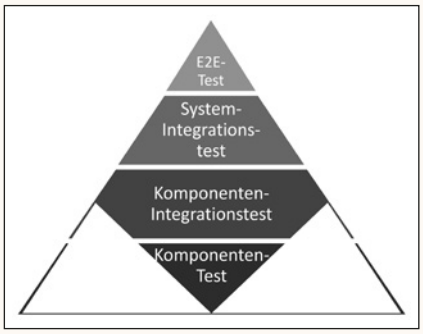


Abb. 2: Testing-Diamant

Somit ist der Einsatz folgender Metriken nützlich:

- ersparte Zeit durch Automatisierung,
- Häufigkeit der Regressionstestdurchführung pro Sprint.

Fazit

Bei dem Softwareunternehmen, welches wir begleitet haben, führten die stetig ganzheitliche Betrachtung sowie die durchgängige Messbarkeit auf verschiedenen Ebenen zu Verbesserungen, welche zudem von gegenseitigen Synergien profitierten.

Die erstmalige Messung von Geschäftszielen verknüpft bis zur Prozessebene verschaffte eine bisher nicht vorhandene Transparenz, was eine fundiertere und schnellere Entscheidungsfindung ermöglichte sowie das Verständnis unter den Teams verstärkte. Durch laufende Risikomessung konnten die begrenzten Ressourcen zur richtigen Zeit auf die richtigen Themen priorisiert werden. Dadurch konnte das Projekt ohne Budgetabweichungen durchgeführt werden. Nicht zuletzt hatte die Anpassung der Testautomatisierungsstrategie die Stabilisierung des Regressionstestsets zur Folge. Somit konnten mehr Fehler früher abgestellt und der Wartungsaufwand reduziert werden. Insgesamt konnten wir im Laufe der Transformation eine Verringerung der kritischen Produktionsfehler feststellen.

Das erste übergreifende Erfolgserlebnis als Ergebnis der Transformation konnten wir verzeichnen, als durch eine Analyse im Fachbereich ein Nachfragetrend für ein Teilprodukt entdeckt wurde. Diese Erweiterung wurde in vergleichsweise kurzer Zeit mit hoher Kundenzufriedenheit erfolgreich eingeführt. Eine solche Reaktionskapazität wäre zuvor undenkbar gewesen.

Zum Abschluss sei zu erwähnen, dass wir bei der Implementierung der KPIs auf Hindernisse gestoßen sind. Um diese zu umschiffen, haben wir folgende Lessons Learned formuliert.

Tipps zur effektiven Nutzung von KPIs

Bei der Einführung von KPIs gibt es keinen „one-size-fits-all"-Ansatz. KPIs sollten abhängig von der Ausgangssituation maßgeschneidert auf die bestehenden Prozesse und die Bedürfnisse aller Stakeholder der verschiedenen Hierarchiestufen zugeschnitten werden. Wir möchten auf vier Tipps hinweisen, welche zur Vermeidung von typischen Fehlerquellen bei der Einführung von KPIs dienlich sind.

1. Definition von Geschäftszielen und Aktionsplan für jede KPI

Übergeordnet der Definition jeder KPI sollte das Geschäftsziel stehen. Oft werden KPIs aus vorhandenen Informationen definiert, ohne sich zu fragen, was damit ausgesagt werden soll. KPIs werden so zu Zielen, die Manipulationen unterliegen können.

Daher sollte zuerst überlegt werden, welche Fragen man über die Darstellung einer KPI beantworten möchte, um danach zu prüfen, welche Metriken bei der Beantwortung der Frage hilfreich sind. KPIs stellen keine Ziele oder Lösungen dar, die man erreichen sollte, sondern Indikatoren für die übergeordneten Ziele. Sie helfen, Lösungen zu identifizieren.

Aktionsplan für ein Geschäftsziel:

- 1. Risikoanalyse und Bewertung
- 2. Durchführung aller Tests für die Funktionen mit hoher Eintrittswahrscheinlichkeit und hohem Impact vor dem Go-Live
- 3. Kontinuierliche Evaluierung und Verbesserung (z. B. Analyse der unerwartet aufgetretenen Risiken und Fehler)

2. Nutzung von Prozessperformanz-KPIs zusammen mit Qualitäts-KPIs

Die alleinige Nutzung von Prozessperformanz-KPIs bietet Raum für Fehlinterpretationen und Fehlkommunikation. Daher sollten diese in Kombination mit Qualitäts-KPIs genutzt werden, welche eine Indikation über den geschaffenen Wert aufzeigen:

- *Performanz-KPI:* Anzahl der durchgeführten Tests im Verhältnis zur Anzahl der geplanten Testfälle
- *Qualitäts-KPI:* Risikobasierte Testabdeckung

3. Ganzheitliche Interpretation von KPIs

KPIs sind nicht isoliert, sondern immer im Gesamtzusammenhang von Außeneinflüssen mit dem Fokus auf das eigene Unternehmen, die Umgebung und dem Gesamtmarkt zu interpretieren. Weitere wichtige Faktoren:

- Anzahl der kritischen Failures während der Sprints,
- blockierte Testfälle,
- Downtimes.

4. Automatisierung

Es ist es wichtig, dass das KPI-Framework, welches etabliert wird, automatisiert zur Verfügung steht und einfach zu pflegen ist, ansonsten besteht die Gefahr, dass der manuelle Aufwand, KPIs zu extrahieren, den Nutzen übersteigt oder zu Fehlern in der Auswertung durch Übertragungsfehlern führt.

Masud Sultan

masud.sultan@sixsentix.com
ist Quality Engineering Experte bei der Sixsentix Deutschland GmbH mit fachlicher und technischer Erfahrung im Finanzdienstleistungssektor und Expertise im Testing komplexer Datawarehouse-Plattformen im Rahmen von Datenmigrationsprojekten.

Daniel Pollig

daniel.pollig@sixsentix.com
ist Experte für Software-Testing bei der Sixsentix Deutschland GmbH mit Erfahrung in den Branchen Automobilindustrie und Maschinenbau; spezialisiert auf den Aufbau von Software-Testteams in Kombination mit Tool- und Prozessberatung.

»5 Gründe, warum Sie jetzt Ihren Test-Bot aus der Cloud treffen sollten«

Amazon, Google, Microsoft, Oracle, Salesforce oder SAP bewegen den digitalen Hub mit Riesenschritten in die Cloud. Aber was bedeutet das für Ihren Job in der Software-QS? Sie müssen immer früher, schneller und aufwendiger testen. Die Anforderungen steigen permanent. Dafür gibt es jetzt eine Lösung: Robotic Testautomation aus der Cloud. Hier sind fünf Gründe, warum Sie Ihren persönlichen Test-Bot aus der Cloud treffen sollten – starten Sie jetzt.

1. Er liebt Open Source.

Viele machen ihre ersten Erfahrungen in der Testautomation mit Open-Source-Tools wie Robot Framework, Selenium, Appium oder Jenkins. Auch wir haben so angefangen, denn Qentinel kommt aus Finnland und ist seit 15 Jahren an der Entwicklung von Robot Framework beteiligt. Und jetzt haben wir diese Tools mit einer Robotic Testing Plattform kombiniert, die Tempo, Stabilität und Usability extrem verbessert: Qentinel Pace.

2. Er ist extrem pflegeleicht.

Qentinel Pace ist eine vorkonfigurierte Testumgebung in der Cloud, die einen Start der Testpipeline innerhalb weniger Minuten möglich macht. Unsere Test-Bots sprechen PaceWords, eine einfache Sprache, die in allen Umgebungen – Web, Mobile, Desktop – funktioniert. Testskripte lassen sich in jeder Anwendung – auch in SAP oder Salesforce – einfach per drag & drop oder mit Live Recording erstellen. Damit können auch End-User Tests schreiben – probieren Sie es aus!

3. Er ist immer da, wenn Sie ihn brauchen.

Ist Ihre Testkapazität immer dann am Limit, wenn sie am dringendsten gebraucht wird? Zahlen Sie viel Geld für Lizenzen, die Sie nur ein paarmal im Jahr wirklich intensiv nutzen? In der herkömmlichen Softwareentwicklung ist das Testen meist Saisonarbeit. Bei Qentinel Pace bezahlen Sie die Kapazitäten nur dann, wenn Sie sie tatsächlich benötigen. Kurzfristiges Upscalen ist problemlos möglich. Wenn Sie Ihren Test-Bot mal nicht brauchen, verschwindet er einfach wieder in der Cloud.

Qentinel | Robotic Software Testing

„Wenn mich jemand fragt, was Cloud-Computing ist, sage ich ihm einfach, dass es eine bessere Art und Weise ist, dein Geschäft zu betreiben." Salesforce-Gründer Marc Benioff

4. Er achtet immer auf Ihre Sicherheit.

Manche denken immer noch, das eigene Netzwerk sei sicherer als die Cloud. Denn das Internet ist eine „freie Wildbahn", die ständig neue Bedrohungen hervorbringt. Und genau deshalb bietet Testautomation aus der Cloud immer die aktuellsten Sicherheits- und Zugriffskontrollen. Sie können auch hybride Setups nutzen, bei denen sensible Daten weiter hinter Ihrer Firewall bleiben, während das Testmanagement komfortabel aus der Cloud erfolgt. So sind auch smarte IoT-Lösungen möglich.

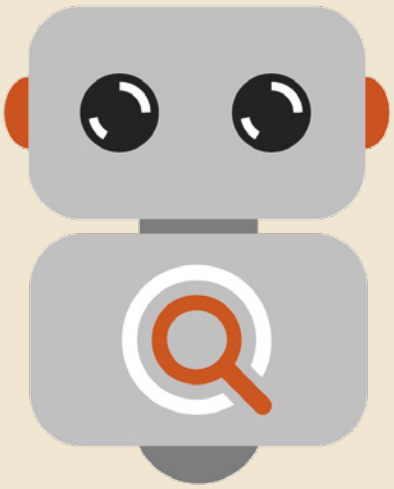
5. Er schenkt Ihnen mehr Zeit für die schönen Dinge im Leben.

Kontinuierliches Testen als Teil einer guten CI/CD-Pipeline ist ein echter Zeitfresser. Gut, dass unsere Test-Bots am liebsten das tun, was Ihnen am meisten Zeit raubt: langwierige E2E- oder Regressionstests. So können Sie sich auf das fokussieren, was Ihnen Freude macht, zum Beispiel Entwicklung und kreatives Testen. Ein weiterer Vorteil: Test-Bots streiten nicht mit den Entwicklern, meckern nicht wegen Überstunden und machen keinen Urlaub, wenn man sie am dringendsten braucht.

Ihr persönlicher Pace-Bot wartet schon auf Sie!

Probieren Sie jetzt unsere Testautomation aus der Cloud aus. Sie können sich erst ein paar Videos anschauen oder direkt einloggen und Ihren persönlichen Pace-Bot treffen, der ab sofort jeden Monat 60 Minuten kostenlos für Ihre Web-Tests da ist! Wenn Sie mehr wissen wollen, zögern Sie bitte nicht, uns zu kontaktieren*.

"If in doubt – do it." Grace Hopper



*) Hey, wir sind keine Chat-Bots, sondern Menschen, versprochen.

Videos: https://www.youtube.com/channel/UCB-UFeV-w6IDDDF0xKdanIQ
Freier Pace-Zugang: https://pace.qentinel.com
Freies Online-Training: https://qentinel.com/test-automation-online-training/
Preise: https://qentinel.com/de/preise/
Kontakt: https://qentinel.com/de/kontakt/
Mail: dirk.strubberg@qentinel.com

»Eine nachhaltige Verbesserung«

Softwarequalität ist längst nicht mehr die Verantwortung eines dedizierten Teams. Ganz im Gegenteil lässt sich Qualität als Gemeinschaftsaufgabe definieren. Um diese Veränderung in der Organisation zu verankern, ist ein Umdenken entscheidend. Anstelle vertikaler, siloartiger Organisationsstrukturen rücken cross-funktionale Teamstrukturen in den Fokus. Nur mit der Ende-zu-Ende-Verantwortung eines Teams für eine Funktionalität gelingt es, die Softwarequalität nachhaltig zu steigern.

Software ist integraler Bestandteil unseres Lebens. Selbst klassische, industrielle Produkte sind inzwischen mit integrierter Software ausgerüstet. Der Wertschöpfungsprozess jedes Unternehmens ist maßgeblich durch Software bestimmt. Nicht grundlos artikuliert Microsofts CEO Satya Nadella: „Every company is a software company" [Mic18]. Doch die Art und Weise, wie Organisationen weltweit Software produzieren, variiert signifikant. Das hat unmittelbare Auswirkung auf die Softwarequalität.

Doch welche Charakteristika kennzeichnen eine bewährte Softwareorganisation und welche Elemente bieten neue Gestaltungsoptionen im Rahmen einer kundenzentrierten Organisation? Ein traditionelles, weitverbreitetes Muster ist eine funktionale Organisation der Softwareentwicklung. Im Sprachgebrauch wird bei dieser Aufstellung häufig von vertikalen Silos gesprochen **(siehe Abbildung 1)**. Die Arbeitsorganisation erfolgt anhand des Aufgabenschwerpunkts eines Mitarbeiters. Mitarbeiter mit dem gleichen Aufgabenschwerpunkt bilden ein homogenes Team. Die Teams werden entlang des klassischen Wertschöpfungsprozesses der Softwareentwicklung, beginnend mit dem Anforderungsmanagement bis hin zum Betrieb und Service-Management, in der logischen Reihenfolge angeordnet.

Durch Homogenität und eine sequenzielle Arbeitsweise entstehen in sich geschlossene Systeme. Die Schnittstellen zum vor- und nachgelagerten Wertschöpfungsschritt werden verwaltet. Eine systemische Betrachtung des gesamten Wertschöpfungsprozesses entfällt hingegen für gewöhnlich. Folglich wird häufig ein Testmanagement- oder Qualitätssicherungsteam mit der Verantwortung für die Qualität der bereitgestellten Software betraut. Es allein übernimmt die Verantwortung für eine Gemeinschaftsaufgabe, an der eine Vielzahl weiterer Beteiligter partizipiert.

Im Kontrast zu dem arbeitsteiligen, klassischen Vorgehen steht die Organisation der Softwareentwicklung entlang eines horizontalen Wertflusses in sogenannten Feature-Teams. Alle Mitarbeiter, die an der Wertschöpfung einer Funktionalität beteiligt

sind, formen ein Team. Dieses ist folglich durch Interdisziplinarität gekennzeichnet. Es übernimmt die Ende-zu-Ende-Verantwortung für den Erfolg oder Misserfolg der Funktionalität.

Charakteristika eines erfolgreichen Feature-Teams

Doch welche Eigenschaften kennzeichnen ein erfolgreiches Feature-Team? Anhand von zehn charakterisierenden Attributen lässt sich dies kompakt beschreiben **(siehe Abbildung 2)**.

Verantwortlich

Unter Ende-zu-Ende-Verantwortung versteht der Autor das Zusammenführen von Teilverantwortungen ehemals funktional getrennter Teams zu einem gemeinsamen Team, welches die Gesamtverantwortung für eine Software trägt.

Die Übernahme von Ende-zu-Ende-Verantwortung mit dem übergeordneten Ziel der Qualitätssteigerung setzt ein systemisches Denken voraus. Ein Vorreiter des systemischen Managementansatzes ist William E. Deming, der eine Organisation als System und damit als gesamtheitlichen Organismus versteht. Aus diesem Verständnis lässt sich ableiten, dass das Optimieren einzelner, funktionaler Silos niemals zu dem Gesamtoptimum des Systems führt. Es führt lediglich zu partiellen Verbesserungen.

In der Praxis ist es besonders herausfordernd, dass bislang vielerorts im funktionalen Maßstab gehandelt wird, sodass ein radikales Umdenken notwendig ist. Ein System und damit die Qualität seiner Erzeugnisse lässt sich schließlich nur dann optimieren, wenn wahrlich gesamtheitlich gehandelt wird und keine funktionalen Organisationsbarrieren einem systemischen Management im Wege stehen. Ein Feature-Team übernimmt die Verantwortung und setzt sich für die gesamtheitliche Systemverbesserung ein.

Divers

Inzwischen ist es eine anerkannte Tatsache, dass diverse, interdisziplinäre Teams die besseren Teams sind. Dabei entsteht die Vielseitigkeit des Teams in vielerlei Dimensionen. Zunächst sind hierbei die personenbezogenen Attribute, wie das Alter und das Geschlecht, zu nennen.

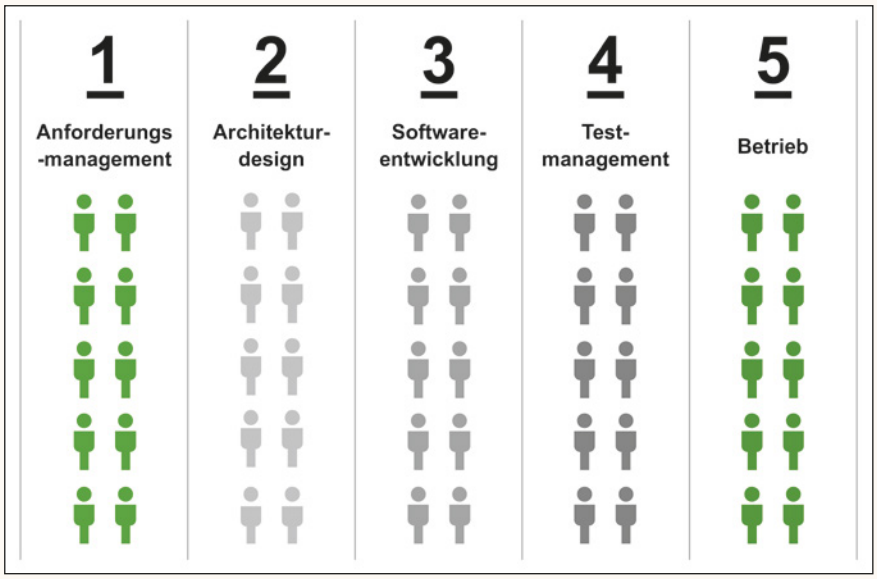


Abb. 1: Arbeitsteilige, funktionale Softwareentwicklungsorganisation

Interdisziplinär

Darüber hinaus ist ein diverses Team nachweislich leistungsstärker, wenn es zusätzlich über eine vielseitige Wissensbasis verfügt. Liang et al. haben diesen Performanzeffekt in Softwareprojekten empirisch nachgewiesen [Lia07]. Folglich ist in Feature-Teams die Diversität auch im Hinblick auf das jeweilige Fachgebiet ein Erfolgsfaktor. Die Zusammenarbeit von Experten mehrerer Disziplinen ist äußerst ratsam. Schließlich sind Feature-Teams in der Softwareentwicklung mit komplexen Herausforderungen betraut. Hierfür ist ein kreativer Problemlösungsansatz zu wählen, der die Expertise aus zahlreichen Disziplinen erfordert.

Cross-funktional

Ein entscheidender Vorteil hierbei ist, dass die einzig wahre Übergabe (engl. handover) zwischen dem Feature-Team und dem Kunden stattfindet. Übergaben zwischen funktionalen Silos und damit potenzielle Nadelöhre (engl. bottlenecks) und Risiken entfallen. Dies minimiert die Gefahr, im Rahmen unternehmensinterner Übergaben Qualitätseinbußen zu verursachen. Auslöser hierfür können beispielsweise eine unzureichende

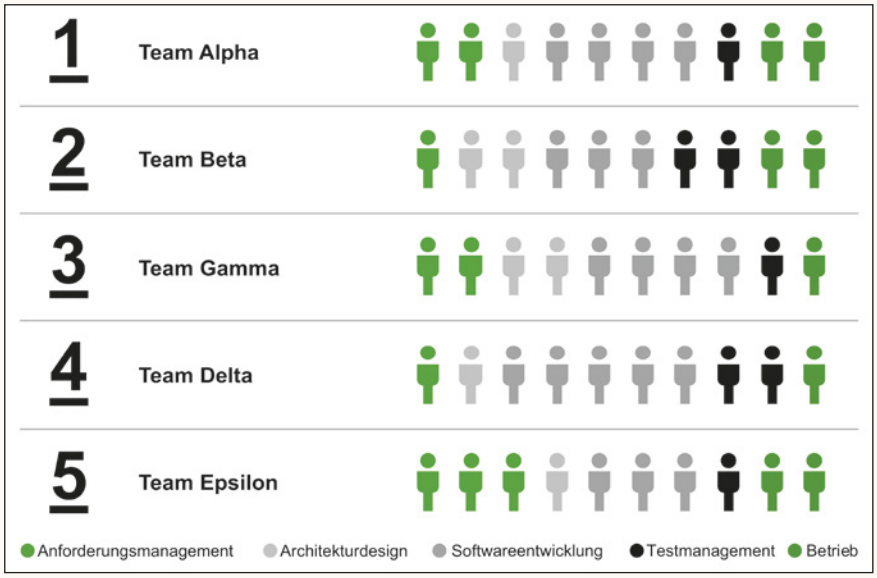


Abb. 2: Cross-funktionale Feature-Teams

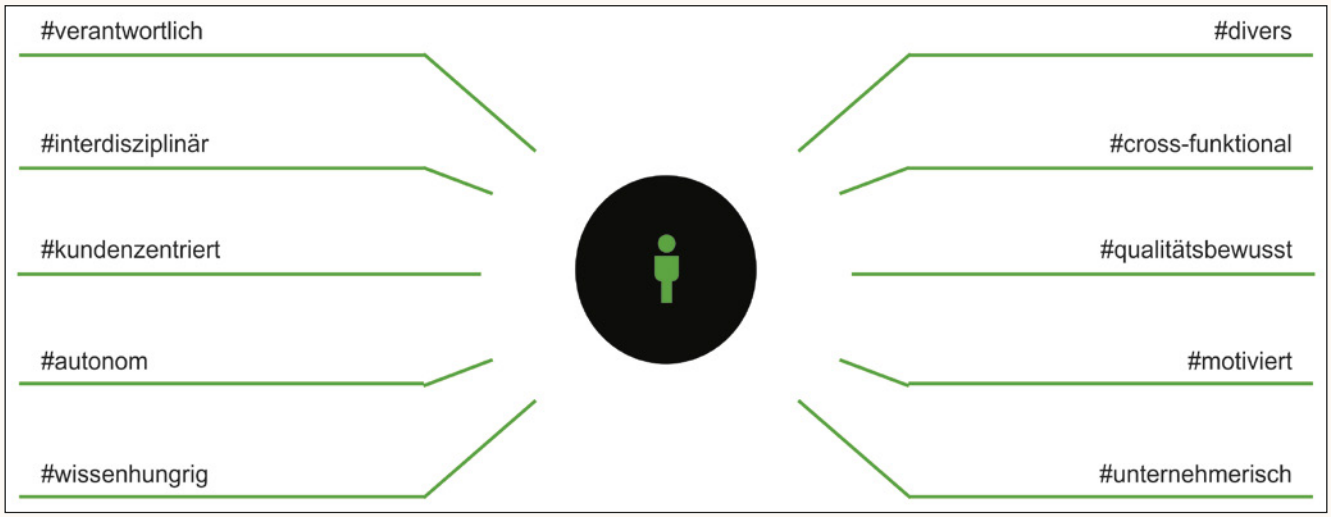


Abb. 3: Zehn Charakteristika eines erfolgreichen Feature-Teams

Kommunikation, eine wenig ausgeprägte Zusammenarbeitskultur oder unabgestimmte Zielparameter und -vereinbarungen sein.

Das cross-funktionale Team hingegen verantwortet die Software entlang des gesamten Produktlebenszyklus. Es inkludiert sämtliche notwendigen Fähigkeiten, um das gewünschte Produkt zu entwickeln. Im Idealzustand sitzen metaphorisch gesprochen Anforderungsmanager, Architekten, Softwareentwickler, Testmanager, Operations Engineers, IT-Service-Manager und weitere erforderliche Rollen im gleichen Boot. Sie bilden ein Team **(siehe Abbildung 3)** mit der Ambition, den Kunden zu begeistern. Denn nur daran wird das Team gemessen.

