Gesellschaft für Informatik e.V. (GI) GI. (Hrsg.): INFORMATIK 2021,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 1839

QA-Strategie oder Flickenteppich?

Herausforderungen an Teststrategien für moderne Softwaresysteme

Ralf Somplatzki[1]

Abstract: Moderne Software-Projektstrukturen stellen veränderte und neue Anforderungen an das Testvorgehen. Aufgaben aus den Bereichen Entwicklung und Softwaretest nähern sich symbiotisch an. Die klassischen Rollenbilder verschwimmen. Was ermöglicht agilen Teams, dem Spagat der Verantwortung für das funktionale Ergebnis, wie auch der erwarteten Qualität gerecht zu werden? Wie kann Qualität in hybriden Systemlandschaften gewährleistet werden? Der Beitrag sucht Antworten auf diese und weitere Fragen, mit denen sich heutige Software-Dienstleister auseinandersetzen. Es wird ein möglicher Ansatz aufgezeigt, um mit der Vielgestalt der Projekte, geprägt durch Umfang, Teamstrukturen und Kundenkulturen, umzugehen.

Keywords: Qualitätssicherung, Softwaretest, Teststrategie, Testmanagement, Vorgehensmodell, Projektstruktur, Agile Softwareentwicklung, SCRUM.

1 Motivation

Die Qualitätssicherung (QA) moderner Softwareprojekte ist über die Evolutionsstufen der Softwareentwicklung in den vergangenen Jahren erheblich anspruchsvoller geworden. Verteilte Systeme, agile Methoden und kultureller Wandel fordern neue Konzepte zur Qualitätssicherung, die über reinen Softwaretest hinaus gehen. Benötigt wird eine Strategie, die der Mannigfaltigkeit moderner Softwareprojekte Rechnung trägt und dabei flexibel und leichtgewichtig bleibt.

2 Ausgangslage

Software-Dienstleister stehen vor der Herausforderung, unterschiedliche Projekt-Setups ebenso wie Kultur und Prioritäten der Kunden, in ihrer Vielschichtigkeit im eigenen Unternehmen zu vereinen. Gleichzeitig ist es aus Gründen der Effizienz und der Pflege einer eigenen Unternehmenskultur und -identität erforderlich, einheitliche Vorgehensweisen zu etablieren.

[1] Ralf Somplatzki/GEBIT Solutions GmbH, Oststraße 10, D-40211 Düsseldorf, ralf.somplatzki@gebit.de

3 Zielsetzung

Eine moderne QA-Strategie für Software-Dienstleister muss unterschiedliche Projektausprägungen unterstützen. Das Spektrum reicht von überschaubaren Dienstleistungsaufträgen mit einem kleinen Team über große, langlaufende Projekte mit verteilten Teams bis hin zu komplexen Lösungsszenarien in Kollaboration zahlreicher Dienstleister und Lieferanten.

An erster Stelle steht nicht, einzelne Tätigkeiten zu optimieren oder Tools einzuführen, sondern geeignete Methoden und Strategien zu finden, die für heutige Projektsituationen geeignet sind.

4 Herausforderungen

Dieser Beitrag adressiert Problemfelder im Kontext der Qualitätssicherung, die im Zusammenhang mit Entwicklungsmethoden und modernen Softwaresystemen auftreten.

Als Lösungsansatz für die heutige Vielschichtigkeit wird mehr als die eine, „richtige" Teststrategie benötigt. Es ist ein adaptierbarer Ansatz erforderlich, der es ermöglicht, unter einem „Unternehmensdach" vielgestaltig und dennoch kompatibel zu arbeiten. Dabei sind insbesondere die folgenden Aspekte zu berücksichtigen:

4.1 Unterstützung der Eigenständigkeit der Teams

Agile Konzepte setzen darauf, dass Teams in Eigenverantwortung für ein Ergebnis „von Wert" (für die Anforderer) stehen. In diesem Kontext ist die Einbeziehung des Teams in die Prozessgestaltung ein unverzichtbares Konzept, auch in Bezug auf die Qualitätssicherung.

4.2 Beachtung der Ziele agiler Strategien, insbesondere „Time-To-Market"

Unter anderem bedingt durch hohen Wettbewerbsdruck der beauftragenden Unternehmen erfreuen sich agile Ansätze einer großen Beliebtheit. Mit „agil" wird dabei assoziiert, gewünschte Lieferungen zu einem bestmöglichen, nahen Termin erwarten zu dürfen.

