Teststrategie festlegen und Teststufen aufeinander abstimmen

Bereich
Projektplanung und -steuerung

Aktivität
Projekt planen

schnell durchführbar ★★☆

einfach durchführbar ★★☆

agil einsetzbar ★★★

Ziele

- Effiziente Testausführung
- Vermeidung von doppelter Arbeit

Motivation/Problemstellung

Bevor mit konkreten Testaktivitäten gestartet wird, stellt sich zunächst die Frage, mit welcher Teststrategie die Entwicklung begleitet und ergänzt werden soll, um die gewünschte Qualität unter weiteren Projektfaktoren wie verfügbarer Zeit und Ressourcen, dem konkreten Entwicklungsprozess oder Projektzielen erreicht werden kann. Dazu stellt sich ebenfalls die Frage, was auf welcher Teststufe getestet werden soll, und was diese Teststufen miteinander in Verbindung bringt. Werden diese Stufen nicht miteinander koordiniert und aufeinander abgestimmt, so entsteht sehr häufig eine ineffektive Testdurchführung bzw. unnötiger Mehraufwand durch doppelte Prüfungen.

Kurzbeschreibung

Diese Best Practice beschreibt Möglichkeiten, wie der Test ausgerichtet werden kann. Zudem wird der Fokus der einzelnen Teststufen im Testprozess definiert und wie diese Teststufen ineinander greifen, um doppelte Prüfungen zu vermeiden. Zu den Teststufen gehören in der Regel der Entwicklertest (Modultest), der Integrationstest, der Systemtest und der Akzeptanztest. Jede dieser Teststufen hat einen bestimmten Fokus und sollte für bestimmte Zwecke genutzt werden. Die durch eine Teststufe abgedeckten Prüfungen müssen auf anderen Teststufen nicht mehr wiederholt werden. In einer Teststrategie, die für das Projekt erstellt wird, sowie eine Detailplanung für jeden Sprint, sollte das Vorgehen frühzeitig definiert werden, um eine effiziente Testausführung zu gewährleisten.

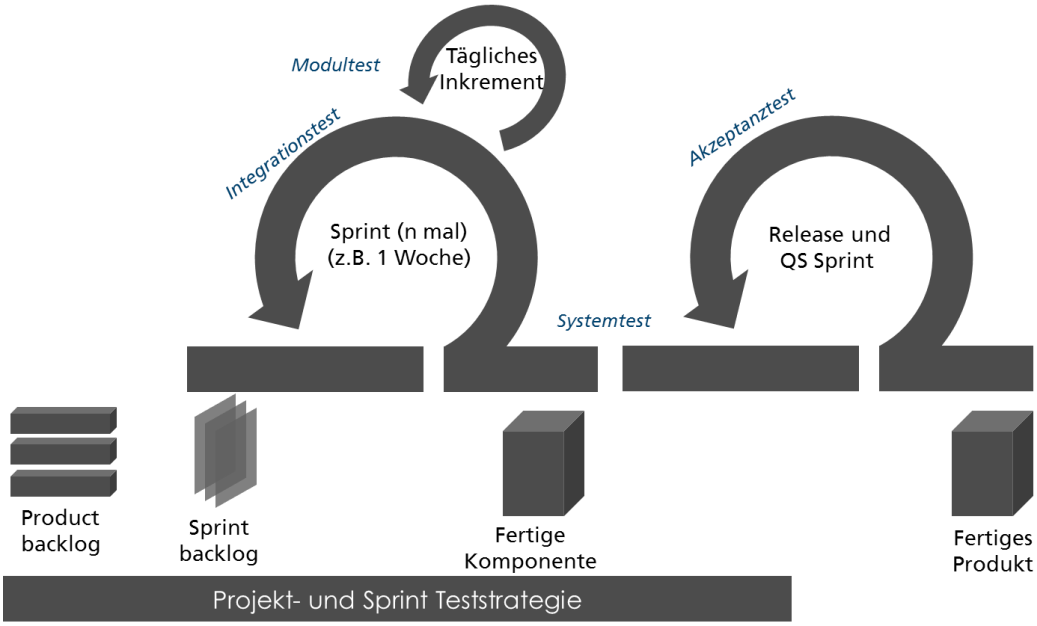

Abbildung 1: Teststufen aufeinander abstimmen

Input

- Anforderungen an den Test wie Testziele, verfügbare Testressourcen, Testprozess
- Entwicklungsprozess