Jedoch erweist sich Cross-Funktionalität in der Realität oftmals als herausfordernd. Gewisse Fachexperten stehen womöglich nicht in ausreichender Anzahl je Team zur Verfügung, sodass alternative Kollaborationsmöglichkeiten zu formen sind. Ihr Know-how muss anteilsmäßig mehreren Teams parallel zur Verfügung stehen, womit die Aufgabenverteilung für diese Fachexperten geschäftskritisch ist.

Kundenzentriert

Ein signifikanter Vorteil der Organisation in Feature-Teams ist die explizite Ausrichtung am Endkunden. Schließlich beginnt die Wertschöpfung initial mit der Anforderungserhebung des Kunden und erstreckt sich bis hin zu einem Release und dem anschließenden Betrieb der Software. Eine qualitativ hochwertige Entwicklung ist durch die enge und kontinuierliche Einbindung des Endkunden sichergestellt.

Diese (Neu-)Ausrichtung am Kunden wird jedoch aus unterschiedlichsten Gründen nicht allen Feature-Teams gelingen. In diesen Fällen ist über stellvertretende Lösungen wie die Repräsentation durch einen Product Owner nachzudenken.

Qualitätsbewusst

Um das Bewusstsein für Qualität nachhaltig aufrechtzuerhalten, empfiehlt sich die bewusste Anpassung des Zielsystems. Ehemals galten für die funktionalen Silos Teamziele, die ein lokales Optimum erzeugten. Diese sind im Rahmen einer Organisation entlang von Wertströmen anzupassen. Dabei ist eine Akzentuierung von kundenzentrierten Parametern ratsam. Dazu zählen insbesondere die Kundenzufriedenheit und die Kundenbindung (engl. customer retention). Diese drücken die wahrgenommene Qualität aus Sicht des Kunden aus und entscheiden über den Markterfolg der Software. Damit sind kundenzentrierte Kennzahlen erfolgsentscheidender als die technischen Key Performance Indicators zur Qualität der Software.

Autonom

Eine Herausforderung bei der Gestaltung von Feature-Teams stellt deren Freiheitsgrad dar. Aufgrund manifestierter Hierarchien und Machtstrukturen in Organisationen ist es ein komplexes Unterfangen, weitestgehend autonome Feature-Teams zu formen. Die Autonomie eines Feature-Teams sorgt derweil für zweierlei Vorteile. Zuerst ist hervorzuheben, dass Entscheidungen stets in dem Personenkreis getroffen werden, der sich mit der Problemstellung am besten auskennt und daher die Implikationen der Entscheidung am treffendsten überblicken kann.

Motiviert

Ferner steigt mit der Übernahme von Entscheidungsverantwortung in der Regel auch das Maß an Motivation. In wenigen Fällen wird Eigenverantwortung zwar auch als Last und Bürde wahrgenommen, jedoch handelt es sich de facto um die Aufwertung der eigenen Rolle, was von einer Vielzahl an Mitarbeitern als ausgesprochen positiv und motivierend bewertet wird. Schließlich ist das Team für sämtliche Entwicklungen eigenverantwortlich und steckt entsprechend viel Engagement in die optimale Entwicklung der eigenen Lösung. Es herrscht eine ausgeprägte Identifikation des Feature-Teams mit ihrem eigenen Produkt.

Wissenshungrig

Im Rahmen des intensiven Wissensaustauschs innerhalb eines Feature-Teams besteht die Ambition, die Teammitglieder langfristig zu sogenannten „T-shaped-Know-how-Trägern" zu entwickeln. Die waagerechte Linie des Buchstabens „T" symbolisiert das breit aufgestellte Wissen in einer Vielzahl an angrenzenden Fachgebieten, während die senkrechte Linie die tiefgreifende Expertise in dem persönlichen Spezialgebiet abbildet. Insbesondere die horizontale Achse profitiert von dem Wissensaustausch

Referenzen

- [Lia07] T.-P. Liang et al., Effect of team diversity on software project performance, in: Industrial Management & Data Systems, 2007, Vol. 107, No. 5, pp. 636 ff., siehe: **https://www.researchgate.net/profile/Ting-Peng_Liang/publication/220672188_Effect_of_team_diversity_on_software_project_performance/links/09e4150ae64316ccb4000000.pdf**
- [Mic18] Microsoft Inc., Microsoft CEO Satya Nadella on fueling 'tech intensity' in the UK, 2018, siehe: **https://news.microsoft.com/en-gb/2018/11/07/microsoft-ceo-satya-nadella-on-fuelling-tech-intensity-in-the-uk/**

im Feature-Team, sodass langfristig Generalisten ausgebildet werden, die notfalls einen ausfallenden Kollegen kompensieren können.

Unternehmerisch

Aus diesem Grund lässt sich ein Feature-Team in gewisser Weise als Unternehmen im Unternehmen charakterisieren. Die Teammitglieder treffen Entscheidungen, die für den Fortbestand der Software von existenzieller Bedeutung sind. In der Konsequenz daraus resultiert unmittelbar auch der betriebswirtschaftliche Erfolg. Die unternehmerische Gewinnerzielungsabsicht dient dabei als geeigneter Gradmesser. Einem Feature-Team sollte es entsprechend gelingen, mithilfe ihrer entwickelten Software ein langfristig rentables Geschäft aufzubauen, dessen Erfolg anhand von Finanzkennzahlen nachweisbar ist. Dies gilt im Erfolgsfall. Umgekehrt ist das Feature-Team unmittelbar für den Misserfolg und das Scheitern eines Service verantwortlich und muss im Zweifel die eigenen Versäumnisse aufarbeiten.

Weitere organisatorische Verbesserungsmaßnahmen

Doch im Zuge einer umfassenden Transformation gilt es, Herausforderungen des Changemanagements nicht außer acht zu lassen. Die Veränderung, fortan in Wertströmen anstelle in funktionalen Verantwortungsbereichen zu denken und handeln, kommt einer Zeitenwende gleich. Vieles, was bisher als selbstverständlich angesehen wurde, wird durch die neue Ausrichtung obsolet. Ein solch gravierender Eingriff in die bisherige Organisation und Ordnung bedeutet eine massive Veränderung für jeden Mitarbeiter. Es bedarf Zeit und Muße, sich der neuen Philosophie anzupassen und Hürden der Transformation zu überspringen.

Weiterhin ist die Ausgangssituation für die Reorganisation entscheidend. Hierbei lässt sich zwischen einem Greenfield- und einem Brownfield-Ansatz differenzieren. Während die grüne Wiese die perfekte Möglichkeit bietet, die Funktionalität, die Architektur und damit auch das Feature-Team zu gestalten, so steigt der Komplexitätsgrad der Transformation bei einem Brownfield-Ansatz. Die Etablierung von Feature-Teams muss in Einklang mit der bestehenden Organisation und Prozess- und IT-Landschaft geschehen, wodurch diverse Abhängigkeiten und Herausforderungen vorbestimmt sind.

Doch ganz gleich, welche Herausforderungen die Transformation begleiten und ob der Greenfield- oder Brownfield-Ansatz verfolgt wird: Die Transformation zu autonomen und kundenzentrierten Feature-Teams mit Ende-zu-Ende-Verantwortung ist erstrebenswert. Sie sind anpassungsfähig und flexibel und liefern nicht zuletzt qualitativ hochwertige Software aus. Der Kunde wird dies honorieren.

Niklas Kirfel

niklas.kirfel@cassini.de
ist Senior Consultant bei der Cassini Consulting AG. Als zertifizierter SAFe Program Consultant gestaltet er agile Organisations-, Transformations- und Softwareprojekte seiner Klienten.

Capgemini | sogeti Part of Capgemini

World Quality Report

2020-21 | TWELFTH EDITION

Die Erwartungen an die Qualitätssicherung steigen weiter.

Laden Sie sich Ihr Exemplar kostenfrei herunter und erfahren Sie mehr über wichtige Entwicklungen und weltweite Trends in Qualitätssicherung und Softwaretest.

www.sogeti.de/wqr2020

In association with: MICRO FOCUS

»Grundlagen und Best Practices aus dem Cloud Performance Engineering«

Neben vieldiskutierten technologischen Trend-Themen wie KI, Blockchain, Edge Computing oder Hyperautomation ist das Thema Cloud-Computing aktuell bei vielen Unternehmen aller Branchen der zentrale Baustein der IT-Strategie für die kommenden Jahre.

Die Vorteile von Flexibilität, Agilität, Geschwindigkeit und Kosteneffizienz sind für viele CIOs in diesen Tagen nicht mehr wegzudenken. Während die ersten Unternehmen noch an der grundlegenden IT-Cloud-Strategie feilen und grundlegende Entscheidungen wie die Deployment-Strategie (Public/Hybrid/Private) diskutieren, haben andere Firmen bereits die ersten Lift-and-Shift-Migrationsprojekte in die Cloud erfolgreich absolviert. Wieder andere Unternehmen sind noch einen Schritt weiter. Sie sind bereits aktiv dabei, ihre Applikationen für die Cloud zu customizen, und nutzen die von den Cloud-Anbietern zahlreich angebotenen PaaS-Services, um weitere Nutzenpotenziale zu heben.

Diese Unternehmen sind es auch, die die Kernherausforderungen der Cloud wie Sicherheit, Datenintegrität, Resilienz und funktionale Integration hinter sich lassen und sich einem Hauptaspekt des Cloud-Computings widmen: der Entmystifizierung der Cloud Performance.

Im Rahmen der Aktivitäten rund um Cloud Performance Testing oder vielmehr Cloud Performance Engineering stellt sich die Frage, welche Performance das migrierte und adjustierte System gegenüber einer reinen On-premise-Landschaft erreichen kann – weniger, gleich viel oder gar mehr Leistung?

Skalierungsmöglichkeiten der Cloud

Mit Verweis auf die dynamischen Skalierungsmöglichkeiten der Cloud wird häufig die Meinung vertreten, dass durch vergleichsweise einfaches Zuschalten von Hardwareressourcen die Performance in der Cloud ohne großes Zutun linear skaliert. Dies ist mit Sicherheit auch der erste zu überdenkende Schritt. Ähnlich zum On-premise-Vorgehen nutzt man zuerst die Möglichkeiten der vertikalen Skalierung und erhöht klassische Hardwarekapazitäten wie CPUs sowie RAM.

Nicht selten stößt man hier jedoch bei IT-Systemen größerer Konzerne mit hohen Durchsätzen, Aufrufen und Lastspitzen zu Stoßzeiten an seine Grenzen. Auch eine aggressive Wachstumsstrategie, mitunter gepaart mit nicht optimierten Applikationen, können Gründe sein, dass die IT-Anforderungen das Mooresche Gesetz überholen und schlichtweg die geforderten Maschinen noch nicht verfügbar sind.

Nun kann man einerseits hoffen, dass die nächste Hardware-Generation bereits kurz bevorsteht und über die Cloud-Anbieter bald bestellt werden kann. Andererseits rückt aber auch das Thema einer horizontalen Skalierung gezielt in den Fokus, also die Hinzunahme zusätzlicher Server für denselben Teil der Applikation. Dies erfordert wiederum in vielen Fällen – ähnlich zu On-premise-Systemen – einen größeren Umbau der Applikation selbst. Insbesondere bei Datenbanken bedarf es eines durchdachten Konzepts, um die Daten auf unterschiedlichen Servern autark zu persistieren.

Auch hier gibt es gegebenenfalls einen Ausweg für Applikationen mit hohen Read-only-Transaktionen. Um die gesetzten Performance-Ziele auch ohne „echte" horizontale Skalierung zu erreichen, kann die Nutzung bestimmter PaaS-Services eine Lösung sein. Beispielsweise bietet Microsoft hier den sogenannten Hyperscaling-Service für SQL-Datenbanken an, der über Caching-Mechanismen die Compute-Power dynamisch skaliert und auf sogenannte Read Replicas als Abbild der Datenbank horizontal verteilt. Auch die AWS Cloud bietet beispielsweise Read Replicas für RDS MySQL, PostgreSQL, MariaDB sowie Amazon Aurora an, und die Kollegen der Oracle Cloud setzen auf ihr altbekanntes Zugpferd Oracle RAC.

Ferner bietet Cloud Performance Engineering zusätzliche Möglichkeiten über vertikale oder horizontale Skalierungsmöglichkeiten hinaus. Viel Altbekanntes aus der On-pre-

mise-Welt hat hierbei nach wie vor auch in der Cloud seine Gültigkeit. Allen voran steht das Thema Index-Optimierung, welches in über 80 Prozent der Performance-Aktivitäten maßgeblich die IO-Performance definiert beziehungsweise im Bedarfsfall verbessert. Schon ein einziger fehlender Index kann die Gesamt-Performance des IT-Systems auf ein Minimum reduzieren. Somit sollte das Thema Datenbank-Indizierung immer ganz oben auf der Prioritätenliste jedes Cloud Performance Engineer stehen.

Darüber hinaus können Themen rund um Transaktionsklammern und Session-Handling mit Definition der maximalen Batchgröße, der Verbindungsdauer, der Lesehäufigkeit, der Idle-Zeit sowie eines möglichen Verbindungspoolings beispielsweise von SSL-Verbindungen ausschlaggebend sein für die Leistung des Systems, um nicht bei jeder HTTPS-Anfrage eine neue Verbindung zu öffnen und somit die CPU des eigenen Systems sowie des Schnittstellenpartners unnötig zu belasten.

Grundsätzlich sind zudem die Aufrufe der Datenbank zu minimieren und Caching-Mechanismen aktiv zu nutzen. Ebenso können die Anzahl an Instanzen, Threads und die Hardware selbst variieren, bis man die selbst gesteckte Definition of Done erreicht hat.

Elastizität

Cloud Performance Engineering geht zudem über die Tests oben genannter Skalierungsmöglichkeiten hinaus. Eines der Cloud-Versprechen ist die Funktionalität einer vollautomatisierten Elastizität, die Ressourcen dynamisch auf den jeweiligen Bedarf anpasst. Die zu überspringende Hürde ist der Umstand, dass viele On-premise-Applikationen bisher auf rein statische Installationen ausgelegt sind und für die dynamische Skalierung erst einmal fit gemacht werden müssen.

Hierfür sind aus Testsicht unterschiedliche Testszenarien zu definieren und im Zusammenspiel mit der Cloud zu verproben. Der Fokus sollte dabei weniger auf die Funktionalität der Cloud selbst als vielmehr auf das Zusammenspiel von der Cloud mit der eigenen Applikation hin gelegt werden. Prüfpunkt ist, inwiefern diese auf die dynamischen Skalierungen der Cloud reagiert, und ob beispielsweise Verbindungsabbrüche oder sonstige Verhaltensauffälligkeiten entstehen oder das System generell Performance-Einbußen erleidet.

Die Cloud-Anbieter bieten hierbei verschiedene neue Möglichkeiten zum schnellen Aufsetzen von Testumgebungen und Analysen von Performance-KPIs an, die zur Laufzeit analysiert und bewertet werden können. Je nach Bedarf ist ein Instrumentenmix aus bereits vorhandenen Testtools und neuen Testmöglichkeiten in der Cloud zu wählen, der das geplante Testkonzept bestmöglich abdeckt.

Statt eines intensiven Customizings von Altapplikationen sollte zuletzt auch immer die Alternative eines Neubaus in Betracht gezogen werden. Dieser kann nützlich sein, wenn sich eine große Anzahl fachlicher, nicht-funktionaler oder technischer Anforderungen ergeben, die sich mit dem aktuellen System nicht oder nur schwer realisieren lassen.

Jegliche Aktivitäten rund um die Cloud wie auch Cloud-Performance-Verbesserungen werden hierbei von einer modernen IT-Organisation begünstigt, die die Vorteile von agilen Verfahren, modularer Container-Architektur oder Time-to-Market-Konzepten wie CI/CD-Pipelines erkannt und auf den Weg gebracht hat. Meistens ist es sinnvoll, derartige Maßnahmen im Vorfeld umzusetzen, bevor man den Schritt in die Cloud geht.

Fazit

Abschließend lässt sich festhalten, dass die Migration in die Cloud zwar mit vielen Vorteilen und Möglichkeiten verbunden ist, jedoch gerade das Cloud Performance Engineering eine Herausforderung darstellt, die es mit teils bewährten, teils neuen Methoden zu meistern gilt. Der naiven Vorstellung einer automatisch nutzbaren freien Skalierbarkeit in der Cloud muss für große Unternehmen vielfach widersprochen werden. Vielmehr sind Zeit und Budget für Machbarkeitsstudien im Vorfeld sowie notwendige Anpassungen während der Realisierung einzuplanen. Empfehlenswert ist hierbei auch ein gutes Monitoring, um möglichst viele Erkenntnisse zu sammeln und die Fehleranalyse zu erleichtern. Über das Cloud Performance Engineering hinaus werden auch Testaktivitäten insbesondere in den Bereichen Resilienz, Datenintegrität und Sicherheit immer wichtiger, um eine immer komplexere und umfangreichere Cloud-Welt zu meistern.

In jedem Fall ist eine gute Zusammenarbeit aller beteiligten Personen wie Vorstand, CIO, Architekten, Cloud-Experten und Performance-Engineering-Experten hilfreich, um die Migration in die Cloud gemeinsam zum Erfolg zu bringen und das Thema Cloud Performance erfolgreich zu entmystifizieren.

Florian Fürst

florian.fuerst@accenture.com
ist IT-Unternehmensberater bei Accenture. Als Projektleiter leitet er aktuell internationale Großkundenprojekte und Cloud-Migrationen im Logistikumfeld. Als DACH-Leiter des Bereichs Cloud Testing treibt er das Cloud Quality Engineering in engem Verbund mit weltweiten Experten voran. Im Jahr 2019 referierte er beim Suisse Testing Day in der Samsung Hall Zürich über das Thema Blockchain Testing.

Tim Pillath

tim.pillath@accenture.com
ist IT-Unternehmensberater bei Accenture. Er ist fester Bestandteil des Bereichs Cloud Testing im DACH-Verbund. Als Testmanager in internationalen Großkundenprojekten entwickelt er neue Cloud-Test-Strategien und -Konzepte für verschiedenste Teststufen. Schwerpunkte sind das agile Testen mit Kanban und Konzepte zur Testautomatisierung im Bereich von Java-Custom-Build-Applikationen.

»Mensch, Robot! Testen bei 1&1 Telecommunications SE«

Bei der Testing Unit der 1&1 Telecommunications SE sorgen schnelle Release-Zyklen einer geschäftskritischen Software für ein hohes Arbeitsaufkommen. Das lässt sich stemmen, wenn Software-Roboter assistieren.

Abb. 1: Die 1&1 Testing Unit liebt Bots: Maskottchen Emil

Telefonisch oder per E-Mail: „Sie erreichen uns 24/7, an 7 Tagen die Woche", verspricht die 1&1-Website. Kundenservice ist Kerngeschäft des deutschen DSL- und Mobilfunkanbieters, das firmeninterne Customer-Relation-Management-System (CRM) eine topkritische Anwendung: Hier verwalten die Mitarbeiter im Fachbereich Customer Care täglich alle eingehenden Anfragen und mehrere Millionen Kontakte. Funktioniert das CRM-System nicht, stagnieren sämtliche kundenorientierten Prozesse.

Herausforderung: Systemstabilität bei kontinuierlicher Veränderung

Für 1&1 ist der stabile Betrieb der Kunden-Management-Software also absolut geschäftskritisch. Gleichzeitig erfordert die Dynamik der Prozesse im Customer Care eine häufige Anpassung der digitalen Umgebung, laufend entstehen neue Anforderungen an das CRM-System. Deswegen wird die Software kontinuierlich weiterentwickelt: In zweiwöchentlichen Intervallen erscheinen neue Releases mit Erweiterungen und Optimierungen.

Ein Update ist gut, aber nur, wenn die Software funktioniert, alle Anforderungen erfüllt – und den laufenden Betrieb nicht stört. Damit jeder Rollout in die Fachabteilungen sicher über die Bühne geht, unterhält die 1&1 eine eigene Testing Unit: „Wir sorgen für einen sauberen Go Live der neuen Version unseres CRM-Systems", erklärt Andreas Förch, der die entwicklungsnahe Abteilung für Software-Qualitätssicherung (Quality Assurance, kurz: QA) leitet. Vier hochqualifizierte Software-Tester führen umfassende Prüfungen auf unterschiedlichen Teststufen durch, „damit die Kollegen im Customer Care mit jedem neuen Release reibungslos weiterarbeiten können."

Kurz vorm Release: So wenig Zeit, so viel zu tun

Hinter dieser entspannten Beschreibung des Aufgabenfelds der Testing Unit verbirgt sich jede Menge Qualitätsarbeit: Klick für Klick prüfen und dokumentieren die Mitarbeiter, ob die Software die Feature Requests aus den Fachbereichen korrekt umsetzt. Sind die neuen Funktionalitäten separat getestet, stellen Systemintegrationstests sicher, dass die Software gesamthaft funktioniert. Finale Regressionstests in einer produktionsnahen Umgebung sollen dafür sorgen, dass sich das neue Release nahtlos und störungsfrei in den Livebetrieb integriert.

Die schnellen Release-Zyklen sorgen dabei für ein hohes Testaufkommen bei enormem Zeitdruck, eventuell verstärkt durch Verzögerungen in der Softwareentwicklung, die die Software-QA bis zum Release Date kompensieren muss: Alle diese Faktoren erzeugen mit jeder neuen Version des CRM eine regelmäßige Spitzenlast auf den Kapazitäten des lokalen Test-Teams.

Automatisierung als Ressourcen-Booster

Besonders die regelmäßigen Abnahmetests des gesamten CRM-Systems können die vier Mitarbeiter der Testing Unit allein nicht stemmen. Es ist schier nicht möglich, innerhalb der kurzen Zeit einer Testphase bis zum Go Live jede denkbare Klickstrecke in der komplexen Anwendung und die Kompatibilität mit dem produktiven System vorab zu prüfen. Der Verzicht auf umfassende Regressions- und Abnahmetests ist aber keine Option. Jedes nicht vollumfänglich geprüfte Release birgt Fehlerrisiken, und jeder Fehler im digitalen System wirkt sich empfindlich auf die Geschäftsabläufe bei 1&1 aus.

„Damit wir trotz Zeitdruck und Arbeitsvolumen immer pünktlich zu jedem Release-Termin wirklich umfassend getestete Software ausliefern können, müssen wir unsere vorhandenen Ressourcen effizient skalieren", beschreibt Abteilungsleiter Andreas Förch die größte Herausforderung im Daily Business. Und wie skaliert die 1&1 Testing Unit? „Ganz einfach", erklärt Förch, „wir automatisieren."

Software-Roboter als Teil des Teams

Während Automatisierung für viele ein Reizwort darstellt, das dystopische Fantasien nach dem Erzählmuster Mensch-gegen-Maschine heraufbeschwört, arbeitet das QA-Team bei 1&1 schon längst in friedlicher und produktiver Koexistenz mit sogenannten Software-Robotern. Seit drei Jahren beschäftigt die Testing Unit neben ihren hochqualifizierten Fachkräften zusätzlich „digitale Kollegen", um das geschilderte Ressourcenproblem elegant zu lösen.

