TESTEN IN AGILEN PROJEKTEN

Agile Entwicklung

Agile Entwicklung verspricht schnellere »Time-to-Market« bei gleichzeitig besserer Ausrichtung des Produkts an den Kundenanforderungen.

Bild: Shutterstock / Den Rise

Und nicht zuletzt auch bessere Qualität. So erstaunt es nicht, dass agile Softwareentwicklung heute weit verbreitet ist – vom Start-up-Unternehmen bis zur Produktentwicklung im Großkonzern - quer durch alle Branchen.

Nachhaltiges agiles Arbeiten ist jedoch nicht ganz einfach. Einen sehr großen Einfluss darauf, ob ein Team, eine Softwareabteilung oder ein ganzes Unternehmen agiles Entwickeln langfristig erfolgreich beherrscht und so die erhofften Vorteile nachhaltig realisieren kann, hat die Art und Weise, wie das Testen der Software gestaltet wird.

In diesem Artikel wird deshalb beleuchtet, was agiles Testen auszeichnet, welche Techniken einem agilen Softwareteam zur Verfügung stehen und worauf zu achten ist, wenn das Testen in einem agilen Projekt verbessert werden soll.

Wodurch unterscheidet sich das Testen in einem klassischen Projekt vom Testen in einem agilen Projekt?

Sicher ist, es bedeutet nicht, dass weniger oder nur oberflächlich getestet wird. Agiles Testen heißt, dass das Team bei Auswahl, Planung und Umsetzung seiner Testaktivitäten den Prinzipien des Agilen Manifests folgt und diese konsequent auch auf das Testen anwendet.

Im agilen Projekt, insbesondere wenn nach Scrum gearbeitet wird, hat jede Iteration beziehungsweise jeder Sprint das Ziel, eine potentiell lieferfähige Software, auch »potentially shippable product« genannt, zu produzieren. Für den Test bedeutet das: Das Testen muss im Takt der Iterationen stattfinden. Alle nötigen Test- und QS-Arbeiten müssen daher immer im Sprint zwingend miteingeplant werden. Es gibt keine Testphase nach dem Sprint. Und es gibt auch keinen Teilprojektleiter Test, der vorgibt, was und wie zu testen ist.

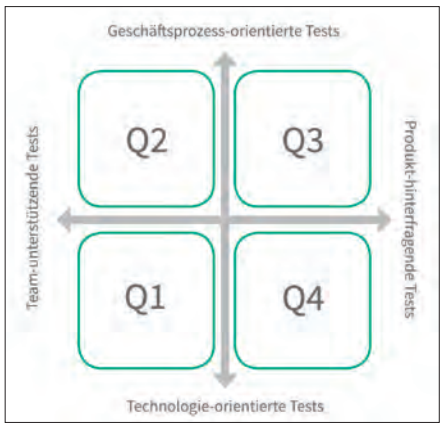
Das Team ist verantwortlich und zuständig. Testaufgaben werden daher genauso wie alle anderen Entwicklungsaufgaben über das (Sprint-)Backlog gesteuert. Dabei können die Testaufgaben entweder explizit als eigenständige Tasks dargestellt, geplant und überwacht werden oder implizit als Teil der Done-Kriterien anderer Entwicklungs-Tasks.

Die Erfassung und Auswertung von Testfortschritt und Testergebnissen erfolgen, bei funktionierender Continuous Integration, hoch automatisiert, sodass manuelle Aufgaben in diesem Bereich minimiert sind. Wenn Testinfrastruktur oder Testwerkzeuge unzureichend sind oder fehlen oder die softwaretestbezogene Ausbildung von Teammitgliedern verbessert werden muss, dann sind das typische »impediments«, um

die sich der Scrum Master kümmern und die er abstellen muss.

In agilen Projekten spielt das Verwalten von gefundenen Fehlern eine untergeordnete Rolle, da der Test parallel während der Realisierung eines Tasks stattfindet. So werden gefundene Fehler an den zuständigen Entwickler zur Korrektur weitergegeben, während dieser noch an der Realisierung arbeitet. Welche Tests zu welchem Zeitpunkt die richtigen sind, damit dieses Fast-Feedback funktioniert, bestimmt die Teststrategie.