Um dies in der Praxis tatsächlich zu ermöglichen, ist zu beachten, dass eine zeitnahe Lieferung nur dann sinnvoll ist, wenn das Produkt „von Wert" ist. Also die gestellten fachlichen Anforderungen erfüllt. [SS20]

Hier setzt die Herausforderung für die Qualitätssicherung deutlich früher an, als zum Zeitpunkt der Softwareentwicklung. Allzu häufig stellt sich mit der Lieferung heraus, dass Erwartungshaltungen und Lieferqualität oder -umfang voneinander abweichen.

4.3 Wandel von Rollenbildern, Skills und Verantwortung

Gegenüber vertikalen Verfahren, wie dem Wasserfallmodell, werden zahlreiche Disziplinen im Rahmen eines Softwareprojektes in einem Team vereint, das aus gleichberechtigten Teilnehmern zusammengesetzt ist. Dies führt dazu, dass auch das Thema QA im Verantwortungsbereich des Teams liegt. Die Aufgabenfelder Softwareentwicklung und Softwaretest wachsen zusammen und stellen neue Anforderungen an die Teammitglieder.

5 Lösungsansatz und Ergebnisse

Für eine vielgestaltige Projektlandschaft werden modulare Methodeneinheiten bereitgestellt. Zwei Schlüssel-Elemente versetzen Teams und Projektverantwortliche in die Lage, dem jeweiligen Projektsetup gerecht zu werden:

- QA-Module (Methoden-Beschreibungen, Vorlagen und Trainingseinheiten)
- Projekt-Schablonen

Die QA-Module beschreiben, die für die Projekte eines Unternehmens in Frage kommenden Vorgehensweisen und Techniken. Projekt-Schablonen stellen beispielhafte Projektmodelle dar, unter Einbeziehung der vorhandenen QA-Module. Im Idealfall kann ein Projektteam eine Projekt-Schablone auf ein eigenes Projekt anwenden und dabei individuelle Anpassungen an den eingesetzten QA-Modulen vornehmen.

Darüber hinaus werden messbare Kriterien benötigt, um Qualität bewerten zu können [Li09]. Sie lassen sich zum Beispiel aus den Testmanagement-Aspekten des ISTQB herleiten. Entscheidend ist hier, dass alle Projektbeteiligten die Messkriterien verstehen, um eine gemeinsame Einigung herstellen und tragen zu können.

Nach diesem Konzept werden einzelnen Etappen und Meilensteinen des Entwicklungsprozesses Qualitäts-Messpunkte zugeordnet. Abb. 1 zeigt ein minimalistisches Beispiel:

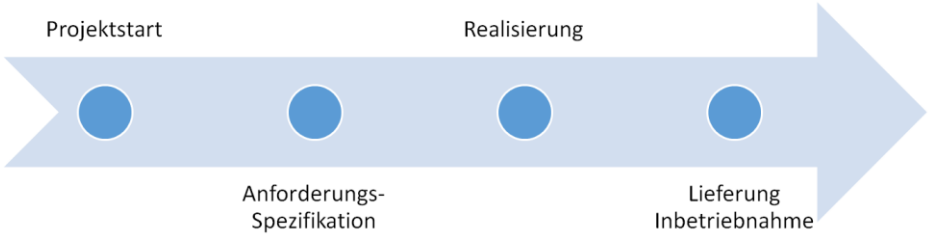

Abb. 1: Qualitäts-Messpunkte im Projekt-Lebenszyklus

Zur Veranschaulichung eine Aufstellung möglicher Messpunkte und Messkriterien. Diese können 1-zu-1 adaptiert oder je nach Kundenanforderungen und weiteren Rahmenbedingungen angepasst werden.