Output

- Abgestimmte und koordinierte Testplanung

Rahmenbedingungen

Ausführender
Testmanager

Werkzeuge, Hilfsmittel
-

Vorkenntnisse/Erfahrungen
Kenntnisse über das Testen und die Entwicklung

Ort/Umgebung
Büro

Weitere Teilnehmer
Projektleiter, Product Owner

Voraussichtliche Dauer
Je nach Umfang des Testprozesses

Vorgehensweise

Vorbereitung
Eine Auswahl und Verfeinerung der Teststufen sollte durchgeführt werden (z.B. kann ein Integrationstest sich aufspalten in einen Modul- und einen Systemintegrationstest), ebenso sollten die Anforderungen an das zu testende System klar sein. Darüber hinaus muss der konkrete Entwicklungsprozess bekannt sein, um die Strategie sowie die konkret durchzuführenden Testaktivitäten auf den jeweiligen Teststufen zuordnen zu können.

Durchführung
Zunächst geht es darum, die Teststrategie zu definieren. Ein wesentlicher Aspekt dabei ist eine Festlegung, wie intensiv auf welcher Stufe getestet werden soll. Das hängt von

unterschiedlichen Faktoren ab, beispielsweise dem verfügbaren Budget für das Testen, dem Risiko, der Komplexität des zu entwickelnden Systems, der Menge der Teams oder auch dem Wissen über Testaktivitäten. Der Testmanager muss möglichst viele dieser Faktoren ermitteln und berücksichtigen, oftmals in enger Abstimmung mit weiteren Rollen wie einem Projektleiter oder dem Product Owner. Eine Strategie kann sein, dass auf der Modulebene jedes Modul mit TDD entwickelt werden muss, bevor es integriert wird und den Integrationstest durchläuft und auf Systemtestebene dann alle User Stories manuell durchlaufen werden müssen. Auf jeder Teststufe kann die Strategie auch weiter verfeinert werden. Beispielsweise kann auf Systemtestebene jede als kritisch eingestufte Userstory von mindestens zwei Testern sowohl erfahrungsbasiert als auch systematisch getestet werden, während für weniger kritische Userstories nur ein Tester Tests durchführt.
Der Entwicklertest auf Unit-Ebene kann sowohl auf White Box als auch auf Black Box-Basis stattfinden und sollte von den Entwicklern durchgeführt werden. Dabei geht es nicht nur um die korrekte Funktionalität bei gültigen Eingaben, sondern auch um das korrekte Abfangen ungültiger Eingabewerte. Jede „kleinste Einheit" („Unit") wird möglichst isoliert getestet und geprüft. Bei agilen Entwicklungslebenszyklen hat sich der Einsatz von Test-First oder Test-Driven-Development etabliert. Im Integrationstest wiederum werden die Schnittstellen zwischen Einheiten geprüft und somit das Zusammenspiel mehrerer Einheiten getestet. Auch wenn jede Einheit für sich korrekt arbeitet, kann das Zusammenspiel trotz allem zu fehlerhaften Ergebnissen führen. Der Integrationstest kann in mehrere Phasen aufgeteilt werden, um von „kleinen" Integrationsschritten zu „großen" Integrationsschritten alles abzudecken. Auf dieser Teststufe spielt insbesondere bei iterativer Entwicklung der Regressionstest eine große Rolle. Mit der Prüfung nichtfunktionaler Aspekte wie Performanz oder Security kann bereits im Integrationstest begonnen werden. Der Systemtest wiederum wird auf Black Box-Basis durchgeführt und umfasst die Prüfung des kompletten Systems, in der Regel von der Benutzeroberfläche aus. Geprüft werden soll, ob auf Basis der Eingaben die Ziele der Benutzer mit der Anwendung erreicht werden können. Auf dieser Teststufe finden auch nichtfunktionale Tests statt. Alle diese Teststufen finden in der Umgebung der Entwicklung statt. Erst der Akzeptanztest findet in der Umgebung der künftigen Anwender statt und wiederholt Testfälle, die im Systemtest bereits durchlaufen wurden. Abbildung 1 zeigt ein Beispiel eines agilen Vorgehens und nennt die Teststufen – der Modultest wird hier täglich durchgeführt, der Integrationstest am Ende des einwöchigen Sprints. Systemtests werden nach einigen Sprints begonnen und erste Testfälle durchlaufen. Der letzte Sprint vor dem Release wird dann noch einmal explizit für die Qualitätssicherung genutzt und Akzeptanztests werden durchgeführt.