Ein Software-Roboter ist kein klassischer Blech-Automat mit starken Greifarmen, wie sie aus der industriellen Produktion bekannt sind, sondern eben: Software. Genauer: ein Programm, das die Aktivitäten menschlicher User auf dem grafischen User Interface (GUI) nach einem vorgegebenen Workflow beziehungsweise Skript automatisiert ausführt. Schaut man einem Software-Roboter bei der Arbeit zu, kann man sehen, wie sich die Maus wie von Zauberhand über den Screen bewegt, wie Elemente angeklickt und Zeichen eingegeben werden. Solche digitalen Assistenten erledigen alle GUI-Tests, die einem standardisierten Ablauf folgen, deutlich schneller als ihre menschlichen Kollegen, immer fehlerfrei und „zu Uhrzeiten, bei denen eigentlich niemand mehr arbeiten mag", lobt Abteilungsleiter Andreas Förch die virtuelle Verstärkung seiner Belegschaft.

Mit Automatisierung einfach mehr und besser testen

„Die Software-Roboter nehmen uns wiederkehrende Testaktivitäten ab und bewältigen schnell und präzise eine enorme Menge an Testfällen", beschreibt Andreas Förch die Zusammenarbeit mit den Co-Bots. Die Effekte der Automatisierung im digitalen Raum sind direkt vergleichbar mit den aus der Industrie bekannten Wirkungen: Erhöhung der Durchsatzleistung, Standardisierung und Optimierung der Qualität, Fehlerreduktion, Entlastung von schwerer und monotoner Arbeit.

Die genannten Vorteile sorgten im 1&1 Testing Unit Team für eine unmittelbare und breite Akzeptanz der Automatisierungs-Technologie. Schnell war klar: Die neuen Roboter-Kollegen sind nicht gekommen, um die Mitarbeiter vor Ort zu ersetzen, sondern um sie tatkräftig zu unterstützen. Das brachte den unsichtbaren Assistenten so viel Sympathie ein, dass das 1&1-Test-Team mit dem 3-D-Drucker ein Roboter-Maskottchen „zum Anfassen" schuf **(siehe Abbildung 1)**, das als Extreme Feedback Device an die Testautomatisierungs-Lösung gekoppelt ist. Der kleine Bot informiert per Sprachausgabe über die Testergebnisse, signalisiert den Status der Testläufe mit grünen und roten Leuchtdioden und hört auf den Namen Emil – der Fleißige.

Menschen und Bots: Arbeitsteilung in der Testing Unit

Mittlerweile hat sich bei der 1&1 eine effiziente Kooperation zwischen menschlichen und digitalen Mitarbeitern eingespielt **(vgl. Abbildung 2)**. Die umfassende Qualitätssicherung des neuen CRM Release Candidate beginnt schon in der Entwicklungsphase mit

Andreas Förch

(andreas.foerch@1und1.de)
ist Head of Quality & Delivery Management bei 1&1. Gemeinsam mit seinem Team verantwortlich für die Qualitätssicherung der Customer Care IT-Systeme.

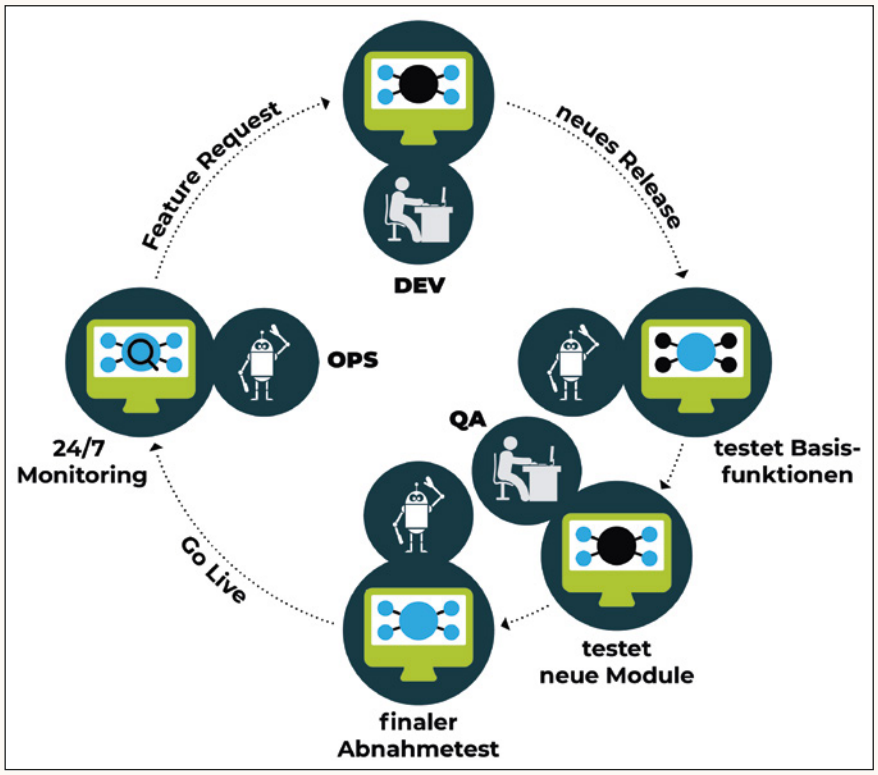


Abb. 2: Softwaretests vor dem Go Live, End-to-End-Monitoring im Betrieb: Die Qualitätssicherung des 1&1 CRM-Systems ist ein Erfolgsmodell für perfekte Mensch-Bot-Kooperation.

einem dauerhaften, automatisierten Check aller Kernfunktionalitäten der neuen Version. Andreas Förch erklärt anhand eines leidlich bekannten „Standard-Szenarios", warum: „Release Date ist übermorgen. Die intensive Qualitätsprüfung eines neuen Moduls steht an. Der vielbeschäftigte Tester hat sich den ganzen Tag extra dafür geblockt, seine Testfälle spezifiziert, macht sich hochmotiviert an die Arbeit. Er startet die Anwendung und es passiert: nichts." Und wieder verrinnt viel wertvolle, knapp bemessene Zeit. Aus dieser Erfahrung heraus prüfen die Software-Roboter bei 1&1 das neue Release vorab auf „alle Service-Abläufe, die einfach gegeben sein müssen, wie die Anwendung starten, einen Kunden laden, einen Vertrag auswählen, Stammdaten ändern."

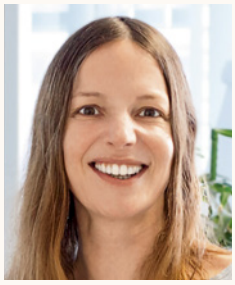
Erst nach diesem Basic-Funktions-Check geben die Software-Roboter grünes Licht für die komplexen manuellen Modul- und Integrationstests der neuen Version. Jetzt können die Mitarbeiter in der QA die neu entwickelten Module intensiv gegen die Spezifikationen aus dem Fachbereich Customer Care testen, bis jedes Projekt die notwendige Reife für eine Aufnahme ins neue Release hat.

Sind die manuellen Tests erfolgreich abgeschlossen, werden die relevanten, neuen Testfälle für die neuen Funktionen in das bestehende Automatisierungs-Portfolio aufgenommen. Am Ende der Testphase übernehmen wieder die Software-Roboter – und prüfen das vollständige, integrierte Release in einer produktionsnahen Umgebung auf Regressionen.

Nach dem Go Live: 24/7 Monitoring des CRM-Systems

Der finale Abnahmetest ist fehlerfrei gelaufen. Alle Lichter stehen auf Grün: Die neue CRM-Version geht im 1&1-Fachbereich Customer Care live. Hier geht die Arbeit für die Software-Roboter weiter: Mit den Workflows aus der Testphase bedienen sie das produktive CRM-System und testen 24/7 Funktionalität, Verfügbarkeit und Performanz der geschäftskritischen Anwendung im Betrieb. Mit der dauerhaften automatisierten Überwachung aus Nutzerperspektive – in der Fachterminologie End to End oder End User Experience Monitoring genannt – stellen die IT-Verantwortlichen bei 1&1 sicher, dass die Mitarbeiter im Customer Care jederzeit auf eine leistungsstarke Anwendung zugreifen können.

„Ohne Automatisierung sind Qualitätstests in diesem Umfang gar nicht denkbar", ist das Fazit des Teamleiters der 1&1 Testing Unit. „Die Software-Roboter können und sollen unsere Mitarbeiter mit ihren spezifisch menschlichen Kompetenzen nicht ersetzen – aber sie können sie von monotonen Aufgaben befreien, die Arbeitslast für das gesamte Team reduzieren, und uns dabei helfen, auch unter Zeitdruck exzellente Software auszuliefern."



Johanna May

jmay@servicetrace.de
www.servicetrace.de
arbeitet als Online Content Manager bei der Servicetrace GmbH, den Spezialisten für Software-Robotics, RPA und Testautomatisierung. „Menschen sollten nicht wie Maschinen arbeiten. Wenn Roboter monotone Massenaufgaben automatisieren, wird Arbeit humaner."

»Testmanagement bei der andsafe AG«

Die andsafe AG ist ein digitales Start-up in der Versicherungsbranche. In nur sieben Monaten wurde der digitale Versicherer auf der grünen Wiese in der Cloud aufgebaut. In diesem technisch wie fachlich agilen Umfeld die Qualität sicherzustellen, ist eine besondere Herausforderung. Diese haben wir als Berater gemeistert, indem wir auf standardisierte Testmethodiken, -frameworks und -werkzeuge zurückgegriffen und diese konkret auf die Bedürfnisse des Kunden angepasst haben. Der Spagat zwischen der kurzfristigen Skalierung, um den Einführungstermin nicht zu gefährden, und der Sicherstellung der Nachhaltigkeit und Effizienz durch konsequente Automatisierung war groß – hier die Erfahrungen bei der Umsetzung.

Die andsafe AG ist ein im Oktober 2018 als 100-prozentige Tochter der Provinzial gegründeter digitaler Versicherer. Bereits im Mai 2019 wurde das erste Versicherungsprodukt mit der Betriebshaftpflichtversicherung gelauncht. Grundlage war die Standardsoftware der adesso insurance solutions, die für die wesentlichen versicherungstechnischen Kernprozesse wie Vertrag, Leistung und Partner zum Einsatz kam. Die Softwarekomponenten für die Endkunden, aber auch für Versicherungsmakler und Vergleicher, wurden Inhouse durch die andsafe AG entwickelt. Hierbei handelt es sich im Wesentlichen um die Abschlusstrecken, Schadenstrecken und das Kundenportal.

Digitaler Vorreiter in der Versicherungsbranche

Das Versicherungs-Start-up fokussiert sich auf die Reduzierung des Verwaltungsaufwands und die Verschlankung der Prozesse, um so mehr Leistung für die eigenen Versicherten erbringen zu können. Von Vorteil ist die gesellschaftsrechtliche Konstellation, denn mit der Provinzial steht ein starker Investor hinter der digitalen Tochter – hier werden profunde Erfahrungen und innovative Technologien effektiv verknüpft:

Umso wichtiger ist es, dieses hohe Niveau nachhaltig zu sichern: Dazu wurde der Fokus auf den Aufbau eines effizienten Software-Qualitätsmanagements gesetzt, das verschiedene Aspekte wie die Toolauswahl, die Einführung einer Testmethodik für agile Projekte, aber auch eine revisionssichere Dokumentation der Testfälle und -ausführungen umfassen sollte. Teil der Lösung war der Einsatz eines Nearshore-Teams, das die Testfälle erstellt und durchführt, sowie der optimale Mix aus Testautomatisierung und manuellen Ausführungen. Eine Anforderung war, dass die vielen unterschiedlichen mobilen Endgeräte bei den manuellen Tests und der Testautomatisierung umfassend berücksichtigt werden.

Die Herausforderungen: Komplexität, Zeitdruck und ein neues Team

Die Zielsetzung war ehrgeizig: Im November 2019 startete der Aufbau der IT, bereits für Mai 2020 war der Go-Live der Anwendungslandschaft in der Amazon Cloud geplant. Neben dem Customizing der versicherungsfachlichen Standardsoftwarelösung wurde ein komplettes Kundenportal mit zahlreichen Services und die Abschlussstrecke für ein innovatives Versicherungsprodukt entwickelt. Direkt zum Start wurden dabei verschiedene Vertriebswege implementiert und neben dem Direktvertrieb auch Versicherungsmakler und Vergleicher angebunden.

Zu Beginn des Projekts lag der Fokus naturgemäß auf dem Aufbau der fachlichen Vision, der Konkretisierung dieser in Form von User-Storys und der Entwicklung von technischen Fähigkeiten, um diese umsetzen zu können. In dieser Phase stand die Qualitätssicherung der Software nicht so stark im Fokus, da sich Ideen auch noch schnell geändert haben und Refactoring zum Tagesgeschäft gehörte.

Als wir im Februar – drei Monate vor Go-Live – als Berater in das Projekt eingestiegen sind, gab es weder einen übergeordneten Test- noch einen einheitlichen Releaseprozess, aber es war schon eine Menge Software entstanden – der Projektstart war demnach eine spannende Herausforderung:

Große Teile der Standardsoftware waren noch nicht fertiggestellt. Der Releasezyklus des Drittanbieters war nicht mit den Sprints bei andsafe synchronisiert. User-Storys, also die fachlichen Beschreibungen aus Sicht des Anwenders, die für agiles Vorgehen essenziell sind, enthielten anfänglich noch wenig stabile Akzeptanzkriterien. Diese bilden jedoch die Basis für die Testfälle und sind notwendig, um feststellen zu können, wann die technische Umsetzung überhaupt korrekt ist.

Eine weitere Herausforderung stellte die parallele Entwicklung mit mehreren Scrum-Teams dar, deren Produkte sich am Ende vollständig integriert für den Kunden wie eine Lösung darstellen sollen. Hier geht es also nicht nur um die Qualität einzelner Bausteine, sondern des großen Ganzen und übergreifender Prozesse. Aber wer ist dafür verantwortlich?

Die Umsetzung: Theorie und Praxis – zwei Seiten einer Medaille

Zur Entwicklung der Software ging andsafe agil auf der Basis des Scrum-Frameworks vor, wobei sämtliche Softwarekomponenten in der Cloud betrieben werden. Im Februar 2019, vier Monate nach dem Entwicklungsstart, sollte ein übergreifend einheitliches Qualitätsmanagement eingeführt werden.

Die Theorie

Die Idee war, das Testteam als eigenes zentrales Team, also als „shared Service" für die Dev-Teams, zu organisieren. Die cross-funktionale Zusammensetzung der Scrum-Teams wurde hier also bewusst durchbrochen. So sollte sichergestellt werden, dass sowohl der Impact für die Dev-Teams, die mitten in der Entwicklung waren, minimal war und gleichzeitig die einzelnen Produkte der Entwicklungsteams vollständig integrativ aus Sicht des Kunden betrachtet wurden.

Die Etablierung der Testprozesse und der Aufbau des Testteams sollten in drei Phasen erfolgen: Auf eine einmonatige Anlaufphase folgten ein Go-Live über drei Monate und eine anschließende Optimierungsphase für einen effizienten, fortlaufenden Regelbetrieb.

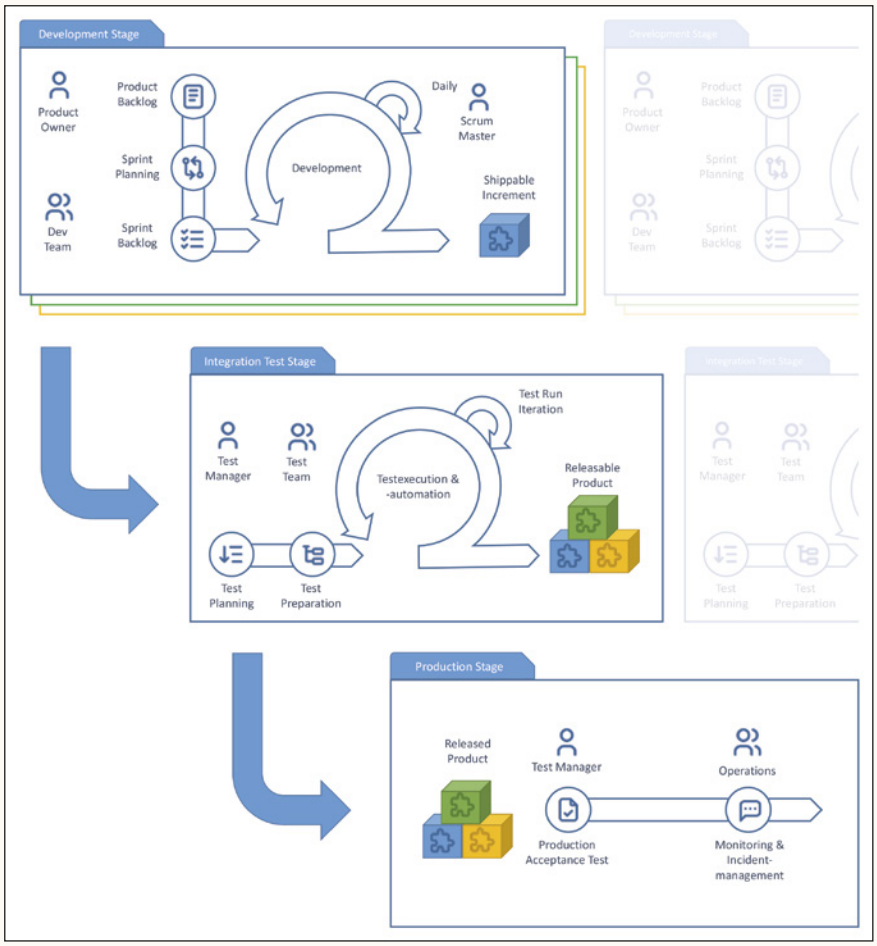


Abb. 1: Agiler Softwareentwicklungs- und Testprozess

Phase 1 – Anlaufphase

In der Anlaufphase wurde der Aufbau des Testteams initiiert, das aus zwei Testmanagern sowie drei Testern vor Ort und zwei deutschsprachigen Testern in Klausenburg (Cluj-Napoca, Rumänien) bestand. Die Tester nahmen an den Daily Meetings der Dev-Teams teil, um so einen bestmöglichen Austausch mit den Softwareentwicklern zu realisieren. Hier zeigte sich, dass durch die Zusammenarbeit des zentralen Testteams mit den Dev-Teams die Entwicklung der Testfälle und der Akzeptanzkriterien der User-Storys immer exakter wurden und damit natürlich auch die Qualität der User-Storys erheblich gesteigert wurde.

Für die agile Entwicklung in den Scrum-Teams wurde bereits Jira von Atlassian eingesetzt, sodass es naheliegend war, das Testmanagement-Tool auf Jira aufzubauen. Aufgrund der vorhandenen Erfahrungen in anderen Projekten wurde kurzfristig das Jira-Plug-in „Xray Test Management for Jira" eingeführt. In dieser Phase erfolgte auch die Definition des Testprozesses.

Zudem wurden Maßnahmen definiert, die die effiziente Zusammenarbeit der einzelnen Teams sicherstellen sollten. So arbeiteten die rumänischen Kollegen in der ersten Zeit vor Ort, um eine konstruktive Zusammenarbeit teamübergreifend zu erleichtern.

Zum Teststart wurde gezielt auch auf explorativ Testen gesetzt. Während drei Tester explorative Tests durchführten, arbeiteten die anderen die vorliegenden User-Storys durch und erarbeiteten eine einheitliche Struktur für die Testfälle. So gelang es, die ersten strukturierten Tests für eine ordentliche Testabdeckung in kurzer Zeit zu erstellen, die wichtigsten Prozesse zwischen Testmanagern und Entwicklern, Product Ownern, Scrum Master und Management abzustimmen und gleichzeitig durch die explorativen Tests erste Fehler aufzudecken und den Testern ein gutes Verständnis der versicherungsfachlichen Systeme zu vermitteln.

Zu guter Letzt stand der Aufbau einer Abnahmetestumgebung an. Hier zeigten sich die Stärken der Cloud und der innovativen Entwicklungswerkzeuge, die es ermöglichten, in kurzer Zeit eine produktionsnahe Umgebung aufzubauen, die eine kontinuierliche Belieferung mit den einzelnen Softwarekomponenten ermöglichte.

Phase 2 – Go-Live

Um eine gute Testabdeckung zu gewährleisten, ging es in der Folge um die strukturierte Erstellung von Testfällen, wobei die einzelnen Anwendungsteile gezielt mit unterschiedlichen Testprioritäten versehen wurden: Themen mit Kundenauswirkung oder im direkten Zusammenhang mit der Zahlungsabwicklung wurden am höchsten priorisiert. Gleichzeitig stellte sich heraus, dass die regelmäßigen explorativen Tests einen wichtigen Beitrag zu einer sinnvollen Priorisierung leisteten, da sie wichtige Hinweise auf fehleranfällige Komponenten gaben.

Die Go-live-Phase war gezeichnet von täglichen Deployments auf der Testumgebung. Hier zeigte sich gleich mehrere Vorteile der autark agierenden zentralen Testteams:

- Das Testteam war durch seine Zusammenarbeit mit den Dev-Teams optimal aufgestellt, um die Koordination der einzelnen Releases und Deployments zu steuern, und übernahm quasi die Rolle des Release Train Engineers, wie sie im agilen Framework SaFe definiert ist.
- Zudem erlaubte die Unabhängigkeit des Testteams, die Zusammenstellung der Dev-Teams von andsafe zu optimieren, ohne Auswirkungen auf das in dieser Phase sehr wichtige Qualitätsmanagement.
- Nicht zuletzt konnte das Testteam unabhängig skalieren, ohne dass sich die Schnittstellen zu den Scrum-Teams änderte. So konnten nach der erfolgreichen Etablierung des Teams und der Prozesse sukzessive der Umfang und die Aufgaben des Testteams erweitert werden.

Phase 3 – Optimierung und Regelbetrieb

Der erste Erfolg war, dass der ambitionierte Zeitplan gehalten wurde und der Go-Live der Gesamtinfrastruktur mit dem Verkaufsstart der andsafe-Betriebshaftpflichtversicherung erfolgte. Nun lag der Fokus auf der kontinuierlichen Weiterentwicklung der Software und des Produktportfolios. Das hieß im Umkehrschluss, dass der manuelle Testumfang sich deutlich vergrößerte.

Ein weiteres Ziel war eine schnellere und regelmäßigere Auslieferung von Softwareanpassungen in Produktion. Dazu wurden Continous-Delivery-Prozesse etabliert – eine besondere Herausforderung für Softwaretests in Hinblick auf die wirtschaftliche und effiziente Qualitätssicherung des Gesamtsystems.

Jetzt war der richtige Zeitpunkt gekommen, die Effizienz des Testteams auch bei steigendem Funktionsumfang und kürzeren Releasezyklen sicherzustellen. Dazu wurden Verfahren zur Testautomatisierung eingeführt. Durch die bestehende Testerfahrung konnten stabile Anwendungsteile identifiziert werden, um mit einem Team von drei Entwicklern die Automatisierung des Regressionstests zu starten. Die automatisierten Testfälle wurden dazu mit Cucumber und Gherkin beschrieben und mit Ruby ausgeführt. Ein Vorteil dieser Lösung war die einfache Integration in das Testmanagement-Werkzeug Xray. Dadurch konnte eine einheitliche Sicht auf manuelle und automatisierte Testfälle sowie deren Ausführung gewährleistet werden. Um die Vielzahl an unterschiedlichen mobilen Endgeräten abzudecken, kam die cloudbasierte Testplattform BrowserStack zum Einsatz. Die Testumgebungen wurden gemäß den Bedürfnissen der agilen Softwareentwicklung aufgebaut und es wurde ein Prozess definiert, der den Integrationstest und die produktive Auslieferung mit jedem Sprint (Zwei-Wochen-Intervalle) definiert **(siehe Abbildung 1)**.