Wie viel Testaufwand angemessen ist, hängt davon ab, wie viel Risiko im jeweiligen Feature beziehungsweise in dessen Implementierung steckt. Jede Iterationsplanung sollte daher eine solche Risikoanalyse beinhalten. Anhand des Konzepts der Testquadranten kann das Team überprüfen, ob es alle wichtigen Testarten auf den dazu passenden Teststufen durch Testfälle berücksichtigt hat. Das Modell klassifiziert Testarten entlang zweier Achsen: Auf der Y-Achse werden technologieorientierte (technology facing) vs. geschäftsprozessorientierte (business facing) Tests unterschieden. Auf der X-Achse teamunterstützende (support programming, support team) vs. produkthinterfragende (critique, evaluate product) Tests (Bild 1). Daraus resultiert die Einteilung von Testarten in die vier folgenden Quadranten:



Testquadranten: Mit diesem Konzept kann das Team überprüfen, ob es alle wichtigen Testarten auf den dazu passenden Teststufen durch Testfälle berücksichtigt hat (Bild 1)

- **Q1:** Die Programmierer unterstützende technologieorientierte Tests zur Absicherung neuen und geänderten Codes, realisiert durch automatisierte Unit- und Integrationstests.
- **Q2:** Geschäftsprozessorientierte Tests zur Absicherung neuen und geänderten Codes, realisiert durch manuelle und automatisierte funktionale Tests auf Systemebene.
- **Q3:** Geschäftsprozessorientierte manuelle Tests, die das Produkt hinterfragen, realisiert durch explorative Tests, Usability-Tests, Benutzerakzeptanz-, Alpha- und Beta-Tests.
- **Q4:** Technologieorientierte Tests, die das Produkt hinterfragen, wie Performanz-, Last-, Stress- und Skalierbarkeitstests, Tests auf Zugriffssicherheit (Security), Wartbarkeit, Kompatibilität, Interoperabilität, Datenmigration, Infrastruktur und Wiederherstellung.

Um den richtigen Mix im richtigen Umfang aus all diesen möglichen Testarten zusammenzustellen und dies auch ▶

Die führenden Macher der TestBench Cloud Services

Die Firma imbus AG aus dem mittelfränkischen Möhrendorf ist ein Lösungspartner für professionelles Testen von Software und intelligente Qualitätssicherung von Software. Rund 280 Mitarbeiter bieten Beratung zu Prozessverbesserung, Softwaretest-Services, Testoutsourcing, Testwerkzeuge und Training. Alle imbus-Testexperten sind ISTQB Certified Tester. Das International Software Testing Qualifications Board (ISTQB) hat das Unternehmen 2016 mit dem Status als Global Partner ausgezeichnet. Im ISTQB, dem GTB sowie weiteren Fachverbänden und Gremien (zum Beispiel DIN/ISO) arbeiten die imbus-Testexperten auch aktiv an der Weiterentwicklung der Fachdisziplin Software-Qualitätssicherung und Softwaretest mit. In bisher über 6.000 Projekten hat imbus dazu beigetragen, dass Software-basierte Produkte und die IT im Unternehmen zuverlässig funktionieren. Die führenden Macher von imbus im Kurzportrait:

Tilo Linz

Tilo Linz ist Vorstand und Mitgründer der imbus AG, einem führenden Lösungsanbieter für Softwaretest und seit mehr als 20 Jahren im Themengebiet Softwarequalitätssicherung und Softwaretest tätig. Die vielfältigen Chancen, aber auch Herausforderungen, die sich aus der Einführung und Anwendung agiler Methoden ergeben, kennt und erlebt er täglich aus nächster Nähe: in Softwareprojekten seiner Kunden, in der imbus-internen TestBench-Produktentwicklung, aber auch außerhalb der Softwareentwicklung, zum Beispiel im imbus-Marketing, wo er ein an Kanban orientiertes agiles Marketing eingeführt hat.