Nachbereitung/Auswertung
Fortlaufend wird geprüft, ob die initiale Teststrategie sinnvoll ist und ggf. angepasst. Stellt sich zum Beispiel während der Entwicklung heraus, dass eine bisher nicht als risikoreich eingeschätzte Komponente risikoreich ist, sollten Testaufwände hier zusätzlich konzentriert werden, beispielsweise durch weitere Modul- oder Integrationstests. Fehler, die während der Tests auf den unterschiedlichen Stufen gefunden werden, sollten dokumentiert werden (z.B. in einem Bugtracking Werkzeug) und analysiert werden, um zukünftige Qualitätssicherung weiter zu optimieren und ggf. zu entscheiden, ob Fehler schon früher gefunden werden können bzw. gänzlich vermieden werden können.

Gütekriterien/Empfehlungen

Jede der Teststufen hat eine spezifische Testbasis, die für die Testableitung genutzt werden sollte. Die Testziele sind zu beachten, so dass nicht mehrere Teststufen das Gleiche testen. Die Abstimmung der Teststufen sollte vor der Testableitung stattfinden. Das Festlegen der Teststrategie muss in enger Abstimmung mit Stakeholdern wie einem Projektleiter oder

Product Owner erfolgen; zudem muss der agile Prozess berücksichtigt werden, so dass die Qualitätssicherung zielgerichtet geplant werden kann.

Risiken

Oftmals werden Aspekte, die bereits in anderen Teststufen abgedeckt wurden, nochmals getestet, was zu unnötigem Mehraufwand führt, was insbesondere bei mehreren unabhängigen Teams passieren kann. Werden bestimmte Aspekte dagegen nicht auf der dafür vorgesehenen Teststufe geprüft, sondern erst später, kann dies durch die Kombinatorik im Zusammenhang mit dem Zusammenspiel mehrerer Komponenten zu einen erheblichen Mehraufwand führen. Wird die Abstimmung der Teststufen nach der systematischen Testfallableitung vorgenommen, so müssen Teile dieser Testfallableitung sehr wahrscheinlich wiederholt werden. Zudem gilt das grundsätzliche Risiko, dass das Entwickeln neuer Funktionalität höher priorisiert wird, als die Durchführung einer ordentlichen Qualitätssicherung, wie sie in der Teststrategie festgehalten wurde.

Einordnung in den agilen Referenzprozess

Mögliche Vorgänger

- Architektur\Architekturlösungen im Team entwickeln

Mögliche Nachfolger

- Testen\Erfahrungsbasiertes Testen durchführen
- Testen\Fehlermanagement einsetzen
- Testen\Reviews von Entwicklungsartefakten durchführen
- Testen\Systematische Testfallableitung und Tests durchführen

Mögliche Alternativen, verwandte Praktiken

- Unterstützung\Standardisierte Dokumentstrukturen verwenden
- Architektur\Feature-Entwicklung und interne Qualitätsaufgaben verzahnen

Einordnung in das PQ4Agile-Qualitätsmodell

Prozessanwendungsqualität: Effektivität, Effizienz.

Schlagworte

Abstimmung, Koordination, Planung

Weiterführende Informationen

Literatur

A. Spillner, T. Linz: Basiswissen Softwaretest: Aus- und Weiterbildung zum Certified Tester – Foundation Level nach ISTQB-Standard, dpunkt.verlag GmbH, 2012.
A. Spillner, T. Roßner, M. Winter, T. Linz: Praxiswissen Softwaretest - Testmanagement: Aus- und Weiterbildung zum Certified Tester - Advanced Level nach ISTQB-Standard, dpunkt.verlag GmbH, 2014.
I. Burnstein: Practical Software Testing. A Process-Oriented Approach. Springer-Verlag, 2003.

Das Projekt PQ4Agile wird vom Bundesministerium für Bildung und Forschung im Rahmen der Maßnahme KMU-innovativ: IKT (01|S13032) gefördert.