Fazit: Flexibles Vorgehen zeigt den gewünschten Erfolg

Das Go-Live der digitalen Versicherungsplattform war ein großer Erfolg. In nur drei Monaten wurde dazu ein individuelles Testvorgehen eingeführt, auf dessen Basis 600 End-to-End-Testfälle geschrieben und durchgeführt wurden. Dabei wurden über 1.600 Fehler gefunden, dokumentiert und behoben. Wesentlicher Erfolgsfaktor war die Entscheidung, von vornherein mit einem autark agierenden Testteam, das parallel zu den bestehenden agilen Dev-Teams aufgebaut wurde, zu starten und zwei parallelen Handlungssträngen zu folgen:

- Einerseits die Durchführung explorativer Tests, um einen qualitativen Überblick zu erhalten, sowie
- andererseits der Aufbau eines professionellen und standardisierten Testprozesses, der die Grundlage für die Automatisierung bildet, welche die Effizienz des Testteams nachhaltig sicherstellt und der Garant für eine hohe Softwarequalität bei steigender Anzahl von Deployments ist.

Viktor Fast

viktor.fast@bbht.de

hat seinen Schwerpunkt im Bereich Qualitätsmanagement bereits als Werkstudent bei der Thalia Bücher GmbH von 2015 bis 2017 im Bereich der Frontendentwicklung für den Webstore diverser Mandanten aufgebaut. Seit 2017 ist er bei der BBHT als Berater tätig und unterstützt dort seither in verschiedenen Kundenprojekten. Seit 2019 ist er bei der andsafe AG als Testmanager im Einsatz und verantwortet mit seinem Team der BBHT Solutions S.R.L. das Qualitätsmanagement.

Christian Treptau

christian.treptau@bbht.de

hat an der Universität Duisburg-Essen Wirtschaftsinformatik studiert. Seit 2015 ist er Geschäftsführer der BBHT Beratungsgesellschaft. Als erfahrener Projektleiter und Softwarearchitekt hat er in vielen, unterschiedlichen Projekten mitgewirkt. Sein Fokus liegt dabei auf Projektmanagement und Prozessoptimierung in komplexen Umfeldern sowie in der Banken- und Versicherungsbranche.

»5-Sterne-Apps werden manuell getestet«

Testautomatisierung, Künstliche Intelligenz, RPA – Qualitätsmanagement bei Mobilen Apps – gehört manuelles Testing von Apps auf Smartphones und Tablets nicht längst der Vergangenheit an? Die erfolgreichsten Apps werden doch bestimmt nicht mehr manuell getestet? Doch – und genau deshalb sind sie so erfolgreich und erzielen die höchsten Bewertungen in den App Stores, räumen Preise und Auszeichnungen ab. Warum ist manuelles App-Testing einer der entscheidenden Erfolgsfaktoren für 5-Sterne-Apps?

Zwei Gesundheits-Apps erhielten beste Bewertungen:

- Rund 16 Millionen Downloads, #1 in der Kategorie Gesundheit und Fitness im Apple App Store, rund 48.000 Bewertungen, 4,5 Sterne (Stand: September 2020) [CWA] – die *Corona-Warn-App* für die Bundesrepublik Deutschland herausgegeben vom Robert-Koch-Institut.
- Rund 2 Millionen Downloads, #19 in der Kategorie Gesundheit und Fitness im Apple App Store, rund 150.000 Bewertungen, 4,8 Sterne (Stand September 2020) [TK] und von Focus Money als beste von Nutzern empfohlene Krankenkassen-App ausgezeichnet [Foc] – die *TK-App*.

Diese beiden Apps werden in diesem Beitrag als anschauliche Beispiele herangezogen, um die Relevanz von Qualitätssicherung, insbesondere von manuellem Testing im Rahmen von App-Projekten zu veranschaulichen. Beide Apps haben trotz unterschiedlicher Zielsetzungen entscheidende Gemeinsamkeiten: Millionen Smartphone-Nutzer haben beide Apps heruntergeladen, verwenden sie regelmäßig und haben eine signifikante Anzahl an App-Store-Bewertungen abgegeben.

Dass die Corona-Warn-App im Vergleich zur TK-App trotz achtmal so vieler Downloads bisher nur 45.000 Bewertungen erhalten hat (TK-App über 143.000 Bewertungen), dürfte vor allem auf den Veröffentlichungszeitraum im App Store zurückzuführen sein und wird in dieser Betrachtung vernachlässigt.

Die entscheidenden Aspekte, die näher betrachtet werden, sind die App Store-Bewertungen der User, insbesondere die qualitativen Rückmeldungen hinsichtlich wahrgenommener Probleme und Verbesserungsvorschläge sowie die daraus resultierenden Sternebewertungen.

Testing oder nicht Testing

Kurz nach der Veröffentlichung der Corona-Warn-App ist bei vielen sehr positiven Bewertungen oftmals vor allem die Hoffnung [FR] auf das neue Instrument im Kampf gegen die Corona-Pandemie als Motivation erkennbar. Denn trotz kritischer Anmerkungen und Verbesserungsvorschlägen in der qualitativen Bewertung haben viele Nutzer fünf Sterne vergeben. Mittlerweile häufen sich jedoch unter den neuesten Bewertungen viele kritische Beiträge. Verschiedenste Probleme, unerwartete Fehlermeldungen und Funktionsbugs offenbaren zahlreiche Qualitätsmängel, über die die Nutzer nicht mehr hinwegsehen. Zwischenzeitlich waren sogar Kernfunktionen betroffen und von den Entwicklern empfohlene Workarounds, wie das tägliche Öffnen der App, waren wenig nutzerfreundlich. [SN-b]

Während die beteiligten Unternehmen versuchen, den Imageschaden, den das Vorzeige-Digitalisierungs-Projekt der deutschen Bundesregierung in der Bekämpfung der Corona-Pandemie genommen hat, den Betriebssystemherstellern anzulasten und durch zahlreiche Updates die Nutzerzufriedenheit wiederherzustellen [HB], stellt sich die Frage, ob optimiertes Testing und Qualitätsmanagement diese Entwicklung nicht verhindert hätte – ausreichend Budget war laut den of-

fiziellen Informationen vorhanden. Allerdings ist nur ein verschwindend geringer Teil für den Bereich Testing ausgewiesen worden, hierbei im Speziellen für Penetrations- und Sicherheitstests. [SN-a] Inwiefern die Corona-Warn-App funktional, auf Usability, auf Performanz, auf Kompatibilität, explorativ usw. ausführlich und manuell getestet wurde, haben die Entwickler bisher nicht transparent gemacht.

Dabei ist es gerade bei einer derartig breiten Zielgruppe in persona der Gesamtbevölkerung (offizielles Nutzungsalter ab 17 Jahren [App_CWA]) umso relevanter, eine App ausführlich und in allen relevanten Aspekten getestet zu releasen, da die App auf möglichst allen am Markt gängigen Geräte-Modellen und Betriebssystemversionen funktionieren sollte.

Die Techniker Krankenkasse (TK) mit ihrer Kunden-App, der TK-App, legt auf eine optimale Marktabdeckung besonderen Wert. Die TK ist die größte deutsche Krankenkasse [Wiki] und lässt ihre TK-App manuell durch externe Experten testen. [TKV]

Warum setzen Unternehmen wie die TK auf manuelles App-Testing und wieso könnte das den entscheidenden Unterschied ausmachen?

Faktoren einer positiven UX

Die Erfahrungen seit dem Start der ersten Mobile Apps für Smartphones haben es deutlich gezeigt: Nur durch positive Benutzererfahrung, positive User Experience, ist der langfristige Erfolg einer App gesichert. ([Hec11], S. 72) Was sind also die entscheidenden Faktoren für eine positive Benutzererfahrung?

Roidl/Rüsing sprechen vom sogenannten Companion in Life als perfektem Zielbild einer App. In ihrer Studie „Erfolgsfaktoren zur Verhaltensveränderung durch mobile Apps" [Roi14] kommen sie zu dem Schluss, dass

- Kontextsensitivität und
- Erwartungskonformität einer App

die entscheidenden Erfolgsfaktoren sind. Das gewünschte Verhalten belohnen die Nutzer am höchsten. Erwartungskonformität und Kontextsensibilität bilden also den Schlüssel für den Erfolg einer App.

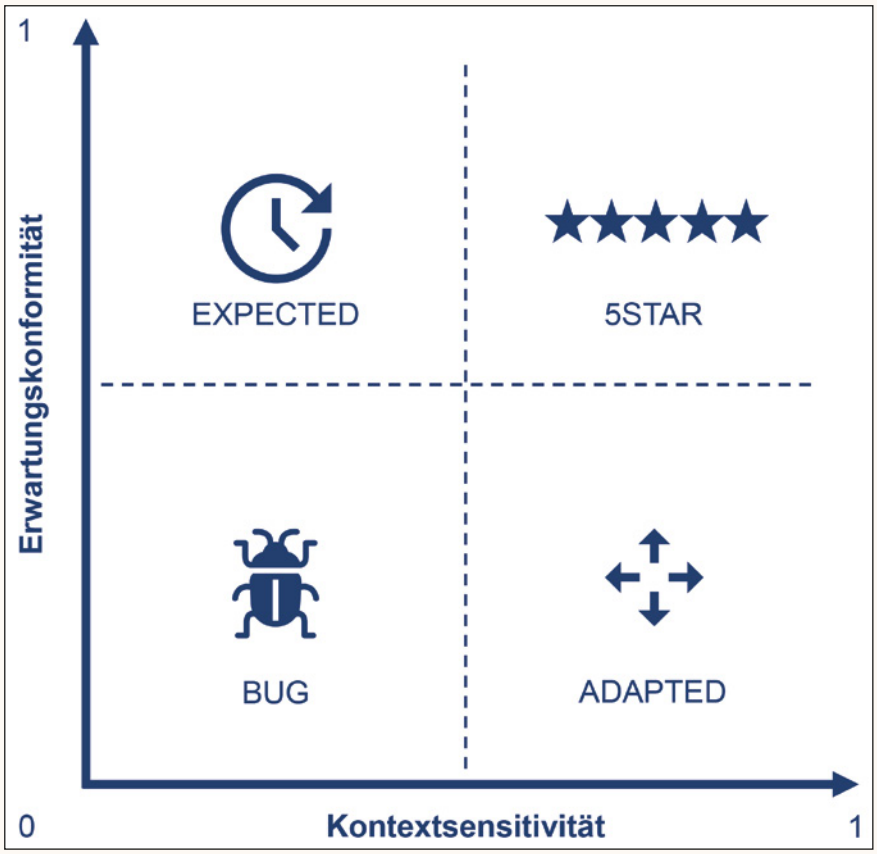


Abb. 1: KE-Matrix zur Analyse der Güte und Erfolgswahrscheinlichkeit von Apps, © S. Darimont, 2020

Erwartungskonformität ist in der Norm DIN-ISO 9241-220:2019 definiert: "Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z. B. seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie den allgemein anerkannten Konventionen." [ISO]

Kontextsensitivität beschreibt das Verhalten von Apps, Informationen über den Kontext ihrer Nutzung zu benutzen, um ihr Verhalten darauf anzupassen. [Ani99]

Die Güte von Apps lässt sich anhand der beiden Kriterien Erwartungskonformität (E) und Kontextsensitivität (K) in der in **Abbildung 1** gezeigten Matrix abbilden.

Das Ziel der Entwickler oder Herausgeber einer App ist im Regelfall, ein 5STAR oder nach [Roi14] ein Companion in Life (Alltagsbegleiter) zu werden. Apps, die sehr gute Werte bezüglich Erwartungskonformität, aber wenig Kontextsensitivität aufweisen, sind oftmals zuverlässige Helfer, nützliche Tools (bspw. Taschenrechner-App), aber keine Lieblings-Apps. Apps, die eine hohe Kontextsensitivität aufweisen, aber nicht den Erwartungsgewohnheiten entsprechen, zeichnen sich oftmals durch innovative Funktionen aus. Um ein 5STAR zu werden, muss die App aber noch zusätzlich auf die Erwartungen der Nutzer hin optimiert werden. Apps, die in beiden Kategorien schlechte Werte erzielen, sollten grundlegend getestet und überarbeitet werden, um nicht länger als BUGgy-App zu gelten.

Manuelles App-Testing

Um die bestmögliche User Experience (UX) zu erreichen, hat der Entwickler einer App sicherzustellen, dass Erwartungskonformität und Kontextsensitivität schon mit dem ersten Release möglichst umfassend erfüllt und die Nutzer somit zufriedengestellt werden.

Dies macht professionelles Testing und Qualitätssicherung erforderlich. Die Tester prüfen die App systematisch hinsichtlich Funktionalität, Performanz, Usability, Kompatibilität sowie explorativ mit Blick auf die Devices, für die sie entwickelt wird, in Abhängigkeit von Gerätemodell, Betriebssystem und Betriebssystemversion sowie

Referenzen

- [Ani99] K. D. Anind, D. A. Gregory, Towards a Better Understanding of Context and Context-Awareness, Graphics, Visualization and Usability Center and College of Computing, Georgia Institute of Technology, 8.7.1999
- [App_CWA] https://apps.apple.com/de/app/corona-warn-app/id1512595757
- [Cap] Capgemini, Back to the Future – Tomorrow's Quality Engineering Today, siehe: https://www.capgemini.com/podcasts/back-to-the-future-tomorrows-quality-engineering-today-podcasts/
- [CWA] Corona-Warn-App, siehe: https://www.spiegel.de/netzwelt/apps/corona-warn-app-bekommt-update-jetzt-auch-auf-tuerkisch-verfuegbar-a-59090403-ebb7-4e8d-956b-8cedb8086ee4 und [App_CWA]
- [Foc] Focus Money, Ausgabe 14/2020, Corona und mein Geld, siehe: https://www.focus-shop.de/focus-magazin-14-2020.html
- [FR] Frankfurter Rundschau, 19.6.2020, siehe: https://www.fr.de/politik/corona-apps-hoffnung-normalitaet-digital-13771596.html
- [HB] Handelsblatt, 28.7.2020, siehe: https://www.handelsblatt.com/technik/it-internet/kontaktverfolgung-sap-und-telekom-rudern-zurueck-kritik-an-apple-fuer-fehler-in-corona-app-revidiert/26045554.html
- [Hec11] M. Heckner et al., Engineering Mobile User Experience: Think. Design. Iterate. Publish, in: M. Eibl (Hrsg.), Mensch & Computer 2011: überMEDIEN|ÜBERmorgen, Oldenbourg, siehe: https://dl.gi.de/bitstream/handle/20.500.12116/7956/Heckner_Schneidermeier_Bazo_etal_2011.pdf?sequence=2
- [ISO] Ergonomie der Mensch-System-Interaktion – Teil 220, EN ISO 9241-220:2019
- [Keu19] A. Keus, Barrierefreie App-Entwicklung – Native, Plattformübergreifende und Web-Apps im Vergleich, RWTH, Aachen, 2019, siehe: https://publications.rwth-aachen.de/record/758635/files/758635.pdf
- [Roi14] E. Roidl, O. Rüsing, Erfolgsfaktoren zur Verhaltensveränderung durch mobile Apps, UP14 – Kurzvorträge, German UPA, 2014
- [SN-a] Spiegel Netzwelt, 17.6.2020, siehe: https://www.spiegel.de/netzwelt/apps/corona-warn-app-wie-erklaeren-sich-die-gesamtkosten-von-68-millionen-euro-a-56b5abe1-e0a6-4b1c-9177-9066df3d9b14
- [SN-b] Spiegel Netzwelt, 29.7.2020, siehe: https://www.spiegel.de/netzwelt/apps/corona-warn-app-telekom-und-sap-empfehlen-app-einmal-am-tag-zu-oeffnen-a-6518f49e-0413-48da-97ee-6ec899c9d542
- [TK] TK-App, siehe: https://www.tk.de/techniker/magazin/digitale-gesundheit/apps/tk-app-2023650 und https://apps.apple.com/de/app/tk-app/id1163694230
- [TKV] Regelmäßige Ausschreibung der Testing-Dienstleistung auf https://vergabe.tk.de/Satellite/company/welcome.do
- [Wiki] https://de.wikipedia.org/wiki/Techniker_Krankenkasse

Auflösung und Zielgruppe(n) der Nutzer. Tool-gestütztes Testing kann auf mögliche Fehlerquellen aufmerksam machen und automatische Codereviews gehören ohnehin zum Standard professioneller Entwicklung. Manuelles App-Testing durch Menschen ist jedoch notwendig zur Verifikation und Analyse etwaiger Fehler. [Keu19]

Können Künstliche Intelligenz, Testautomatisierung oder Robotic Process Automation (RPA) bereits jetzt oder in naher Zukunft manuelles Testen substituieren?

Nick Utley, Transformation Leader in Analytics bei Capgemini, und Eran Bachar, Executive Product Manager, Functional Testing bei Micro Focus, kommen zu einem eindeutigen Urteil: Der Anteil automatisierter Tests ist immer noch relativ niedrig, verursacht sehr hohe Kosten für Maintenance und Anpassung der Testfälle, insbesondere bei integrierten Systemen, und erfordert sehr spezifisches Know-how. Die Zeit, die Testautomatisierungs-Implementierung anzupassen, ist oftmals höher, als die Tests direkt manuell durch qualifizierte Tester durchzuführen. Machine Learning (ML) und Künstliche Intelligenz (KI) können Testautomatisierung erweitern und menschlichen, manuellen Testern assistieren, aber diese nicht in näherer Zukunft ersetzen oder ablösen. [Cap]

Erwartungskonformität und Kontextsensitivität, die entscheidenden Faktoren um nicht nur eine gute, sondern eine exzellente App zu realisieren, umfassen dabei so viele weiche Faktoren, Erfahrungen, kulturelle wie soziale Aspekte, die zumindest mit den aktuell am Markt verfügbaren Technologien wirtschaftlich nicht substituierbar sind.

Die erfolgreichsten Apps werden heute und zukünftig manuell durch professionelle App-Testing-Experten getestet. App-Testing-Expertise, jahrelange Erfahrung und Know-how katapultieren Apps in die Liga der 5STAR-Apps und lassen sie zum Companion in Life ihrer Nutzer, ihrer Kunden werden.

Sebastian Darimont

sebastian.darimont@almato.com

ist Leiter Business Development bei der Almato AG. Er unterstützt Unternehmen und öffentliche Institutionen bei der intelligenten Digitalisierung ihrer Prozesse und Geschäftsmodelle. Schwerpunkt vieler seiner Projekte bildeten in den letzten Jahren Mobile Apps, die er unter anderem mit einem Team aus professionellen App-Testern fortlaufend für die beste User Experience der Zielgruppen und den Erfolg seiner Kunden optimiert.

»Schlüssel für die erfolgreiche Integration«

Bei Anwendungen, die heutzutage entwickelt werden, steigt die Anzahl an Anforderungen und Komplexität an. Vor allem durch die wachsende Anzahl an Schnittstellenpartnern wird die Kommunikation beziehungsweise Abstimmung enorm schwierig. Für den Erfolg größerer Projekte wird es immer essenzieller, dass ein gutes Zusammenspiel zwischen den verschiedenen Teams/Abteilungen herrscht. Das ist die Basis, um als Ergebnis eine Anwendung an den Markt bringen zu können, die sowohl eine entsprechende Qualität besitzt, als auch in einem angemessenen Zeitraum entwickelt wurde. Ein weiterer entscheidender Hebel, um einen stabilen Produktionsbetrieb zu erzielen, liegt vor allem im Testdatenmanagement.

Durch Agile- und DevOps-Methoden werden die Releasezyklen immer kürzer und in diesem Kontext ist ein aktives Testdatenmanagement zwingend erforderlich. Das Gesamtvorhaben funktioniert nur durch „Continuous Testing". Das heißt, dass der Test frühzeitig eingebunden wird und die Testdaten bei Bedarf „auf Knopfdruck" zur Verfügung stehen. Dafür ist zum Beispiel eine hohe Automatisierungsrate sehr wichtig. Grundsätzlich geht es aber um den Shift-left-Ansatz, das heißt, um die Fragestellung, wie die Test(daten)prozesse beschleunigt beziehungsweise verschlankt werden können, um schneller auf Marktveränderungen zu reagieren.

Erschwerend kommt die Datenschutzgrundverordnung EU-DSGVO hinzu, die am 25. Mai 2018 in Kraft getreten ist. Seitdem ist ein noch sensiblerer Umgang mit personenbezogenen Daten notwendig. Die anfängliche Schonfrist scheint vorbei zu sein, denn mittlerweile wurden Sanktionen im zweistelligen Millionenbereich ausgesprochen. Nun sollte auch der allerletzte ITler die neuen gesetzlichen Regelungen beachten. Dies gilt natürlich auch für die Entwicklungsteams, denn Echtdaten haben im Test nichts verloren, da die Verwendung nicht zweckgemäß wäre. Diese Problematik lässt sich im Rahmen von Testdatenmanagement lösen. Die Datenschutzbrisanz löst sich nämlich bei der Nutzung von synthetischen Testdaten komplett auf. Eine Alternative ist die Verfremdung von Produktionsdaten, wobei sichergestellt werden muss, dass der Verfremdungsalgorithmus nicht zurückverfolgt werden kann.

Im Folgenden wird die Testdaten-Thematik beleuchtet und einige Best Practices aus der Praxis erläutert.

Die Signifikanz von synthetischen Testdaten

Von synthetischen Testdaten ist die Rede, wenn in der Testumgebung künstliche Testdaten erzeugt werden, um das Verhalten von Echtdaten in der Produktion zu simulieren. Je realitätsnaher der Test ist, desto aussagekräftiger sind auch die Testergebnisse. Die synthetischen Testdaten werden so geformt, dass möglichst alle Korrelationen der realen Daten abgebildet werden.

Die Erstellung der Testdaten kann entweder manuell über die Anwendung erfolgen oder automatisiert über ein geeignetes Testdaten-Tool. Die initialen Aufwände, um die Skripte für die Automatisierung zu erstellen, lohnen sich insbesondere ab einer gewissen Anzahl an Test-Wiederholungen. Synthetische Testdaten sind besonders wertvoll, um zum Beispiel ein strukturiertes Regressionstestset

aufzubauen oder aber um spezielle Konstellationen zu testen, die produktiv noch nicht erhoben werden können. Bei der Weiterentwicklung einer Anwendung können somit unter anderem neue Funktionalitäten durch synthetische Testdaten getestet werden. Dies inkludiert auch die Einbeziehung von negativen Testfällen.

Theoretisch ist es möglich, jegliche Testdatenkonstellation synthetisch zu erstellen. Für die Erreichung einer hohen Testabdeckung ist das hervorragend. Es ist sehr wichtig, sorgfältig zu testen, um vor der Freigabeerteilung beziehungsweise -empfehlung Transparenz zu haben bezüglich der Funktionsfähigkeit der Anwendung. Zum Beispiel müssen die Sachbearbeiter vor dem Go-Live geschult werden, wie sie mit der neuen Version der Anwendung zu arbeiten haben. Sie müssen wissen, welche Funktionalitäten korrekt umgesetzt wurden, welche Einschränkungen (ggf. Workarounds) es gibt und welche Funktionalitäten gar nicht erst lauffähig sind, sodass beispielsweise gewisse Vorgänge händisch erfasst werden müssen. Um diese Informationen übergeben zu können, wird eine hohe Testabdeckung benötigt sowie eine ausgezeichnete Testdaten-Qualität.

Frühzeitige Schnittstellentests durch Service-Virtualisierung

Mit steigender Komplexität in der Anwendung, zum Beispiel durch erhöhte Anzahl an Schnittstellen, wird das Testen entsprechend aufwendiger oder schwieriger. Dadurch verkompliziert sich auch die Testdatenbereitstellung. Angenommen, ein Schnittstellenpartner ist im Test blockiert oder noch nicht bereit für den Test, so kann der Test durch einen Mock trotzdem ermöglicht werden. Hierbei wird das Schnittstellen-Verhalten für die Partner-Anwendung über eine Test-Attrappe simuliert. Dafür müssen die zugehörigen Eingabe- und Ausgabewerte bekannt sein. Falls der Ausgabewert unabhängig ist vom Eingabewert und beliebig sein darf, so kann ein fixer Dummywert definiert werden. Diese Technik wird als Stub bezeichnet und ist eine vereinfachte Form.