Dierk Engelhardt

Dierk Engelhardt ist Produktmanager der TestBench – das Testmanagement- und Testdesignwerkzeug der imbus AG, das eines der innovativsten seiner Klasse ist. Mit seiner langjährigen Erfahrung berät und begleitet Dierk Engelhardt Kunden bei der Einführung und maßgeschneiderten Integration von Werkzeugen in den Software-Entwicklungsprozess. Weiterhin berät er beim Aufbau von agilen Teams und der Integration des Tests in den agilen Kontext.

Thomas Schulte

Thomas Schulte ist seit November 2017 Head of Development der TestBench Cloud Services. Davor war er in unterschiedlichen Bereichen des IT Consulting und der Entwicklung tätig. Er hat in verschiedenen Rollen alle Teile des Software Life Cycle Managements – von der Planung über die Entwicklung bis hin zum Support – abgedeckt.

immer wieder an die sich von Iteration zu Iteration verändernde Situation anzupassen, ist ein ausreichendes Know-how und Erfahrung im Fachgebiet Softwaretest Voraussetzung. Vielen agilen Teams mangelt es jedoch an Expertise über Softwaretest.

Der Scrum Master sollte deshalb darauf achten, dass mindestens eine Person im Team über eine entsprechende Ausbildung, zum Beispiel als Certified-Tester und Erfahrung als professioneller Softwaretester verfügt. Diese Person steuert dann ihre spezielle Expertise bei, um die Softwaretests inhaltlich fachgerecht, risikoorientiert und wirtschaftlich für alle vier Quadranten aufzusetzen. Sie berät auch den Product Owner bei der Bewertung der Produktqualität und im Vorfeld von Produktfreigaben.

Es spricht nichts dagegen, diese Person auch im agilen Team Testmanager zu nennen, auch wenn sich die Rolle inhaltlich zur Qualitätsberatung verschiebt. Einen guten Beitrag können natürlich auch externe Testspezialisten leisten, die zur methodischen Unterstützung angefordert werden und dann im Team mitwirken.

Buch: Testen in Scrum-Projekten

Das Buch von Tilo Linz (ISBN PDF: 978-3-86491-279-5) behandelt und erläutert den Stoff des ISTQB Certified Tester – Foundation Level Extension Syllabus Agile Tester (Version 2014). Mitte 2016 haben weltweit über 4000 Personen dieses auf dem Foundation Level aufbauende Zusatzzertifikat absolviert. Das ISTQB geht hier von hohen Wachstumsraten aus. Weitere Lehrplanmodule für agile Tester auf Advanced Level werden deshalb in einer ISTQB-Arbeitsgruppe, in der der Autor mitarbeitet, vorbereitet beziehungsweise diskutiert.

Viele Aspekte, die beim Testen in agilen Projekten eine Rolle spielen, werden im Buch umfassender und über den ISTQB-Lehrplan hinausgehend behandelt. Hierzu gehören insbesondere die Erläuterungen zum Unit Test und Integrationstest und die Gegenüberstellung der Qualitätsmanagementansätze in klassischen vs. agilen Projekten.

In der 2. Auflage wurden Erläuterungen zu verschiedenen Begriffen ergänzt oder erweitert, so zum Beispiel in Kapitel 3 die Begriffe User Stories, Release- und Sprint-Planung, agile Testpraktiken, Sprint Null sowie die Konzepte der Testpyramide und der Testquadranten. In Kapitel 6 wird der Zusammenhang zwischen User Stories, Akzeptanzkriterien, Akzeptanztests und Abnahmetest-getriebener Entwicklung besser dargelegt sowie der Begriff »Hardening Iteration« vorgestellt. Kapitel 8 wurde um eine weitere Fallstudie (die aus der englischen Ausgabe übernommen wurde) ergänzt.

Open Activities: 13 offene Aktivitäten, in drei verschiedenen Dringlichkeiten (Bild 2)

Das Testen muss im Takt der Iterationen stattfinden! Damit dies dauerhaft und nachhaltig gelingt, muss es soweit wie möglich durch Tools beschleunigt und automatisiert werden. Die meisten agilen Projekte investieren daher zu Recht stark in automatisierte Unit-Tests. Ziel muss hier sein, möglichst jede Methode jeder Klasse der in Entwicklung befindlichen Software durch geeignete Unit-Tests abzusichern. Es gibt allerdings weitere Tests und Testaufgaben, die nicht durch Unit-Tests abgedeckt werden können.