Datenbankmanipulation zur Erfüllung von Vorbedingungen

Für bestimmte Testfälle gibt es Vorbedingungen, die erfüllt sein müssen. Diese notwendigen Testdatenkonstellationen lassen sich in manchen Fällen synthetisch, über eine Datenbankmanipulation, schneller erzeugen als direkt über die Anwendung. Wenn zum Beispiel Testdaten benötigt werden, die in einem Vorgang einen gewissen Status erreicht haben, so verläuft der Prozess wesentlich effizienter, wenn der gewünschte Status direkt in der Datenbank eingetragen wird.

Für Tester mit Datenbank-Skills sollten diese kleineren Anpassungen in der Datenbank ein leichtes Spiel sein. Wichtig ist es, die Szenarien zu identifizieren, bei denen dieser Ansatz sinnvoll ist.

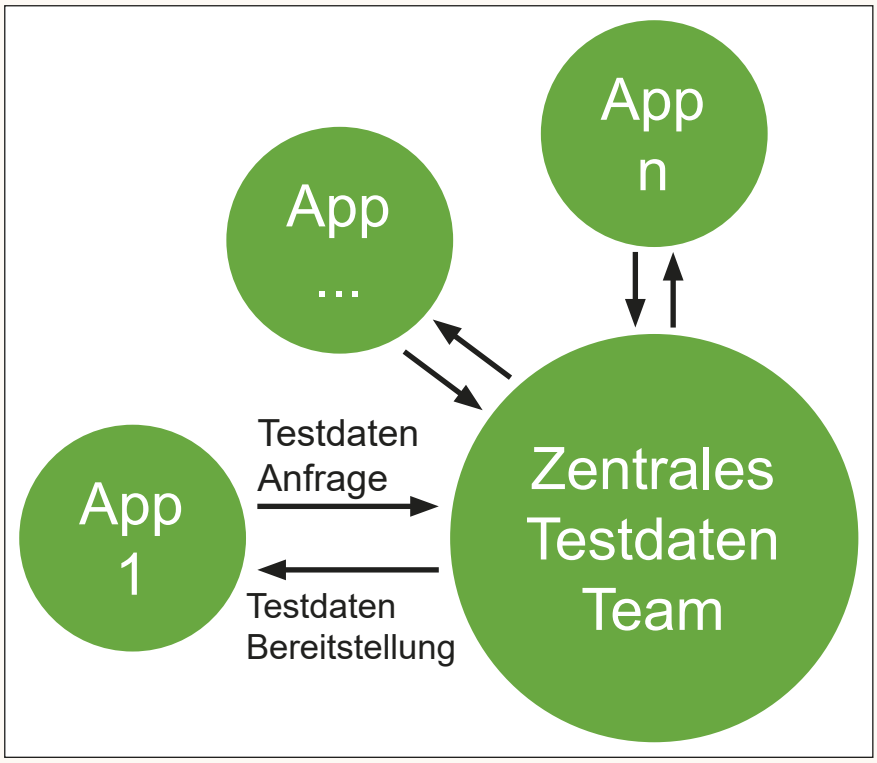


Abb. 1: Zentrales Testdatenmanagement-Team

Automatisierte Generierung von Testdaten

Einer der größten Schmerzpunkte im Testdatenprozess sind die hohen manuellen Aufwände. Deshalb spielt die Automatisierung in diesem Zusammenhang eine sehr wichtige Rolle. Sie hilft, die Geschwindigkeit und Qualität im Test zu steigern, sofern eine sinnvolle Strategie dafür entwickelt wurde. Die Automatisierung ist ein Hebel um, die „Time-to-Market" zu verringern, weil die Tests nach Entwicklungsende ohne Zeitverluste durchgeführt werden können. Durch den Aufbau eines entsprechenden Frameworks ist es sogar möglich, die automatisierten Tests über Nacht laufen zu lassen.

Auf dem Markt herrscht großer Konkurrenzkampf zwischen den Toolherstellern. Die Entscheidung, welches Tool zur Automatisierung implementiert werden sollte, hat nach genauer Prüfung der projekt-spezifischen Faktoren zu erfolgen. Hierzu sollte unbedingt die gesamte Testdatenstrategie herangezogen werden, um gegebenenfalls weitere Synergieeffekte durch die Tool-Funktionalitäten zu erzielen.

Gold Copy als Testdaten Baseline

Bei diesem Ansatz wird einmalig (bzw. in vorbestimmten Intervallen) eine repräsentative Untermenge aus der Produktion extrahiert (Data Subsetting), verfremdet und in die Testumgebung eingespielt. Mit der Erstellung von synthetischen Testdaten sollten fehlende Konstellationen ergänzt werden. Diese fixierten Testdaten bilden die Baseline und können bei Bedarf zur Wiederverwendung zurückgesetzt werden. Alle Entwicklungsteams können somit auf denselben Stand der Testdaten zugreifen.

Die Gold Copy kann durch die Entwicklungsteams bestimmt werden, die alle Konstellationen, die für den Test benötigt werden, zentral anfordern. Andererseits können Machine Learning-Algorithmen eingesetzt wer-

den (Data Profiling), um die Datenstrukturen in der Produktion zu analysieren und entsprechend die repräsentative Untermenge an Daten zu bestimmen.

Mithilfe von künstlichen neuronalen Netzwerken ist es heutzutage auch möglich, aus einem kleinen Datensatz aus der Produktion synthetische Testdaten abzuleiten, in denen alle Korrelationen der realen Daten berücksichtigt sind.

Am Rande sei hier auch erwähnt, dass durch Machine Learning-Algorithmen Erkenntnisse darüber gewonnen werden können, welche Anwendungs- beziehungsweise Geschäftsprozesse im Echtbetrieb besonders häufig genutzt werden. Diese Abläufe können im Test intensiver geprüft werden.

Herausforderungen bei synthetischen Testdaten

Nicht zu unterschätzen sind die Komplexität und der zeitliche Aufwand, um synthetische Testdaten in entsprechender Qualität zu generieren. Die Integrität beziehungsweise Vollständigkeit kann nur erreicht werden, wenn exzellentes Wissen über die Anwendung im Team vorhanden ist.

Eine weitere Herausforderung ist es, Testdaten so zu generieren, dass sie nicht nur über Schnittstellen hinweg konsistent sind, sondern auch über mehrere Teststufen hinweg konsistente Testergebnisse liefern. Wenn beispielsweise Testdaten im Komponententest erfolgreich eingesetzt wurden, sollten diese Daten im Systemtest ebenfalls erfolgreich eingesetzt werden können. Sonst entstehen unnötige zeitliche Verzüge in der nächsten Teststufe. Insbesondere für den Fall, dass unterschiedliche Teams verantwortlich sind für die jeweiligen Teststufen, sollte der Einsatz konsistenter Testdaten vom Team der nächsten Teststufe durchaus aktiv gefordert werden.

Testdaten von schlechter Qualität sind nutzlos und können sogar unnötige hohe Aufwände erzeugen, wie bei sogenannten „false positives". Hierbei handelt es sich um Fehlalarme, die auf mangelhafte Testdaten-Qualität zurückzuführen sind. Somit werden vermeidbare Aufwände durch unnötige Testdurchführung, Defect-Erstellung und Analyse des vermeintlichen Fehlers erzeugt.

Kontext zum Testumgebungsmanagement

Die Basis, um überhaupt testen zu können, ist eine stabile Testumgebung. Downtimes in der Testumgebung können aus unterschiedlichen Gründen auftreten. Wichtig dabei ist, dass diese sofort von den Testern an das Testumgebungsmanagement kommuniziert werden. Sobald eine Testumgebung (wieder) verfügbar ist, sollten die Tester umgehend informiert werden. Eine effektive Kommunikation bezüglich der Testumgebungen ist also auch elementar für den Shift-left-Ansatz.

Aus Kostengründen ist es auch bedeutend, dass die Kapazitäten der Testumgebung effizient genutzt werden. Die Testumgebung sollte deshalb nicht mit redundanten Testdaten überflutet werden und nicht mehr benötigte Testdaten sollten in regelmäßigen Abständen bereinigt werden.

Dediziertes Testdaten-Team

Hilfreich sind dedizierte und serviceorientierte Testdatenexperten im Projekt-Team, die sich nicht nur mit der Anwendung bestens auskennen, sondern auch technisch fit sind, das heißt, Wissen mitbringen für die notwendigen Tools, Skripte und Datenbanken. Wie die Testdaten-Experten in das Projekt-Team eingebettet werden, hängt ganz davon ab, in welchem Modell operiert werden soll. Beispielsweise könnte in einem besonders großen Projekt, als Startpunkt, ein zentrales Testdaten-Team aufgesetzt werden, das im Laufe der Zeit anfängt, hybrider zu agieren. Eine andere Debatte könnte zum Beispiel darüber geführt werden, ob das Projekt geeignet ist, um die Zusammenarbeit mit einem Delivery-Team aus Offshore oder Nearshore anzugehen.

Das Testdaten-Team trägt, unabhängig vom Modell, die Verantwortung dafür, dass die oben genannten Potenziale entfaltet werden. Darüber hinaus sind alle Testdaten-Prozesse angefangen von der Identifizierung der Testdaten, über Akquisition, Modifizierung, Bereitstellung bis hin zur Wartung der Testdaten in den Händen dieses Teams. Die Definierung von performanzbasierten Metriken ist essenziell, um den Erfolg messbar zu machen.

Fazit

Dieser Artikel beschreibt die aktuellen Trends im Testdatenmanagement. Der Leser sollte mitnehmen, dass die Komplexität in den Anwendungen steigt und dass die Rahmenbedingungen für die Softwareentwicklung sich verändern. Aufgrund von DevOps und Agile Praktiken gibt es kürzere Releasezyklen und auch die Datenschutzregelung bringt weitere Herausforderungen. Die Qualität der Software sollte nicht darunter leiden. Im Gegenteil, das ultimative Ziel ist es, seinen Kunden möglichst eine fehlerfreie Anwendung bereitzustellen. Deshalb sollten Qualitätsmaßnahmen in jede Phase der Softwareentwicklung integriert werden. Der Schlüssel für die erfolgreiche Integration ist die Einbettung eines professionellen Testdatenmanagements. Die Potenziale, zum Beispiel durch den Einsatz von synthetischen Testdaten, wurden anhand von Best Practices erläutert.

Beyhan Kizilyokus

beyhan.kizilyokus@accenture.com

hat an der TU Berlin Wirtschaftsmathematik mit Spezialisierung auf Finanzmathematik, Finanzierung & Investition und Software-Engineering studiert. Seit 2015 arbeitet er bei Accenture mit dem Schwerpunkt Testmanagement.

»Faktor Mensch!«

Mehr als die Hälfte aller Softwareprojekte sind nicht erfolgreich [PRO]. Was sind die Gründe dafür? Was zeichnet diese gescheiterten – meist großen – Projekte aus und wie kann man deren Komplexität begegnen? Wie bekommt man Abhängigkeiten in den Griff? Reicht es aus, wenn entsprechende Werkzeuge eingesetzt werden? Oder muss mehr getan werden? Der Artikel zeigt auf, dass der Mensch stets der entscheidende Faktor bei der Komplexitätsbewältigung bleibt und wie wichtig die Organisationskultur und deren Werte in diesem Zusammenhang sind.

Die Größe und Komplexität von Projekten definiert sich nicht nur durch die Anzahl der Teammitglieder oder der eingesetzten Technologien, sondern wird mehrheitlich durch die beteiligten Organisations- und Kommunikationspunkte bestimmt. Eine wissenschaftliche Begründung dafür liefert das Gesetz von Conway, wonach Strukturen von Systemen durch die Kommunikationsstrukturen der umsetzenden Organisationen geprägt sind [CON].

Die Komplexität wird daher ganz wesentlich durch die damit auftretenden Abhängigkeiten bestimmt. Ein entscheidender Erfolgsfaktor für eine erfolgreiche Projektabwicklung ist daher das Management der Abhängigkeiten, die unweigerlich auftreten, wenn mehrere Personen oder Teams zusammenarbeiten.

Zentral versus dezentral

Zentrale Ansätze, bei denen Planung an einer Stelle stattfindet und Entscheidungen von wenigen getroffen werden, sind häufig wenig hilfreich: Viele Dinge werden schlicht übersehen und das schlägt schnell durch ungeplante Mehraufwände und somit zusätzliche Kosten zu Buche. Der aus Kundensicht entscheidende Nachteil ist aber die mangelnde Qualität des gelieferten Produkts, da die Fehlerhäufigkeit steigt.

Wir bevorzugen daher in unseren Projekten den dezentralen Weg. Jeder soll sich in seinem Wirkungsbereich mit seinem speziellen Fachwissen und seinen Erfahrungen einbringen. Das funktioniert aber nur, wenn jeder als Teamplayer agiert, (Selbst-)Verantwortung übernimmt und sich auch als aktiver „Kommunikations-Knotenpunkt" im Projekt versteht. Dieses Bewusstsein ist keine Selbstverständlichkeit. Daher sehen wir die wesentliche Aufgabe unserer Führungskräfte in der strategischen Arbeit, nämlich permanent die dafür nötigen Unternehmenswerte zu fördern und die Mitarbeiter zu diesem Verhalten zu befähigen.

Die Projekte und Herausforderungen

Obwohl unsere Projekte fachlich unterschiedlichen Branchen zugeordnet sind, weisen sie doch Gemeinsamkeiten auf: bestehend aus mehreren Teams, verteilt über zahlreiche Standorte in verschiedenen Ländern, unterschiedlich eingesetzte Technologien, eine Vielzahl externer Zulieferer und ein internationales Projektumfeld. Das stellt

die Projektorganisation täglich vor neue Herausforderungen im Bereich Information und Kommunikation.

Eines der Projekte ist inzwischen auf acht interne Teams mit mehr als 70 Mitarbeitern und nochmals rund der doppelten Anzahl an externen Mitarbeitern im Projektumfeld angewachsen.

Während die Hinzunahme eines zweiten Teams meist noch sehr reibungslos erfolgt, funktionieren ab dem dritten Team die vormals erfolgreichen Methoden plötzlich nicht mehr so richtig und die Effizienz im Gesamtprojekt beginnt zu sinken. Beispielsweise kann es passieren, dass Deployments eines Teams immer häufiger Inkonsistenzen und Fehler in einem anderen Team auslösen – klassische, nicht ausreichend bedachte Seiteneffekte. Kostspielige Fehlersuche und Rebuilds sind die Folge. Schnell kann dadurch auch die Stimmung und Motivation im Gesamtprojekt darunter leiden. Das bewirkt wiederum eine stetig abnehmende Qualität, was natürlich auch dem Kunden nicht lange verborgen bleibt.

Um dem entgegenzuwirken, setzen wir verstärkt auf dezentrale Ansätze und appellieren an die Eigenverantwortung der Mitarbeiter. Das hört sich wie eine Selbstverständlichkeit an, oder wie eine „Ausrede" des Managements, welches damit die Verantwortung auf die Leistungsfähigkeit der operativen Arbeitsebene delegiert. Tatsächlich müssen dafür aber die notwendigen organisatorischen Rahmenbedingungen geschaffen werden, die offene Kommunikation und Fehlerkultur erlauben und sogar unterstützen. Diese Kommunikation stellt nicht nur die Basis erfolgreicher Projektarbeit dar, sondern ermöglicht allen Projektbeteiligten eine konstruktive und wertschätzende Zusammenarbeit.

Der monetäre Erfolg und die kundenseitige Zufriedenheit lassen sich auch an den folgenden Auswirkungen festmachen:

- Da ausreichendes Wissen über Abhängigkeiten vorhanden und sichtbar ist, können wir das aktiv beim Testen und der Qualitätssicherung berücksichtigen.
- Brüche in der Software können damit rechtzeitig erkannt und beseitigt werden. Damit ist es uns auch gelungen, die kundenseitig gemeldete Fehlerrate deutlich zu reduzieren.

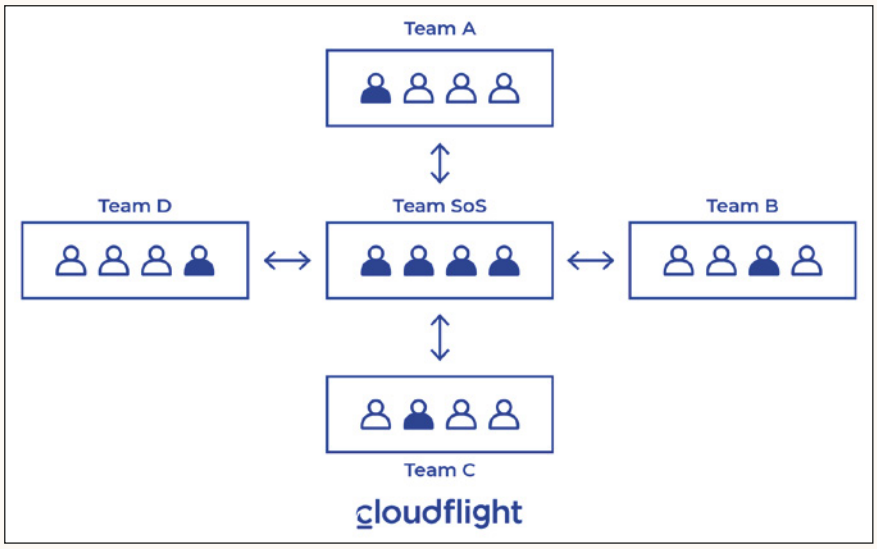


Abb. 1: Scrum-of-Scrum

- Durch signifikantes Reduzieren von Nacharbeiten sparen wir uns nicht nur Zeit und Geld, auch die Motivation und Zufriedenheit im Team wird dadurch spürbar erhöht.

Kultur, Kommunikation und Organisation

Wie kann man nun in großen Projekten dafür sorgen, dass jeder stets zu den Informationen kommt, die für erfolgreiches Arbeiten benötigt werden? Mit zunehmender Anzahl der Projektbeteiligten steigen die Komplexität und der Kommunikationsaufwand exponentiell, sodass es immer schwieriger wird, den Überblick zu bewahren. Agile Methoden, wie etwa Scrum, haben sich in der Praxis zwar bewährt, trotzdem ist es nicht einfach, die gewünschte Planungs- und Umsetzungsqualität einzuhalten.

Die folgenden drei Verfahren kommen bei uns situationsbezogen zum Einsatz:

Ein gängiger Ansatz liegt in einer Erweiterung der Projektorganisation, indem eine *zusätzliche Ebene* eingezogen wird. Diese kann aus übergeordneten Rollen bestehen. Hierbei ist „übergeordnet" nicht notwendigerweise hierarchisch zu verstehen. Es kann sich dabei auch um bestehende Teammitglieder handeln, die sich über die aktuellen Themen abstimmen. Eine solche Vorgehensweise ist mit der Idee von Scrum-of-Scrum vergleichbar **(siehe Abbildung 1)**.

Als Alternative können einzelne Teammitglieder ausgewählt werden, die regelmäßig an den Grooming-, Planungs- sowie den Daily Stand-up-Meetings der jeweils anderen Teams teilnehmen. Als eine Art *Botschafter* sind sie dann für den Wissenstransfer zuständig. Das funktioniert aber nur bei Projekten mit wenigen Teams und ist für große Projekte mit vielen Teams nur schwer skalierbar.

Eine weitere Möglichkeit liegt in der *Team-Rotation*. Teammitglieder werden zyklisch

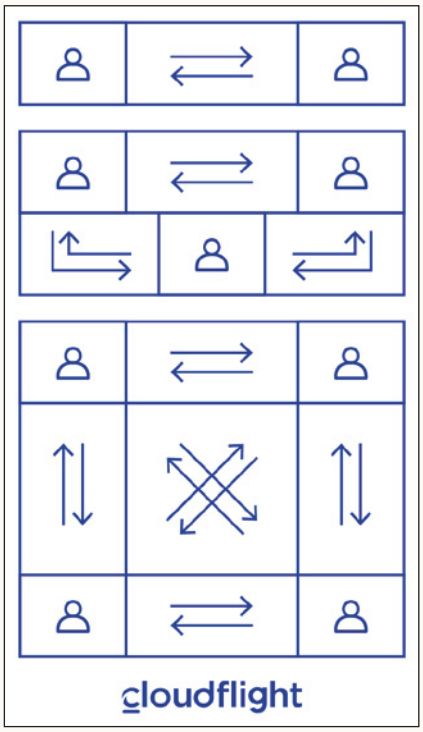


Abb. 2: Vernetzte Kommunikation

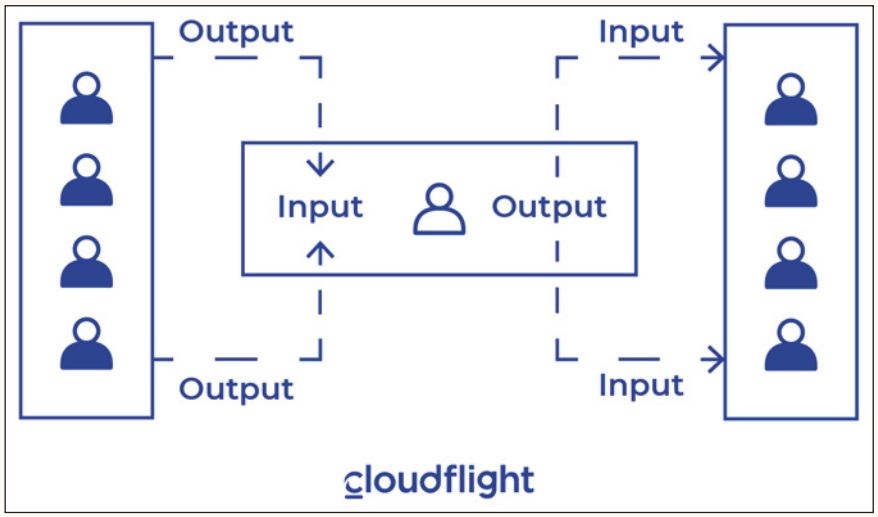


Abb. 3: Kommunikationsknoten und Ausrichtung

(zeitlich begrenzt) in jeweils anderen Teams eingesetzt und dienen danach als Wissensträger. Ebenfalls können auch Pair-Programming sowie Code Reviews über Teamgrenzen hinweg in Betracht gezogen werden.

Bei allen Verfahren ist es nicht nur wichtig, dass kommuniziert wird, sondern auch wie diese Kommunikation erfolgt **(siehe Abbildung 2)**. Die Herausforderungen sowie mögliche Lösungsansätze sollen stets objektiv und sachlich dargestellt und behandelt werden, um einen Mehrwert für die beteiligten Organisationen zu generieren. Diese Herangehensweise inkludiert auch eine entsprechende Vorbereitung der Themen, um dem Gegenüber die Wertschätzung auszudrücken (Harvard-Konzept, s. [HAR]).

Abhängigkeiten managen

Wir verstehen eine Organisation als (komplexes, neuronales) Netzwerk, an dessen Knoten die Menschen stehen. Jeder Akteur ist gleichzeitig Sender und Empfänger. Wir haben es also mit einem gerichteten Graphen zu tun, in dem jeder Knoten von den Vorgängerknoten abhängt und seinerseits die nachfolgenden Knoten beeinflusst **(siehe Abbildung 3)**.

Natürlich handelt es sich dabei immer um eine bidirektionale Kommunikation. Allerdings wollen wir deutlich machen, dass es die „aktive" Verantwortung des „Senders" ist, dass seine „Botschaft" von den Nachfolgern gehört, verstanden und auch zu einem späteren Zeitpunkt wieder aufgefunden werden kann.

Die folgenden Punkte sind wesentliche Bestandteile unserer Unternehmenskultur. Dabei erfordert es allerdings ständiger Anstrengungen aller Beteiligten, damit dieses Wissen und Verhalten auch in den Köpfen der Mitarbeiter verankert bleibt, und stellt eine der Hauptaufgaben unserer Führungskräfte dar:

- Stelle organisatorisch sicher, dass (möglichst) alle Abhängigkeiten und Abweichungen dokumentiert werden.
- Sorge dafür, dass sich die Mitarbeiter jederzeit darüber bewusst sind, welche Konsequenzen ihre Arbeit auf andere haben kann. Auswirkungen werden daher nicht einfach nur dokumentiert, sondern das Wissen auch aktiv verteilt.
- Stelle die notwendigen Werkzeuge zur Verfügung, sodass sich jeder leicht ein Bild davon machen kann, wovon seine aktuelle Arbeit abhängt (Visualisierung).

In unserem Unternehmen ermöglichen wir unseren Mitarbeitern, Kommunikation offen zu leben. Das beinhaltet nicht nur Kommunikation innerhalb der Organisation, sondern auch gegenüber unseren Kunden und Partnern. Eigenverantwortung ist dabei nur eine Säule des Erfolgs, ebenso wie Unterstützung und Training.

Besonders im Bereich Testen und Qualität ist es von entscheidender Bedeutung, Kommunikation zu beherrschen, da hier oftmals nicht nur „gute" Nachrichten überbracht werden. Dabei spielt es keine Rolle, ob hier im Team, über Teamgrenzen oder mit einem Kunden kommuniziert wird – die Kommunikation muss stets konstruktiv erfolgen.

Zusammengefasst ergibt sich daraus der folgende Grundsatz für unsere Arbeitsweise, den jeder einzelne Mitarbeiter verinnerlicht haben muss:

- *Meine Arbeit basiert auf der Arbeit von anderen <=> Meine Arbeit ist die Basis für andere*

Unser Fazit

Fortlaufendes Wachstum mit der Erweiterung um zusätzliche Teams in einem Projekt führt rasch zu immer größeren Qualitätsproblemen. Diese beschränkten sich nicht nur auf die Software, sondern beginnen häufig schon bei der Anforderungsanalyse/-erhebung (Requirements Engineering). Unvollständige, falsche oder inkonsistente Anforderungen setzen sich dann über den gesamten Umsetzungsprozess fort und wirken wie ein Multiplikator in der finalen Fehlerrate.

Damit ein Projekt gemeinsam erfolgreich abgeschlossen werden kann, muss die Erwartungshaltung von Anfang an klar kommuniziert werden. Das betrifft hier nicht nur niedergeschriebene Anforderungen, die relativ einfach validiert und verifiziert werden können, sondern auch sehr viele gefühlte (weiche) Qualitätskriterien, die bereits im Laufe der Entwicklung zutage treten. Hierbei geht es vor allem darum, wie das „finale" Produkt (Potentially Shippable Product [PSP]) vom Kunden wahrgenommen wird. Oftmals werden mögliche Abweichungen (häufig nur durch Unwissenheit oder Verunsicherung ausgelöst) schon durch schnelle und konstruktive Kommunikation einfach aus der Welt geschafft.

Wir wollen noch einmal klar hervorheben, dass ganz wesentlich für den Erfolg die innere Einstellung (das „Mindset") der Mitarbeiter ist: nämlich sich stets eigenverantwortlich im Kommunikationsprozess zu sehen. Um das auch dauerhaft zu gewährleisten, muss seitens der Organisation permanent investiert werden. Dabei ist es wichtig, diese Kommunikation zielgerichtet anzubringen **(siehe Kasten 1)**. Damit hat sich für uns auch die Regel bewahrheitet: „80 Prozent am Erfolg trägt die Organisation bei, 20 Prozent kommen vom richtigen Werkzeug!"

- Produktqualität ist direkt abhängig von der Qualität der Kommunikation in der Projektorganisation.
- Offene Kommunikation und Fehlerkultur fördern und aktiv unterstützen.
- Erwartungshaltung von Anfang an klar kommunizieren.
- Kommunikation ist immer bidirektional.
- Mache deine Arbeit sichtbar!

Kasten 1: Empfehlungen zur innerbetrieblichen Kommunikationskultur

Referenzen

- [CON] Mel Conway's Law, siehe: https://www.melconway.com/Home/Conways_Law.html
- [HAR] R. Fisher, W. Ury, B. M. Patton, Das Harvard-Konzept, DVA, 3. Aufl., 2018
- [PRO] https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/
- [PSP] Potentially Shippable Product, siehe: https://www.mountaingoatsoftware.com/blog/what-does-it-mean-to-be-potentially-releasable

Marco Hampel

marco.hampel@cloudflight.io
ist als Software Quality Coach für große Softwareprojekte im agilen Umfeld tätig. Neben seiner aktuellen Tätigkeit bei Cloudflight verfügt er als Entwickler, Team- und Projektleiter über fünfzehn Jahre Erfahrung in der Softwareentwicklung.

Robert Steinbauer

robert.steinbauer@cloudflight.io
ist Testmanager bei Cloudflight. Er arbeitet seit über zwei Jahrzehnten als Projektmanager, Requirements Engineer und Product Owner an IT-Projekten und Unternehmensentwicklungen.

Know-how (nicht nur) für die Zertifizierung

A. Spillner · T. Linz

Basiswissen Softwaretest

Aus- und Weiterbildung zum Certified Tester – Foundation Level nach ISTQB®-Standard

6. Auflage
2019, 378 Seiten
€ 39,90 (D)
ISBN 978-3-86490-583-4

K. Franz · T. Tremmel · E. Kruse

Basiswissen Testdatenmanagement

Aus- und Weiterbildung zum Test Data Specialist – Certified Tester Foundation Level nach GTB

2018, 208 Seiten
€ 32,90 (D)
ISBN 978-3-86490-558-2

T. Linz

Testen in Scrum-Projekten

Leitfaden für Softwarequalität in der agilen Welt – Aus- und Weiterbildung zum ISTQB® Certified Agile Tester – Foundation Extension

2. Auflage
2017, 270 Seiten
€ 34,90 (D)
ISBN 978-3-86490-414-1

F. Simon · J. Grossmann · C. A. Graf · J. Mottok · M. A. Schneider

Basiswissen Sicherheitstests

Aus- und Weiterbildung zum ISTQB® Advanced Level Specialist – Certified Security Tester

2019, 414 Seiten
€ 39,90 (D)
ISBN 978-3-86490-618-3

R. Bongard · K. Dussa-Zieger · R. Reißing · A. Schulz

Basiswissen Automotive Softwaretest

Aus- und Weiterbildung zum ISTQB® Certified Tester Foundation Level Specialist – Automotive Software Tester

2020, 252 Seiten
€ 34,90 (D)
ISBN 978-3-86490-580-3

M. Baumgartner · S. Gwihs · R. Seidl · T. Steirer · M.-F. Wendland

Basiswissen Testautomatisierung

Aus- und Weiterbildung zum ISTQB® Advanced Level Specialist – Certified Test Automation Engineer

3. Auflage
2021, ca. 292 Seiten
ca. € 34,90 (D)
ISBN 978-3-86490-675-6

dpunkt.verlag

bestellung@dpunkt.de · **www.dpunkt.de**

»Die Bedeutung der Architectural Runway aus Managementsicht«

Neue aufkommende Marktentwicklungen erhöhen den Druck auf die IT der Unternehmen zusehends. ERP-Systeme wurden bisher aufgrund ihrer Größe und Komplexität meist klassisch entwickelt, doch strategische Entscheidungen von Softwareanbietern wie SAP führen immer mehr zu der Notwendigkeit, auch diese Systeme agil zu entwickeln. In der agilen Softwareentwicklung steigt die Bedeutung des automatisierten Softwaretests stark an. Dieser ist entscheidend für die Beschleunigung der Rollout-Frequenz, die Steigerung der Qualität und die Senkung der Transaktionskosten. Voraussetzung dafür ist ein agiles Skalierungsframework mit einer passenden Architectural Runway. Diese bildet die notwendige technisch-architektonische Basis für die Orchestrierung und Weiterentwicklung der Automatisierungsansätze in allen Bereichen. In diesem Beitrag zeigen wir, welche entscheidende Bedeutung die Architectural Runway als technisch-architektonische Basis für die Automatisierung in allen Bereichen hat und welche Vorteile sich für Unternehmen ergeben können.

SAP geht in Richtung Cloud-first, und SAP HANA-bezogene Dienste stellen das am schnellsten wachsende Segment dar. Gleichzeitig sind groß angelegte SAP-Systemimplementierungen durch ständige Veränderungen der Technologie, der Systeme, der Mitarbeiter und der Geschäftsanforderungen und -modelle geprägt.

Marktentwicklungen und Auswirkungen auf das ERP-Ökosystem von Unternehmen

Die Herausforderungen und Nachteile einer klassischen Enterprise Resource Planning (ERP)-Implementierung haben die meisten Unternehmen in der Vergangenheit erlebt. Trotzdem ist der Übergang zu einer skalierten agilen Lieferung nach wie vor selten. Untersuchungen von Gartner zeigen, dass die Unternehmensleitung oft nur über begrenzte Kenntnisse und Erfahrungen in der Anwendung von lean-agilen Prinzipien und Praktiken zur Implementierung und zum Betrieb von ERP-Lösungen verfügt [Gar17]. Der Aufbau dieser Fähigkeit ist aber überlebensnotwendig, um die oben genannten Herausforderungen bewältigen zu können.

Für die erfolgreiche Umsetzung von groß angelegten lean-agilen ERP-Implementierungen, haben sich die folgenden Praktiken besonders bewährt [Acc17]:

- *Planen und organisieren* Sie die Arbeit für die Lieferung innerhalb eines regelmäßigen Entwicklungsrhythmus, einer *Kadenz*, wobei bereichsübergreifende, fähige Teams, die für Effizienz skaliert sind, eingesetzt werden, um die Übergaben zu reduzieren und die Produktivität zu verbessern.
- *Visualisieren* Sie Arbeit und Abhängigkeiten, wobei der Schwerpunkt auf der Begrenzung der Arbeit in Bearbeitung *(WIP)* liegt, um Verzögerungen zu verringern und die Produktivität durch weniger Aufgabenwechsel zu verbessern.
- *Beschleunigen* Sie *automatisierte Tests*, um Transaktionskosten zu reduzieren und die Qualität zu verbessern.

Nachfolgend wird der Schwerpunkt der Betrachtung auf den dritten Punkt, die Beschleunigung, gelegt und der Einfluss der Architectural Runway darauf herausgearbeitet.

Lean-agiler Entwicklungsansatz – Das SAFe®-Framework

SAFe® bietet ein Betriebsmodell, um lean-agile Transformationen von Großunternehmen zu ermöglichen, und liefert zehn kritische Agile Release Train (ART)-Erfolgsfaktoren [SAFeARTSF], die komplexen und großen Unternehmenslösungen ein kontinuierlich profitables Wachstum ermöglichen. Der ART ist ein auf Dauer angelegtes Team von agilen Teams, das zusammen mit seinen Stakeholdern innerhalb eines Value Streams inkrementell eine oder mehrere Lösungen entwickelt, bereitstellt und wo möglich betreibt.

SAFe®s zehn kritische ART-Erfolgsfaktoren

Alle zehn kritischen ART-Erfolgsfaktoren aus **Tabelle 1** sind für das Qualitätsmanagement höchst relevant, dieser Artikel möchte besonders die Bedeutung des Erfolgsfaktors *# 9 – Architectural Runway* hervorheben. Dessen Bedeutung für den Test als Beschleunigungsfaktor entsprechend der dritten Schlüsselpraktik aus obiger Auflistung wird oftmals unterschätzt. Eine ausführliche Beschreibung aller zehn Erfolgsfaktoren findet sich in dem Whitepaper [Kum19].

# 1 - Lean-Agile-Prinzipien	# 6 - Systemdemo
# 2 - Echte Agile Teams und Züge (ART)	# 7 - Inspizieren und Anpassen
# 3 - Kadenz und Synchronisation	# 8 - Innovations- und Planungs-Iteration
# 4 - Program Inkrement Planung	# 9 - Architectural Runway
# 5 - Kundenorientierung, DevOps und Release on Demand	# 10 - Lean-Agile-Führung

Tabelle 1: Die zehn kritischen ART-Erfolgsfaktoren von SAFe® [SAFeARTSF]

Erfolgsfaktor #9 und sein Einfluss als Beschleuniger für den Test

Der Begriff *Architectural Runway* im SAFe® Framework ist auf eine Analogie zur Luftfahrt bei der Beobachtung von Program Increment (PI)-Level Burndown Charts zurückzuführen. Denn wenn die Architectural Runway zu Beginn eines PI zu „kurz" ist, das heißt, sehr viele Features hängen von einer neuen noch nicht fertiggestellten Architektur ab, können die ARTs das PI nicht erfolgreich „landen". Damit ist gemeint, die PI-Ziele werden nicht erreicht, denn das Burndown-Chart am Ende des PI ist nicht auf null gebracht worden. Ein PI benötigt also wie ein Flugzeug eine ausreichend lange Runway (Landebahn). Dieser Analogie folgend, bedeutet dies auch, dass je größer ein System ist und je schneller es fliegt (je höher die Velocity) umso mehr Runway wird für eine sichere Landung benötigt. [SAFeAR]

„Der *Architectural Runway* dient der kurzfristigen Umsetzung von Features. Er besteht aus vorhandenem Code, Komponenten und technischer Infrastruktur, die ohne exzessives Redesign und Verzögerung für die Umsetzung dieser erforderlich sind." [SAFeGl] Der Architectural Runway unterstützt den kontinuierlichen Fluss von „Value" durch die CD-Pipeline, durch die Bereitstellung der

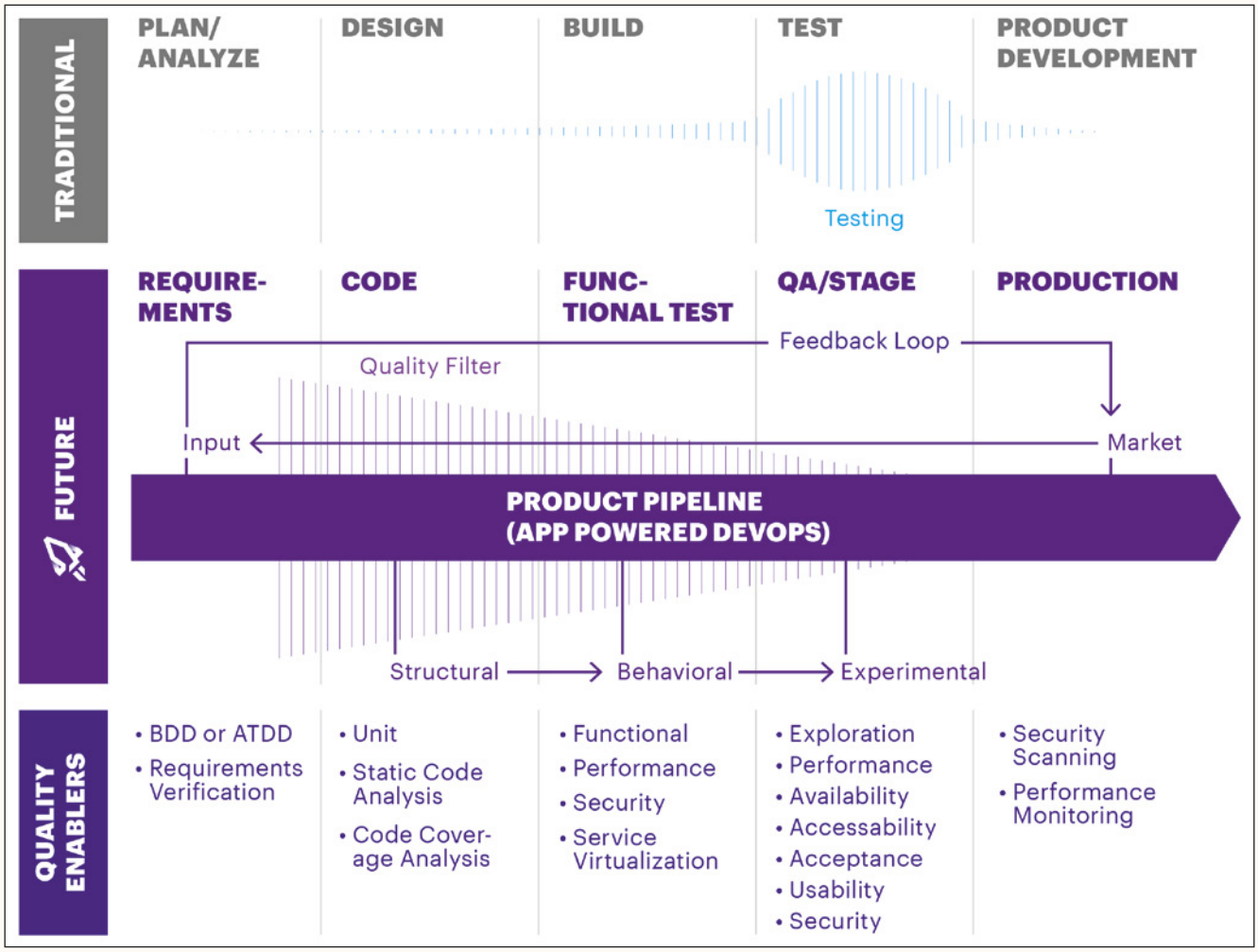


Abb. 1: Eine automatisierte Produkt-Pipeline ist der Kern des Qualitätstrichters und ein wesentlicher Hebel zur Senkung der Transaktionskosten

notwendigen technischen Grundlage. Der Architectural Runway ist dabei das aus unserer Sicht wichtigste Instrument im SAFe® Framework, um die agile Architekturstrategie zu implementieren.

Ein Feature stellt in diesem Kontext einen Service dar, welcher die Bedürfnisse eines Stakeholders erfüllt. Jedes Feature enthält eine Nutzenhypothese sowie Akzeptanzkriterien und ist so klein, dass es ein ART innerhalb eines PI bereitstellen kann. Die permanente Wartung und Ausbau der Architectural Runway durch eine kontinuierliche Investition in „Enabler Stories" unterstützt eine stabile Velocity über alle Teams hinweg. Je größer das zu entwickelnde IT-System ist und je innovativer die technische Plattform (z. B. SAP S/4HANA), desto größer wird die Bedeutung der Architectural Runway, da dann in der Regel mehrere ARTs mit wenig Erfahrung bezüglich der neuen Architektur an der Systementwicklung beteiligt sind. [SAFeAR]

Die Innovationsgeschwindigkeit und die Fähigkeit, schnell und effizient auf Veränderungen zu reagieren, entscheiden darüber, wer auf dem heutigen Markt gewinnt. Eingebaute Qualität, einer der Kernwerte von SAFe®, ist ein integraler Bestandteil bei der Erstellung des Architectural Runway, da sie Systemqualität, Ausführungsgeschwindigkeit und Vorhersagbarkeit gewährleisten kann. Dies erfordert eine Kombination aus technischen Praktiken, Prozessen und einer passenden Unternehmenskultur.

Architectural Runway – Voraussetzung für DevSecOps

Der Architectural Runway bildet somit auch die notwendige Basis für jegliche Art von DevSecOps-Ansätzen und der intensiven Nutzung von Testautomatisierung wie in **Abbildung 1** dargestellt.

DevSecOps schließt die Lücke von der Entwicklungsumgebung zum produktiven Betrieb durch die Bereitstellung von Automatisierungs-, Integrations-, Sicherheits-, Mess- und Wiederherstellungsfunktionen. Es hilft dabei, die verschiedenen Abteilungen, die an der Bereitstellung einer Lösung beteiligt sind, auf ein gemeinsames Ziel auszurichten. Die Einführung von häufigen Integrations- und „Shift-left"-Testmethoden bricht Silos auf und treibt die frühzeitige Fehlererkennung voran [Acc17].

Schlankes Engineering, qualitätssicherungsgeführte Prozesse und eine automatisierte Pipeline bilden den Kern des Qualitätstrichters und sind der Nordstern des agilen Qualitätsmanagements. Neue Automatisierungsfunktionen ermöglichen eine höhere Qualität und Agilität der Software- und Prozessimplementierung, indem Regressionstests und sogar das Testen einer SAP S/4HANA-Implementierung selbst automatisiert werden [Acc19]. Die Transformation von der traditionellen Welt wie in **Abbildung 1** zu einer leanagilen ist ohne eine Architectural Runway als Enabler nicht möglich.

Zwar heißt es im Agilen Manifest, dass „die besten Architekturen, Anforderungen und Entwürfe aus selbstorganisierenden Teams hervorgehen" [MAN], dies bedeutet jedoch nicht, dass „[...] man niemals im Vorfeld Zeit für Architekturentscheidungen aufwenden sollte." [Mar09]

Es ist eine gute Praxis für SAP-Implementierungen, vor Beginn der Implementierung den grundlegenden Architectural Runway und die dazugehörigen Key Design Decisions (KDDs) zu treffen, um eine umfangreiche Neugestaltung zu einem späteren Zeitpunkt zu vermeiden. Dadurch wird der spezifischen Architektur von SAP-Systemen Rechnung getragen.

Die Automatisierung der meisten manuellen Prozesse stellt die Realisierung des ersten Lean-Agile-Prinzips von SAFe®: „take an economic view" dar. Dies ermöglicht die Senkung der Aufwände und der Transaktionskosten im Zusammenhang mit Änderungen

Abb. 2: Accenture-Fallstudie - Vorteile von SAP S/4HANA Automation auf Basis einer Architectural Runway

Referenzen

- [Acc17] Agile for SAP Solutions, Accenture, 2017, siehe: https://accenture.com/_acnmedia/PDF-71/Accenture-Agile-for-SAP-solutions-PoV-nov17-web.pdf
- [Acc19-a] Test Automation, Accenture, 2019, siehe: https://accenture.com/us-en/success-sap-test-automation
- [Ber20] T. Bergander, J. Prumann, DevOps und SAP – Wie passt das zusammen? in: German Testing Magazin, 2/2020
- [Gar17] Gartner, First Steps in Applying Agile and DevOps to ERP Support, 2017
- [Kum19] M. Kumlehn, SAP Delivery Agility – Applying Essential SAFe® to Large-Scale SAP Implementations and Operations, Whitepaper, 2019, siehe: https://www.scaledagileframework.com/?ddownload=46949
- [MAN] Manifesto for Agile Software Development, Juli 2019, siehe: http//agilemanifesto.org
- [Mar09] R. C. Martin, The Scatology of Agile Architecture, Clean Coder, 2009, siehe: https://sites.google.com/site/unclebobconsultingllc/home/articles/the-scatology-of-agile-architecture
- [SAFeAR] Architectural Runway. Scaled Agile Inc.,2020, siehe: https://www.scaledagileframework.com/architectural-runway/
- [SAFeGl] SAFe® 5.0 Glossary. Scaled Agile Framework Terms and Definitions, German, Scaled Agile Inc., 2020, siehe: https://www.scaledagileframework.com/wp-content/uploads/2020/06/Glossary-5.0-Translations-German_v2.pdf
- [SAFeARTSF] Essential SAFe® - Ten Critical ART Success Factors. Scaled Agile Inc., 2020, siehe: https://www.scaledagileframework.com/essential-safe/

signifikant. In **Abbildung 2** ist das Ergebnis einer konsequenten Umsetzung dieses Prinzips zu sehen.

Durch die konsequente Umsetzung aller oben genannten Prinzipien und Empfehlungen ist es gelungen, die S/4HANA-Implementierung mit 0 Fehlern in Produktion live zu setzen. Ein wesentlicher Erfolgsfaktor dafür war die Automatisierungsrate von über 80 Prozent im Produkt- und Regressionstest über alle Teststufen hinweg. Dies führte als positiver Nebeneffekt auch dazu, dass das Testing Budget im Folgejahr trotz der Investitionen in die Architectural Runway um 50 Prozent reduziert werden konnte, aufgrund deutlich niedrigerer Transaktionskosten. Die Grundvoraussetzung für die Realisierung all dieser Leistungen bildete die Architectural Runway. Eine detaillierte Beschreibung, wie genau die Vorteile erzielt werden konnten, findet sich im Whitepaper „Applying Essential SAFe® to Large-Scale SAP Implementations and Operations" [Kum19], da dies den Umfang eines einzelnen Artikels bei Weitem sprengen würde.