Beispiele dafür sind Last-Tests oder Usability-Tests (siehe die genannten Testquadranten). Auch funktionale Systemtests, die systemweite Workflows oder Geschäftsprozesse aus Endanwendersicht abprüfen, sind nicht durch Unit-Tests darstellbar. Solche Tests erfordern andere, spezifische Testautomatisierungstools.

Manuelles Testen

Manuelles Testen bleibt ebenfalls wichtig. Nicht für jedes Feature, das neu entwickelt wird, ist genug Zeit, die zugehörige Testautomatisierung im selben Sprint mit zu realisieren. Und wenn unklar ist, wie das betreffende Feature letztlich genau aussehen wird, ist eine zu frühe Investition in die Automatisierung der zugehörigen Tests Verschwendung. Manuelles Testen ist dann wirtschaftlicher!

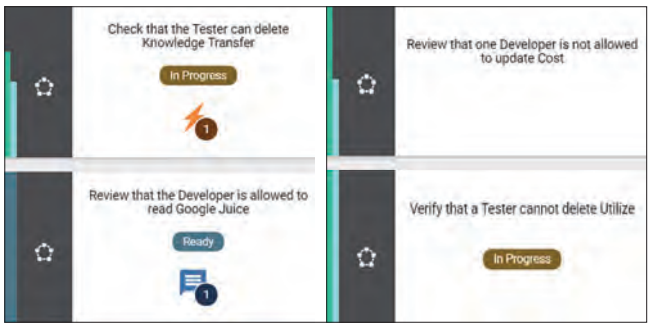
Neben einer Automatisierung der Testdurchführung ist auch eine gute, effiziente Tool-Unterstützung der Abläufe und Tätigkeiten im Testmanagement notwendig. Zwei wesentliche Fragen muss ein hierzu geeignetes Tool jedem Teammitglied schnell und einfach beantworten:

- Welche Test-relevanten Aufgaben stehen für mich an?
- Welchen Qualitäts-Level hat unser Produkt bzw. haben wir ein »shippable product«?

Kann die erste Frage nicht beantwortet werden oder gehen die Test-relevanten Tasks in der Flut der Programmiertasks unter, wird das Team von den Testaufgaben überrascht. Das für die Programmierer essentielle Fast-Feedback kommt dann regelmäßig zu spät oder bleibt ganz aus. Wird Frage Zwei nicht beantwortet, unterschätzt das Team was an Testarbeit, Bugfixing und Regressionstest noch alles zu tun ist, um das »shippable product« tatsächlich abzuliefern.

Viele agile Teams nutzen Jira zur Verwaltung ihrer Testtasks. Das ist sicherlich besser, als Testaufgaben gar nicht zu verwalten oder mit Excel zu arbeiten. Auch verschiedene Jira-Plugins helfen, Testtasks zu organisieren. Aus unserer Sicht hat aber der Einsatz eines spezialisierten agilen Testmanagement-Tools deutliche Vorteile. Ein neues Tool in dieser Kategorie ist TestBench Cloud Services.

TestBench Cloud Services realisiert konsequent das Ziel, dem Team auf maximal einfache Weise über den Stand der



Test Cases: Vier Testfälle einer User Story mit ihren offenen Aktivitäten, dem aktuellen Status der Testdurchführung und den Statistiken (Bild 3)

Testarbeiten und den Stand der Produktqualität kontinuierliches Feedback zu liefern. Dazu kann das System nicht nur Testaufgaben, Testfälle und Defects erfassen und verwalten, sondern auch Epics und User Stories. Das Tool stellt die Verbindung zwischen allen Elementen her, so dass sämtliche Informationen direkt an den für das agile Team maßgeblichen Elementen, den User Stories, ablesbar werden: Testprozeduren, Testdaten, Testergebnisse und Defects.