Architectural Runway – Eine ökonomische Notwendigkeit im Zeitalter der Digitalisierung

Zusammenfassend bleibt festzuhalten, dass agile Softwareentwicklung zu einer steigenden Bedeutung des automatisierten Softwaretests führt, da dieser entscheidend ist für die Beschleunigung der Rollout-Frequenz, für die Steigerung der Qualität und für die Senkung der Transaktionskosten. Darüber hinaus erzeugt der hohe (Zeit-)Druck Fehler im manuellen Test und damit Folge- und Fehlerkosten. Testautomatisierung hilft also, diese Probleme zu vermeiden.

Notwendige Voraussetzung hierfür ist eine passende Architectural Runway. Diese bildet die technisch-architektonische Basis für die Orchestrierung und Weiterentwicklung der Automatisierungsansätze in allen Bereichen. Eine detaillierte technische Antwort auf die Frage, wie dies konkret im SAP-Kontext umgesetzt wird, findet sich in [Ber20] in diesem Heft.

Thomas Karl

thomas.karl@accenture.com
ist Organizational Change Manager, LSS MBB, SAFe® SPC5, IC-Agile Enterprise Agile Coach, ISTQB Full Advanced Level zertifiziert, Kanban Trainer und Systemischer Coach. Die Schwerpunkte seiner Tätigkeit sind Qualitätsmanagement und die Entwicklung von lean-agilen Organisationen von der Team- bis zur Strategie- & Portfolio-Ebene. Er ist Sprecher auf internationalen Konferenzen und berät Vorstände und Führungskräfte im agilen Wandel.

Malte Kumlehn

malte.kumlehn@accenture.com
ist der erste SAFe® Fellow, IC-Agile Enterprise Agile Coach und (Remote-) Instructor für komplexe SAP®-Lieferung. Als SAFe® Fellow ist Malte Vordenker und verfügt über umfangreiche Erfahrung in lean-agilen SAP-Liefer- und Betriebsmodellen. Malte hilft Führungskräften und Organisationen dabei, die tiefgreifende Veränderungen von Mitarbeitern, Organisationen und Technologien schnell und in großem Maßstab anzugehen, um nachhaltig profitables Wachstum zu erzielen.

»Der Faktor Mensch«

Testautomatisierung bewältigt eine Vielzahl von Herausforderungen – ein Mythos jedoch ist der Glaube, dass die Qualität der Software alleine durch Testautomatisierung sichergestellt werden kann. Heutzutage sollte man auch das Feedback von echten Nutzern einbeziehen und den Faktor Mensch bei der Entwicklung digitaler Produkte berücksichtigen – erfahren Sie hier, welche Möglichkeiten es gibt, seine Kunden und Fans zu einem Teil der digitalen Produktentwicklung zu machen.

Die wesentlichen Merkmale einer agilen Entwicklung sind ihre Schnelligkeit und ein hohes Maß an Flexibilität. Zeit und Ressourcen sind begrenzt, und daher ist es oft schwierig, Tests in Sprints zu integrieren, auch wenn dies unerlässlich ist, um die Qualitätsanforderungen der Benutzer zu erfüllen. Agile Entwicklungsprozesse sollen dabei die Transparenz und Flexibilität erhöhen und zu einem schnelleren Einsatz der entwickelten Systeme führen.

Testautomatisierung in agilen Projekten

Agile Softwareentwicklung stellt aber nicht nur an die Entwicklungs- und Testteams spezielle Herausforderungen, sondern auch an die Testautomatisierung – unter anderem verlangt agile Softwareentwicklung zwingend nach agilen und möglichst vollautomatisierten Tests, die ein schnelleres Ausliefern ermöglichen. Außerdem wird ein ausgereiftes Automatisierungsframework ausschlaggebend, damit die Teams neue Testfälle schnell skripten und hinzufügen können.

In jedem Fall sind automatisierte Tests aber mit einem erheblichen Aufwand verbunden. Die Zeitersparnisse, die in der Praxis erreicht werden, können diesen natürlich ausgleichen. Agile Methoden eignen sich aber nicht für jedes Entwicklungsprojekt. Sie funktionieren nur im Zusammenspiel mit einer agilen Unternehmenskultur. Die stärkere Einbindung des Kunden und die Selbstorganisation der Entwicklerteams sind Gefahren, die einer erfolgreichen Umsetzung oftmals im Wege stehen.

Crowdtesting – Das Testen von Software mithilfe der Internetgemeinde

Damit die Qualität unter diesen Voraussetzungen dennoch erstklassig ist, benötigen Unternehmen ein ebenso agiles und professionelles Test-Management, bei welchen auch der menschliche Faktor berücksichtigt wird. Egal, ob agile Entwicklung oder Wasserfallmodell – Testing ist ein fester Bestandteil des Software Development Lifecycles und Crowdtesting kann bei beiden Methoden eine sinnvolle Ergänzung sein.

Bei der Auslagerung von Testprojekten an die Internetgemeinde ist es wichtig, die Zuständigkeiten und Verantwortlichkeiten klar zu definieren und Crowdtesting sinnvoll in die Gesamtstrategie einzubetten. Auch wenn die Crowd schnell, flexibel und sehr kurzfristig einsatzbereit ist, so muss die Testing-Phase dennoch von Grund auf realistisch geplant und die passende Testart für das jeweilige Projekt oder die jeweilige Entwicklungsstufe gefunden werden.

Die Anforderungen an die Tester sind dabei von Projekt zu Projekt unterschiedlich. Um die passenden Tester auszuwählen, sollten möglichst viele relevante Informationen über diese zur Verfügung stehen. Während bei Usability-Tests Personen ausgewählt werden, die den Merkmalen der potenziellen Zielgruppe entsprechen, sind die demografischen Faktoren bei funktionalen Tests kaum von Bedeutung. Hier sind vielmehr die Erfahrung und die zur Verfügung stehenden Geräte entscheidend.

Einer der kritischsten Faktoren für die erfolgreiche Durchführung von Remote Tests ist aber die Dokumentation der Ergebnisse. Die Tester müssen ihr Vorgehen so ausführlich und präzise wie möglich beschreiben. Die eingereichten Berichte und Bugs müssen im Anschluss vom Testleiter auf Vollständigkeit, Nachvollziehbarkeit sowie Qualität überprüft werden. Haben die Tester wirklich auf den angegebenen Systemen getestet? Wurden alle Arbeitsschritte abgearbeitet? Welchen Schweregrad hat ein Bug?

Wie eine erfolgreiche Durchführung von Remote Tests in der Praxis aussehen kann, erfahren Sie im folgenden Praxisbeispiel.

Individueller Testaufbau mithilfe der Crowd

Connected-Car-Apps bieten heute eine Vielzahl nützlicher Funktionen – Nutzer erhalten zahlreiche Möglichkeiten zur Navigation, dem Tracking der Fahrt sowie dem Abrufen von Fahrzeugdaten. Die App wird dabei via Bluetooth mit dem Motorrad verbunden und spiegelt die Features auf dem TFT-Display des Motorrads wider. Der entscheidende

Punkt ist hierbei das Zusammenspiel von App und Motorrad-Hardware.

Crowdtesting stellt dabei eine mögliche Ergänzung zu traditionellen Testmethoden dar, da es sich auf unvoreingenommene reale Benutzer stützt. Die Aufgabenstellung an die Tester in unserem Praxisbeispiel war kurz und einfach: Sie wurden gebeten, die Beta-Version einer neuen Connected-App in typischen Anwendungsfällen mit ihrem Motorrad zu testen. Dadurch sollten Fehler und Funktionalitätsprobleme bei Bedienung und Nutzererlebnis gefunden sowie sämtliche auftretende Bugs aufgedeckt werden. Die Use Cases enthielten regelmäßig genutzte Anwendungsfälle wie allgemeine Einstellungen, die Verbindung von Smartphone und Motorrad über Bluetooth, die Navigation, das Abrufen von Fahrzeug- und Tour-Daten sowie das eigenständige Entdecken der App.

Im Rahmen parallel ablaufender Bug-Tests wurde die App mit simulierten Daten gefüllt, um häufig auftretende Fehler finden zu können sowie Lokalisierungstests auf unterschiedlichen Smartphones durchzuführen. Die Bereitstellung der Ergebnisse erfolgte dabei über eine direkte Schnittstelle zum JIRA-System der App-Entwickler, wodurch Bugs im Rahmen der agilen Arbeitsumgebung der Entwickler direkt behoben werden konnten.

Aufgrund besonderer Anforderungen waren die Tester-Rekrutierung und das Community-Management in diesem Fall eine besondere Herausforderung. Gemeinsam mit dem App-Entwickler wurde die Crowd über diverse Online-Kanäle wie einschlägige Foren, Facebook-Gruppen und Motorrad-Websites angesprochen.

In Remote Interviews konnte den Testern bereits vor Release der App Screens der neuen Version gezeigt und somit zusätzlicher Input und Verbesserungsvorschläge eingeholt werden. Bei dieser Art von Interviews können Tests live mitverfolgt werden, was ungefilterte Einblicke in die Gedanken der Tester ermöglicht. Durch die parallele Echtzeit-Kommunikation mit den Testern können gezielt Fragen gestellt und unmittelbares Feedback gesammelt werden.

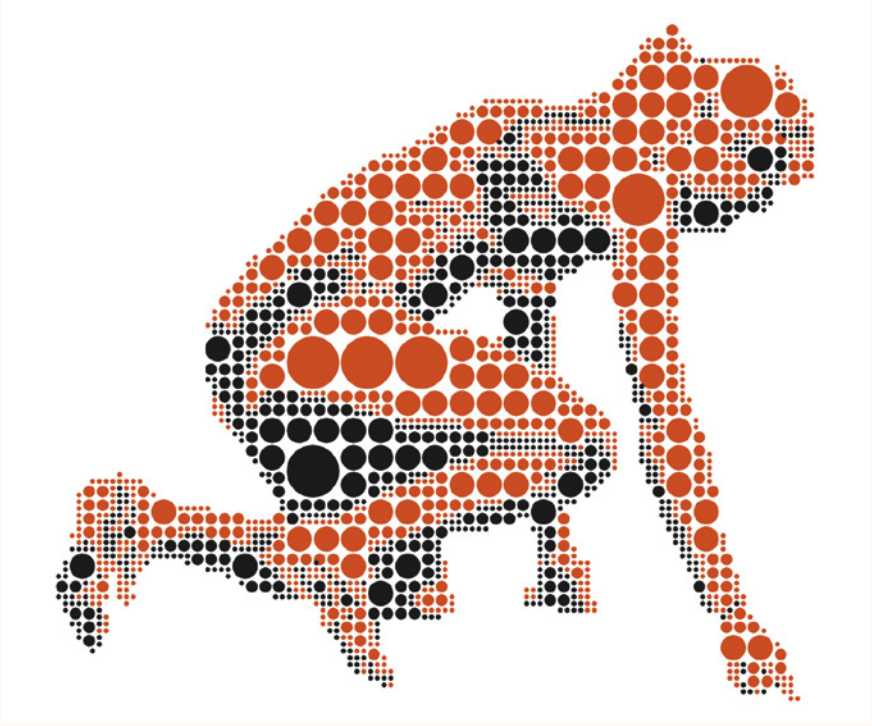
Alles in allem gibt es viel zu bedenken, wenn Sie Crowdtesting in Ihren Development Lifecycle einbinden wollen – doch dies sollte Sie nicht abschrecken, denn die Vorteile überwiegen den Aufwand um ein Vielfaches und ebnen Ihnen den Weg, Software zu entwickeln, die Ihre Kunden gerne nutzt.

Georg Hansbauer

g.hansbauer@testbirds.com
ist Mitgründer und Managing Director des Softwaretesting-Spezialisten Testbirds. Er verantwortet die Weiterentwicklung der Dienstleistungen und der IT-Infrastruktur. Georg verfügt über umfassende Erfahrung im Feld des Enterprise Testing – von automatisierten Tests kompletter IT-Service-Desks bis hin zu Lasttests – und hat zahlreiche IT-Projekte für internationale Konzerne betreut. Er absolvierte den Elitestudiengang „Finanz- und Informationsmanagement" an der Universität Augsburg und der TU München und besitzt einen Bachelor in Wirtschaftsinformatik.

»Wie passen DevOps und SAP zusammen?«

Aller Anfang ist schwer. Gerade haben sich Geschäftsbereiche und IT einer Unternehmung für eine Transformation von bestehendem SAP ECC in Richtung SAP S/4HANA entschieden. Es wurde klar, dass es um Wertschöpfungspotenziale und nachhaltigen Erhalt des Marktvorteils geht. Lean, Agile und DevOps sind wahrscheinlich die am häufigsten angeführten Schlagwörter auf dem Weg dorthin. Angestrebt wird die Verkürzung der Einführungszeiten von Produktverbesserungen und damit eine signifikant verbesserte Kundenzufriedenheit. Insbesondere der Anfang einer solchen Transformation ist von entscheidender Bedeutung.



Im Folgenden werden wir im Kontext der Anfangsphasen einer solchen Transformation und an einem Praxisbeispiel einige Antworten aufzeigen zu den Fragen: Welche Fähigkeiten sind zu mobilisieren? Was muss für einen erfolgreichen „Sprint 0" bereitgestellt werden? Wie kann ich manuelle Aktivitäten vermeiden? Wie stellt man sicher, dass Automatisierungspotenziale geschaffen und im Transformationsverlauf mit entsprechender Qualität auch ausgeschöpft werden? Wie passen also DevOps und SAP zusammen?

Mobilisierung

Zwei Ressourcenprofile, die von zentraler Bedeutung sind, aber oftmals erst in späteren Phasen eingesetzt werden, sind *Quality Engineer (QE)* mit Automatisierungsexpertise und *DevOps Engineer*. Idealerweise haben beide ein gutes Verständnis der SAP S/4HANA-Geschäftsprozesse und von gängigem Tooling der Lieferprozesse für SAP- und Non-SAP-Lösungen. Diese Profile auf Expertenniveau sind rar und damit relativ schwer zu beschaffen.

Gleichzeitig sind diese Experten unabdingbar, um das Fundament für eine erfolgreiche SAP-Transformation richtig zu gestalten **(siehe Abbildung 1)**. Sie müssen das Zielbild des Toolings erarbeiten und auf das spezifische Liefervorgehen abstimmen. Dabei muss die Automatisierung sowohl von Lieferprozessen als auch Tests berücksichtigt und mittels einer übergreifenden Strategie für die gesamte Transformation nutzbar gemacht werden. Erschwerend kommt hinzu, dass dies in einer Phase geschehen muss, in der praktisch alles „nur auf Folien" existiert.

Die Folgen einer zu späten Einbindung von Quality Engineer und DevOps Engineer zeigen sich dann oft zu Beginn von „Sprint 1". Die Teams generieren keine automatisierten Tests, sondern erstellen stattdessen manuelle Testdokumente. Damit ergeben sich Zusatzaufwände und Abhängigkeiten zwischen den Tests, respektive Testern, der einzelnen Teams – was zu vermeiden gewesen wäre.

Sprint 0

Vereinfacht gesagt, handelt es sich bei „Sprint 0" um eine klassische Fit-Gap-Analyse, bei der festgestellt wird, in welchen Bereichen Diskrepanzen zwischen SAP-Standardprozessen und den gewünschten Arbeitsabläufen bestehen. Allerdings werden hier User-Storys anstelle von fachlichen Anforderungen definiert.

Einerseits fordert die Arbeit mit User-Storys eine möglichst hohe Flexibilität, um die Dynamik und Volatilität im Backlog-Management in den Griff zu bekommen (z. B. splitting und grooming). Andererseits basieren SAP-Prozesse und deren Dokumentation auf relativ statischen, hierarchischen Strukturen, die meist in Form von Best-Practice-Sammlungen, auch „Accelerators" genannt, geliefert werden und bei der Transformation einen Schnellstart speziell in „Sprint 0" ermöglichen sollen. Erschwerend kommt hinzu, dass das gesamtheitliche Zielbild der Geschäftsprozesse sowohl SAP als auch Non-SAP-Lösungskomponenten umfassen muss.

Dementsprechend müssen das Backlog-Management und die Solution Documentation (kurz: SolDoc, eine mehrstufige hierarchische Repräsentation der Geschäftsprozesse in den 5 Ebenen L1-L5) zunächst technisch getrennt aufgesetzt werden und es muss lediglich eine lose Kopplung erfolgen. Dazu wird das Zielbild der Geschäftsprozesse bis auf Ebene L3 möglichst stabilisiert und im Backlog-Management als führende Struktur zu Beginn von „Sprint 0" hinterlegt. Nur in der SolDoc wird bis auf L5 hinab modelliert. Strukturelle Abstimmungen zwischen beiden erfolgen lediglich bis auf L3. Dadurch wird der Synchronisationsaufwand minimiert, die Agilität im Backlog-Management maximiert und eine

relativ leicht zu stabilisierende hierarchische Struktur in der SolDoc sichergestellt.

Stichwort *Accelerators*. Es empfiehlt sich, die Gesamtheit der einzusetzenden Accelerators beziehungsweise Best-Practice-Sammlungen frühzeitig zu analysieren. Die tatsächliche Nutzbarkeit dieser Artefakte muss im Detail betrachtet werden, da sich sonst damit versprochene Aufwandsreduktionen nicht realisieren lassen oder gar Mehraufwand generiert wird.

Stichwort *Schätzung*. Hierbei ist zwingend zu beachten, dass bereits ab „Sprint 1" lauffähige Teilprodukte geliefert werden müssen. Dies beinhaltet auch eine möglichst vollständig automatische Testbarkeit. Die damit verbundenen Aufwände hängen von der „Passgenauigkeit" der Accelerators und den Automatisierungsfähigkeiten in den Teams ab – beides ist unbedingt im gewählten Schätzvorgehen zu berücksichtigen.

Stichwort *Fähigkeiten*. Die Teams sind frühzeitig an die zu nutzenden Tools, wie Jira, SAP Solution Manager und Tricentis Tosca, heranzuführen. Ein möglichst praxisorientiertes und intensives Training zu Beginn von „Sprint 0" verbunden mit kontinuierlichem Coaching, Austausch und Wiederholungen ist notwendig, um erfolgreiche Sprints zu ermöglichen.

Stichwort *Berechtigungen*. Ein Albtraum für die Liefernden – unbedingt so schlank wie irgend möglich gestalten. Grundprinzip: Alle können alles, was nicht irreversibel ist. Ganz wenige dürfen auch irreversible Sachen.

Besondere Aufmerksamkeit sollte auch der Lizenzierung geschenkt werden beim Aufspringen auf bereits vorhandene Lösungen. Oftmals ist es kosteneffizienter, eine zusätzliche Instanz extra für das Transformationsvorhaben zu akquirieren, anstatt ein dediziertes Projekt in einer bestehenden Instanz zu nutzen, weil die Abrechnung in vielen Fällen instanzweit über alle Nutzer erfolgt. Zudem sind projektbezogene Anpassungsmöglichkeiten oft eingeschränkt durch instanzweite Vorgaben – was wiederum nachteilig für die notwendige Flexibilität ist.

Sprint 1

Die User-Storys sind hinreichend detailliert, evaluiert und priorisiert im Backlog vorgehalten. Im ersten Program Increment Planning (PIP) wird der umzusetzende Lieferumfang festgelegt und die entsprechenden User-Storys werden für „Sprint 1" ausgewählt. Jetzt müssen Backlog-Management (z. B. Jira) und Entwicklungsumgebung (z. B. SAP Solution Manager, SolMan) so verbunden werden, dass möglichst einfach das Reporting auf Programm-Ebene gemacht werden und gleichzeitig jedes Team detaillierte und problemspezifische Analysen auf der gleichen Datenbasis durchführen kann. Ziel ist es, die Durchlaufzeiten (Lead Times) im Lieferprozess schnell zu verstehen, Optimierungspotenziale zu identifizieren und zu quantifizieren und entsprechend messbare Maßnahmen zur Verbesserung umzusetzen – Continuous Improvement.

Schritt *Null*. Der End-to-End-Lieferprozess wird möglichst detailliert von den entsprechenden Experten erarbeitet in Abstimmung mit den Lieferverantwortlichen. Die aus Erfahrung wahrscheinlichen Treiber für Lead Times sowie sich anbietenden Automatisierungspotenziale werden identifiziert, priorisiert und in zeitliche Abhängigkeit bezüglich Transformationsvorhaben gebracht.

Schritt *Eins*. Die Statusabfolgen für lieferungsrelevante Artefakte in den genutzten Tools (z. B. Jira und SAP SolMan) werden konsistent aufeinander abgestimmt. Die Pflege der Status wird in die Verantwortung der Teams gegeben. Durch Training und Coaching wird für die zeitnahe und akkurate Pflege der Status sensibilisiert. Hierzu sollte, wenn irgend möglich 1-1-Beziehungen zwischen Artefakten in unterschiedlichen Tools hergestellt werden, zum Beispiel eine User-Story (in Jira) entspricht einem Work Package oder Change Dokument (in SAP SolMan).

Schritt *Zwei*. Frühzeitig und regelmäßig sind die Komponenten der Lead Times im Lieferprozess zu analysieren und entsprechende Verbesserungsmaßnahmen umzusetzen. Bei Wartezeiten in der Genehmigungskette sollten die verursachenden Genehmigungsschritte eliminiert oder vereinfacht werden.

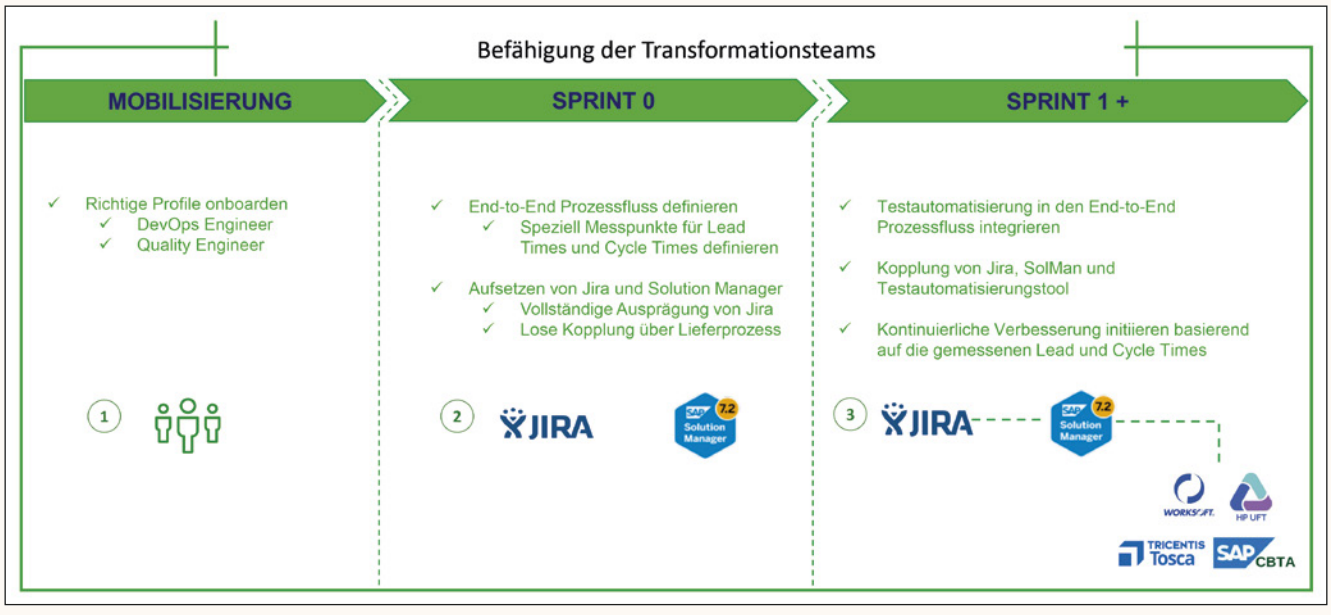


Abb. 1: Befähigung der Transformationsteams

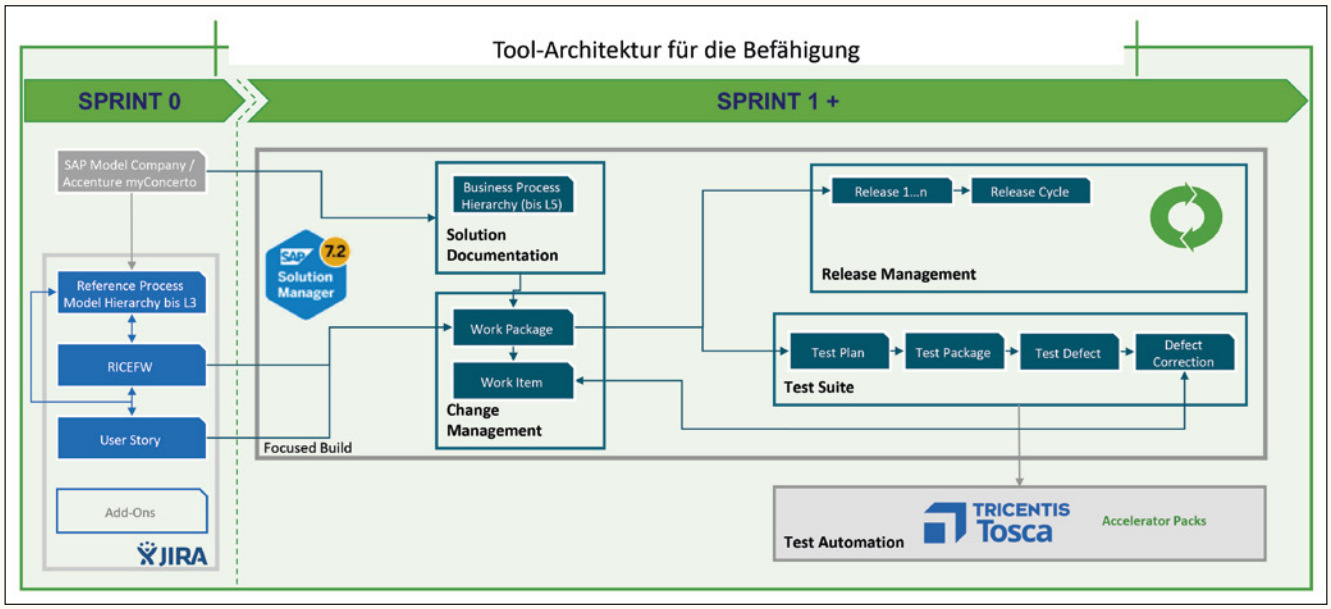


Abb. 2: Tool-Architektur für die Befähigung

Zum Beispiel bei Verzögerungen im Statusabgleich bietet sich ein automatisiertes Interface zwischen den beteiligten Tools an (z. B. User-Story – Change-Dokument zwischen Jira und SAP SolMan).

Schritt *Drei*. Wiederholen von Schritt Zwei mit weiteren Komponenten der Lead Times oder Umsetzungszeiten (Cycle Times) im Lieferprozess. Zum Beispiel statusbasiertes Anstoßen von automatisierten Tests oder Ausnutzung vorhandener automatisierter Tests zur Sicherung der Lieferqualität mit geringerem Ressourceneinsatz (siehe auch [DOS]).

Praxisbeispiel

Im folgenden Praxisbeispiel stellen wir eine DevOps-Toolkette und die Lean Quality Engineering-Vorgehensweise vor, die wir bereits bei mehreren Kunden erfolgreich implementiert haben **(siehe Abbildung 2)**.

Jira

Zu Beginn stellt sich die Frage, welches Tool zur Organisation der agilen Arbeitsweise einzusetzen ist. In unserem Beispiel hatte der Kunde das Tool Jira bereits im Einsatz, sodass wir darauf aufgesetzt haben. Vorhandenes Tool-Wissen beschleunigt den Design- und Implementierungsprozess, Installationsaufwände neuer Software entfallen, Trainings können wiederverwertet werden. Somit konnten unmittelbar die Arbeitsweise der agilen Teams betrachtet werden, Anforderungen an Jira aufgenommen und zügig in die Konfiguration übergegangen werden.

Das Programm erhielt seine eigene Jira-Instanz, um Freiheitsgrad und Kosteneffizienz zu maximieren. In der Instanz wurde ein programmspezifisches Projekt angelegt. Im Anschluss wurden die minimal benötigten Issue Types definiert, die Anforderungen der agilen Teams in Konfigurationen übersetzt und der Workflow verfeinert. Im Sinne der agilen Selbstorganisation gibt es zwei Rollen. Einige User sind Projekt-Admins, die restlichen User dürfen den gesamten Workflow mit minimalen Einschränkungen bearbeiten. Wir setzen auf Vertrauen und Selbstverantwortung – unser Motto: „Kontrolle ist gut, Vertrauen ist besser".

Einige handverlesene Add-ons erleichtern uns zukünftig die Arbeit. Zeitnahes Reporting, Backlog-Analysen, Dashboard-Erweiterungen waren mittels verfügbarer Add-ons in kürzester Zeit vorhanden! Hier hat auch COVID-19 seine Spuren hinterlassen. PI Plannings in einer umfangreichen SAP-Implementierung mussten zeitnah virtualisiert werden. Auch hierfür wurde ein Add-on eingesetzt, das die Anforderungen erfüllte. Hier spielte die eigene Jira-Instanz ihre Vorteile aus, denn die Lizenzen mussten lediglich für die Anzahl unserer Projektuser erworben werden. Damit konnte die User-Story-Schlacht beginnen.

SAP

Die Verbindung von agilen Methoden, Jira und SAP-Entwicklung war als Nächstes zu meistern. Die Lösung: SAP SolMan 7.2 mit Focused Build. Selbstverständlich wäre die Toolkette ohne Focused Build möglich, dazu hätten wir jedoch den gesamten ChaRM-Prozess abstimmen, aufsetzen und konfigurieren müssen – zu zeit- und kostenintensiv. Focused Build kam bereits vorkonfiguriert und war hinreichend passgenau auf unsere Anforderungen.

Als erster Schritt wurde im SAP SolMan die SolDoc zur Verfügung gestellt. Hier wird die gesamte Dokumentation der Entwicklungen abgelegt und verwaltet. Die SolDoc ist der Grundstein für Focused Build. Work Packages und die entsprechenden Umsetzungsaufgaben (Work Items) werden auf Basis der in der SolDoc verwalteten Prozessstruktur erstellt.

Und nun? Wir haben Issue Types in Jira, eine Prozessstruktur und Work Packages sowie Work Items im SolMan. Wir wollten flexibel bleiben und zunächst keine technische Integration zwischen Jira und SolMan herstellen – verbinden mussten wir die Systeme aber trotzdem irgendwie. Also wurde der relativ stabile Teil der Prozessstruktur in Jira hochgeladen und ein Hyperlink zur SolDoc eingebaut. Die Standard-Rollen in Focused Build waren allerdings zu komplex und wirkten als Bremse auf die agile Vorgehensweise. Um

Referenzen

- [AMC] Accenture myConcerto with new SAP solutions, siehe: https://www.accenture.com/us-en/services/sap/myconcerto-sap-integration
- [DOS] DevOps for SAP, siehe: https://www.accenture.com/t00010101T000000Z__w__/de-de/_acnmedia/PDF-93/Accenture-SAP-Testing.pdf#zoom=50
- [SMC] SAP Model Company, siehe: https://www.sap.com/services/implementation/preconfigured-industry-solutions.html

Effizienz und Reibungsfreiheit zu schaffen, wurden die Rollen auch hier verschlankt. Die einen durften bis zur Produktion, die anderen auf die Produktion. Fertig.

Aber was kommt vor der Produktion? Testing, Testing und noch mehr Testing. Die Konfiguration der SAP-Test-Suite lief parallel, außerdem wurde Tricentis Tosca zur Testautomatisierung in die Toolkette aufgenommen und mit SolMan integriert.

Geschwindigkeit

Eine Prozessstruktur als Grundlage der Implementierung aufzubauen, dauert seine Zeit und ist kostspielig. Auch hier konnten wir Vorhandenes wiederverwenden. Dazu nutzen wir ausgewählte Komponenten der Model Company von SAP [SMC] sowie myConcerto von Accenture [AMC]. Die Prozessstruktur sowie der Inhalt musste lediglich in die SolDoc importiert werden und schon konnten die funktionalen Teams mit ihrer Arbeit beginnen.

In Workshops wurden die Prozesse auf Basis von funktionierenden System-Demos analysiert und der Input der verschiedenen Accelerators auf das Unternehmen angepasst. Ein entscheidender Zeit- und Qualitätsvorteil gegenüber einer Prozessmodellierung ohne Grundlage.

Accelerator-Packs für SAP von Tricentis beschleunigten die Testautomatisierung durch vorgefertigte Standardmodule zusätzlich und dienten als Grundstein für die Erstellung eigener Testsequenzen passend zu den modellierten Prozessen.

Testautomatisierung Schritt für Schritt

Im Kontext von SAP und DevOps spielt das Thema Testautomatisierung eine entscheidende Rolle. Es legt den Grundstein für eine schlanke Qualitätssicherung und damit verbundene Testoptimierungsstrategie. Gleichzeitig vermindert frühzeitige Automatisierung von Tests die Abhängigkeiten zwischen den Teams, da nicht mehr auf teamexterne Tester zurückgegriffen werden muss, zum Beispiel im Falle von vorgängig auszuführenden (manuellen) Tests in abhängigen, aber teamexternen Prozessketten.

Manuelle Schritte wie einfache Unittests und später funktionale Tests wurden mithilfe von Tricentis Tosca so automatisiert, dass letztendlich nur ein Knopfdruck in SolMan notwendig war, um einen Test in Tosca durchzuführen, entsprechende Fehler zu finden und die Resultate in SolMan zurückzumelden. Die Automatisierung kann so von Anfang an in allen Bereichen genutzt und damit nach und nach erweitert werden.

Im weiteren Verlauf folgten Tests von längerer Dauer und höherer Komplexität (z. B. Integrations- und Regressionstests). Dabei wurden die modularisierten Testbausteine zu längeren Prozessketten zusammengefügt und auf Wiederverwendung der Module geachtet.

Zu Beginn jedes automatisierten Testlaufs mussten im Rahmen des Sprints die notwendigen Testdaten generiert werden (z. B. Anlegen eines Kundenauftrags). Hierzu wurden möglichst bereits vorhandene, automatisierte Testmodule verwendet, sodass eine Abhängigkeit zu statischen Anfangsbedingungen bezüglich Datenkonstellation minimiert wurde.

Die Pflege der zu verwendenden Testdaten lag in der Verantwortung des Fachexperten und nicht beim Testautomatisierer, um eine größere Testabdeckung und Testrelevanz zu erzielen. Durch die Automatisierung und Verkettung der einzelnen Module waren wir in der Lage, die Anwendung wesentlich häufiger und einfacher zu prüfen und damit frühestmöglich äußerst zielgenau eine Korrektur nachzuschieben und damit die Qualität wesentlich zu erhöhen.

Wichtig zu berücksichtigen: Die Testskripte müssen laufend angepasst werden. Bei einer kontinuierlichen Weiterentwicklung der Anwendungen ist das mitunter herausfordernd und nicht berücksichtigte Funktionen verursachen Seiteneffekte wie beispielsweise falsche Ergebnisse oder Fehler bei der Ausführung von automatisierten Tests. Entscheidend ist hier, auf ein Automatisierungstool der neusten Generation zurückzugreifen, welches die agile Entwicklung und die andauernden Veränderungen schnell umsetzen kann. Dabei sollte es nicht auf Programmierung basieren, damit nicht die Wartungsaufwände den schlanken Ansatz zerstören.

Hybrid-Agiles Testing

DevOps ist im Kern transformativ und erfordert moderne Ansätze wie Agilität, die insbesondere durch Quality Engineering unterstützt werden. Dennoch bietet die Kombination aus klassischem Wasserfall und neuen agilen Methoden größte Flexibilität und gleichzeitig Planungssicherheit im Gesamttestzyklus. Wir bildeten ein agiles Projekt ab, welches schrittweise in kleinen Produktinkrementen getestet und geliefert wurde. Der Testansatz musste entsprechend ganzheitlich in die Projektarbeitsweise integriert werden, um eine einheitliche und komplett durchdachte Teststrategie zu definieren.

Ganz klassisch erfolgte die Definition der Teststrategie und Testplanung, gefolgt von agilen Testansätzen für die Testdurchführung. Agiles Coaching, Change-Management und entsprechende Testressourcenplanung waren kritische Voraussetzungen für die testgetriebene Entwicklung. Für manuelles Testen haben wir uns für die Test-Suite SolMan 7.2 entschieden, da diese leicht mit Tricentis Tosca integriert werden kann. Sie bietet die Verwaltung aller Testaktivitäten von der Testplanung bis hin zur Testausführung, welches ein wichtiges Werkzeug für den Testmanager darstellt.

Einzelne Unittests und funktionale Tests als Fertigstellungskriterien im Sprintzyklus stellten sicher, dass die Qualität frühzeitig geprüft wurde und die User-Story-Abnahme iterativ erfolgte. Frühzeitige Priorisierung und Beseitigung von Fehlern sowie Integra-

tionstest-Phasen gegen Ende des Produktinkrements trugen dazu bei, die Qualität des gelieferten Produkts und den Wissensstand speziell des Quality Engineering-Personals kontinuierlich zu verbessern.

Ein neuer Testansatz bedeutet aber auch, dass man alte Traditionen brechen muss. Durch die starke Unterstützung der Führungskräfte wurde der organisatorische Wandel der Tester-Rolle erfolgreich eingeführt. Die Bildung eines Teams, in welchem fachliche und technische Kenntnisse vorhanden und jeweils Tester integriert sind, führte zu einem großen Mehrwert durch verbesserte Qualität und beschleunigte die Erstellung und Wartung der Testautomatisierung. Damit brachte der Wandel vom klassischen Testen zum Quality Engineering den entscheidenden Erfolg.

Torsten Bergander

torsten.bergander@accenture.com
ist SAP DevOps Advisor mit langjähriger Erfahrung in der Lieferung von großen Transformationsprogrammen mit überwiegendem SAP-Anteil. Seine Expertise liegt dabei in der durchgängigen Gestaltung der Lieferorganisation, Umsetzung von agilen Ansätzen und Einsatz von DevOps für SAP. Insbesondere konzentriert sich seine Erfahrung auf die Ingangsetzung von Programmen und in Folge den optimierten Einsatz von Automatisierungskonzepten speziell für Liefer- und Testprozesse – was er bei weltweit führenden Kunden in unterschiedlichen Branchen bereits mehrfach erfolgreich umgesetzt hat.

Jennifer Prumann

jennifer.prumann@accenture.com
ist Quality Engineer Advisor mit langjähriger Erfahrung im Bereich Test und Design großer IT-Lösungen mit Spezialisierung auf Testmanagement sowohl für komplexe kundenspezifische als auch für SAP-Lösungen. Ihre Expertise liegt dabei sowohl auf funktionalen Tests als auch auf automatisierten Lösungen. Sie hat einschlägige Erfahrung im Aufbau einer Testfabrik und in der komplexen Lösungslieferung bei führenden europäischen Kunden in verschiedenen Branchen. Darüber hinaus hat sie als Quality Engineer SME an Testassessments und Teststrategieberatungen gearbeitet.

Zusammenfassung

DevOps ist das Weiterdenken von Agilität. SAP und DevOps passen sehr wohl zusammen. Eine schrittweise Einführung von DevOps-Elementen ganz an den jeweiligen Bedürfnissen der Lieferphasen orientiert – speziell die verzögerte integrative Kopplung von Tools und die Automatisierung von Anfang an – sind zentrale Erfolgsfaktoren für das Transformationsvorhaben. Richtig eingeführt mithilfe der Experten im Bereich Quality Engineering und DevOps Engineering können die Erwartungen der Organisation hinsichtlich der Schaffung eines Wettbewerbsvorteils Schritt für Schritt erfüllt und nachhaltig gesichert werden.

Impressum

VERLAG:

SIGS DATACOM GmbH
Lindlaustraße 2c, D-53842 Troisdorf
Tel.: +49 (0) 2241/2341-100
Fax: +49 (0) 2241/2341-199
www.sigs-datacom.de
E-Mail: info@sigs-datacom.de

CHEFREDAKTION:

Dr. Stephan Weißleder und Richard Seidl

SCHLUSSREDAKTION:

Annette Weinert
E-Mail: schlussredaktion@objektspektrum.de

VERLAGSLEITUNG:

Günter Fuhrmeister
Emanuel Rosenauer
E-Mail: emanuel.rosenauer@sigs-datacom.de

LESERSERVICE:

info@sigs-datacom.de

VERTRIEBSLEITUNG:

Florian Bender
E-Mail: florian.bender@sigs-datacom.de

GRAFIK & PRODUKTION:

F&W Perfect Image GmbH
Am Oberfeld 1 · D-83026 Rosenheim,
E-Mail: info@perfectimage-gmbh.de

Gestaltung Titelseite: Sarah Filla

DRUCK:

F&W Druck- und Mediencenter GmbH
Holzhauser Feld 2 · D-83361 Kienberg
Tel.: +49 (0) 8628/9884-52
E-Mail: anzeigendaten@objektspektrum.de

BILDNACHWEISE:

Titelbild, 7, 82: Shutterstock
Seiten 21, 32, 50, 68, 71, 74, 78, 84: Pixabay
Seite 54: Photo by Jamie Street on Unsplash
Seiten 4-9, 18-19, 24-26, 33-45, 51-53, 74, 81-83, 87-90: Accenture · Seite 62: iStock

MANUSKRIPTEINSENDUNGEN:

Manuskripte werden von der Redaktion – ausschließlich in elektronischer Form – gerne angenommen. Mit der Einsendung von Manuskripten gibt der Verfasser die Zustimmung zum Abdruck, zur Vervielfältigung auf Datenträgern und zur Übersetzung.

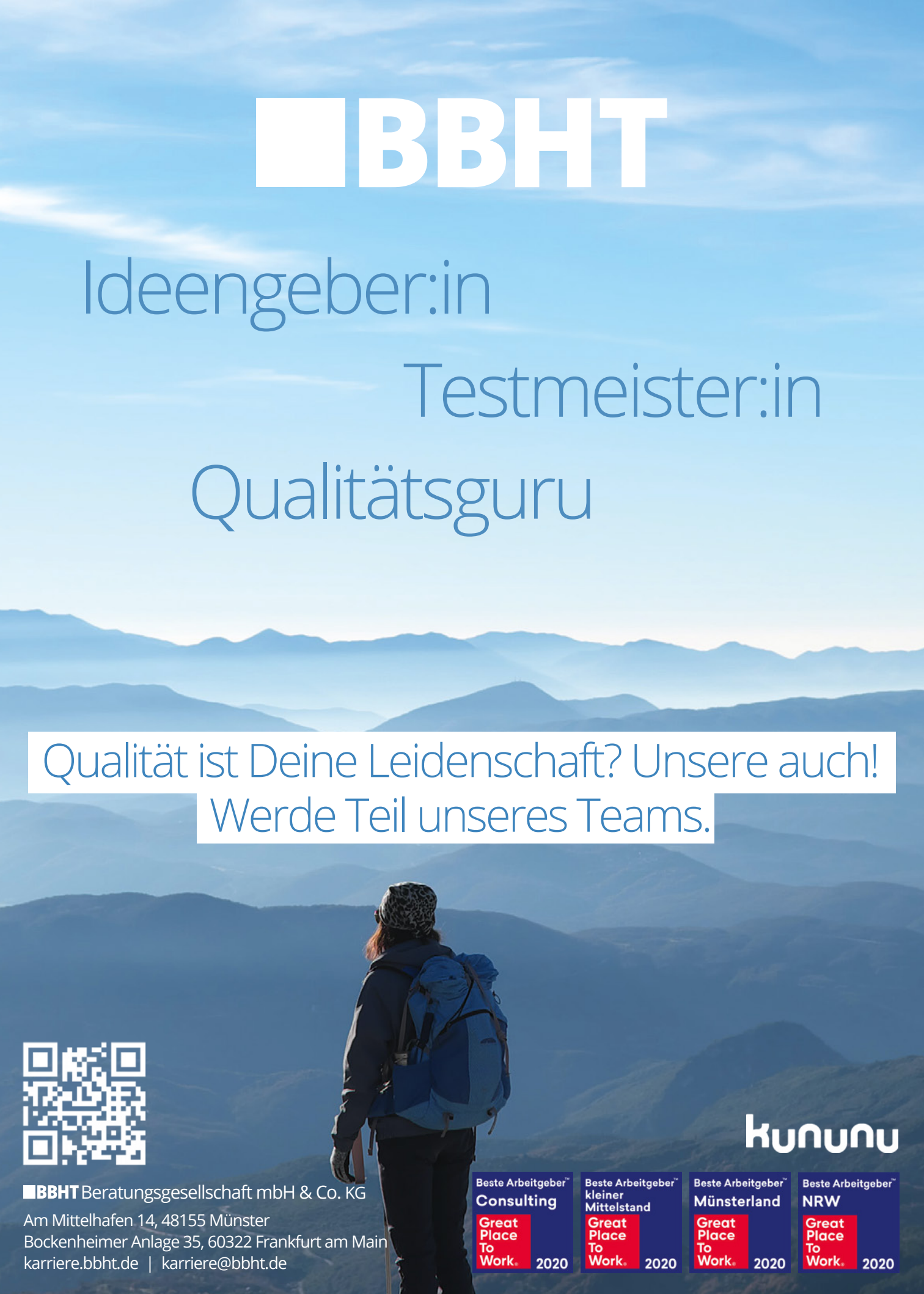
URHEBERRECHT:

Für Programme, die als Beispiele veröffentlicht werden, kann der Herausgeber weder Gewähr noch irgendwelche Haftung übernehmen. Aus der Veröffentlichung kann nicht geschlossen werden, dass die beschriebenen Lösungen oder verwendeten Bezeichnungen frei von gewerblichen Schutzrechten sind. Die Erwähnung oder Beurteilung von Produkten stellt keine irgendwie geartete Empfehlung dar. Für die mit Namen oder Signatur gekennzeichneten Beiträge übernehmen der Verlag und die Redaktion lediglich die presserechtliche Verantwortung.

COPYRIGHT:

2020 SIGS DATACOM GmbH

Das Magazin entstand
in enger Zusammenarbeit mit
Accenture Dienstleistungen GmbH
Campus Kronberg 1
D-61476 Kronberg im Taunus
Web: https://www.accenture.com



BBHT Beratungsgesellschaft mbH & Co. KG
Am Mittelhafen 14, 48155 Münster
Bockenheimer Anlage 35, 60322 Frankfurt am Main
karriere.bbht.de | karriere@bbht.de

accenture

Make Software Solutions more State-of-the-Art.

Make Accenture more you. Mit deinen Talenten und Stärken werden wir jeden Tag ein bisschen besser. Komm in unser agiles Team der Möglichmacher, um gemeinsam Innovationen zu entfachen und die Welt zu verbessern, in der wir leben und arbeiten.

Nutze moderne Tools, Technologien und Methoden und lebe einen Qualitätsstandard für unsere Software Engineering Services. Jetzt bewerben: accenture.de/software-engineering

Let there be change