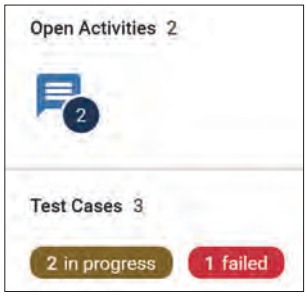
TestBench Cloud Services

TestBench Cloud Services zeigt dem Nutzer dabei nicht nur die noch zu erledigenden Aktivitäten an, sondern auch deren Dringlichkeit und kumuliert diese Information bis auf die Ebene der User Stories und Epics (Bild 2).

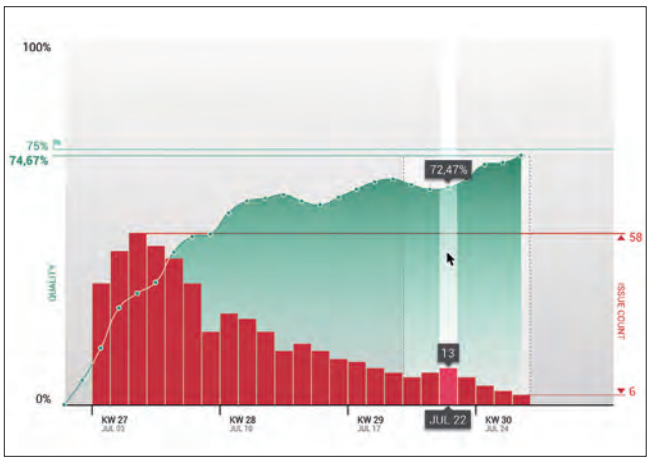
Jedes Element besitzt zusätzlich seine eigene Fortschrittsstatistik. Diese zeigt dem Benutzer den Fortschritt der Testspezifikation, aber auch die Erfüllung der Testvoraussetzungen, wie beispielsweise das Vorhandensein der nötigen Testdaten. Auch hier werden die Informationen auf den oberen Ebenen kumuliert dargestellt.

Damit der aktuelle Stand der Testergebnisse jederzeit ablesbar ist, werden die Status der Testfälle ebenfalls auf allen Ebenen visualisiert. So ist auch der Bezug zwischen einem gefundenen Defect und der zugehörigen User Story jederzeit sichtbar (Bild 3).

An den Testfällen kann der Status direkt abgelesen werden. Auch welcher Testfall bereit für die Testdurchführung ist (Status »Ready« mit einer offenen Aktivität »Testfall durchführen«). Auf der Ebene der zugehörigen User Story werden diese Status dann kumuliert über alle Testfälle dieser User Story visualisiert (Bild 4).



An den Testfällen kann der Status direkt abgelesen werden (Bild 4)



Product Quality Metric: Entwicklung der abgesicherten Produkt Qualität über die Sprints (Bild 5)

Neben den gefundenen und noch nicht korrigierten Fehlern wird zusätzlich die durch Tests abgesicherte Produktqualität dargestellt. Die Produktqualität berechnet sich dabei aus der Anzahl der mit „passed" gelaufenen Testfälle in Relation zur Abdeckung der User Stories mit hinterlegten Testfällen (Bild 5).

Fazit

Mit TestBench Cloud Services steht für agile Teams eine schnell und einfach einsetzbare Lösung bereit, um Test und Qualitätssicherung agil zu organisieren. Das agile Team muss hier keine Ressourcen investieren, um ein generisches Taskmanagement-Tool aufwändig anzupassen oder nachzurüsten. Es kann sich stattdessen voll und ganz auf die Erledigung der eigentlichen Aufgaben fokussieren und TestBench Cloud Services das Testmanagement und das Reporting überlassen.

Interessierte Leser können eine zeitlich auf 90 Tage verlängerte Trial-Lizenz **(Code: TBWMD052018)** kostenfrei unter **90days.testbench.com** abrufen. ■

Links zum Thema

- Agiles Testen
  de.wikipedia.org/wiki/Agiles_Testen
- Brian Marick
  www.exampler.com/old-blog/2003/08/21/#agile-testing-project-1
- International Software Testing Qualifications Board
  www.istqb.org
- Testtool Review Informations-Portal
  www.testtoolreview.de
- Jira
  www.atlassian.com/software/jira
- TestBench Cloud Services
  www.testbench.com