Messpunkt	Messparameter
Projektstart	Bestimmung der Wichtung der Qualitätsparameter: Zeit, Funktionalität, Qualität, Kosten
Anforderungen (DoR)	Erforderlicher Detailgrad, Verständlichkeit, Abnahme durch welche Rollen?
Tagesabschluss (Build)	Fehlertoleranz (z.B. Wertung von Warnungen)
Sprintergebnis (DoD)	Umfang erforderlicher Artefakte, wie kommentierter Sourcecode, Spezifikation, Testfallbeschreibungen, Review-Abnahme, Testabdeckung
Produktionskandidat	Funktionale Testergebnisse, Testabdeckung nach Teststufen, kommentierte Testberichte

Tab. 1: Beispiele für Messpunkte und Messparameter

Für alle Messpunkte werden Vorlagen zur Verfügung gestellt, die an die projektspezifischen Bedürfnisse angepasst werden. Auf diese Weise werden Erwartungshaltungen schon vor Produktionsbeginn abgestimmt und können in weiteren Evolutionsstufen, zum Beispiel bei jeder Retrospektive überprüft und angepasst werden [Be01]

Abb. 2: Kontinuierlicher Abgleich der Qualitätskriterien im Projekt-Lebenszyklus

Die folgenden Absätze zeigen auf, wie das Lösungskonzept den Herausforderungen, aus Kapitel 4, Rechnung trägt:

5.1 Unterstützung der Eigenständigkeit der Teams

QA-Module unterstützen Teams dabei, im Hinblick auf Umfang und Nutzen von QA-relevanten Aspekten eigenverantwortlich zu agieren. Als Beispiele seien „DoR“ („Definition of Ready“) und „DoD“ („Definition of Done“) genannt. Während die „DoR“ Qualitätskriterien für Anforderungen festlegt, bestimmt die „DoD“, wann eine Aufgabe (ein Teilprodukt) abgeschlossen/fertiggestellt ist. Darüber hinaus bieten QA-Module

Vorlagen für Testverfahren, Techniken und Werkzeuge, über deren Einsatz die Teammitglieder autark entscheiden können.

5.2 Beachtung der Ziele agiler Strategien, insbesondere „Time-To-Market“

Im Rahmen einer tragfähigen QA-Strategie einigen sich die Projekt-Beteiligten über die Kriterien zur Definition „von Wert“. Zum Beispiel ist für einen Kunden, der ein neues Softwareprodukt entwickeln lässt, „von Wert“, dass sein Produkt zum Termin einer Messe vorführbar ist ohne vollständig zu sein. Für einen Kunden, der eine Softwarelösung für den produktiven Einsatz benötigt, ist das Produkt „von Wert“, wenn es stabil und zuverlässig die erforderlichen Standard-Geschäftsvorfälle unterstützt.

QA-Module ermöglichen Teams, im Hinblick auf die abgestimmte Erwartungshaltung, geeignete Elemente zur Qualitätssicherung auszuwählen. In Bezug auf das Beispiel „Messemodell“ kann ein minimalistisches Automatisierungsverfahren und für die Vorführszenarien ein explorativer Testansatz gewählt werden.

5.3 Wandel von Rollenbildern, Skills und Verantwortung

Um die Eigenständigkeit der Teams zu unterstützen hat sich die Integration von Teammitgliedern als sinnvoll erwiesen, die im Bereich Softwaretest ausgebildet und erfahren sind. Dies führt dazu, dass in einem ausgewogenen Team-Setup nicht nur Wissen über konkretes Testvorgehen, sondern auch das Verständnis („Mindset“) in Bezug auf Qualität, und damit die Kultur, entwickelt wird.

Die QA Module helfen den Teams, die Vielfältigkeit in der SW QA zu überblicken. QA-erfahrene Teammitglieder sind nicht nur auf SW-Test fokussiert, sondern unterstützen die Teammitglieder in ihrer QA-Entwicklung.

6 Lösungsbeispiel

Zur Veranschaulichung des Lösungsansatzes werden drei Projektvarianten angenommen[2]: Ein präsentationsfähiges Pilotsystems, das lediglich Funktionen bereitstellt, die für eine weitergehende Entscheidung relevant sind, ein Softwareprojekt, das eine kundenspezifische Lösung implementiert und ein Standardprodukt, das Drittanbietern („Consumer“) den Zugriff auf bereitgestellte Funktionen ermöglicht.

Zu Beginn des Projektes werden die Qualitätsaspekte priorisiert (Abb.3).

[2] Für das Beispiel wird ein Prozess in Anlehnung an das SCRUM-Modell angenommen. Das zugrunde liegende Konzept der Adaptierbarkeit lässt sich aber auch auf andere Vorgehensweisen anwenden.

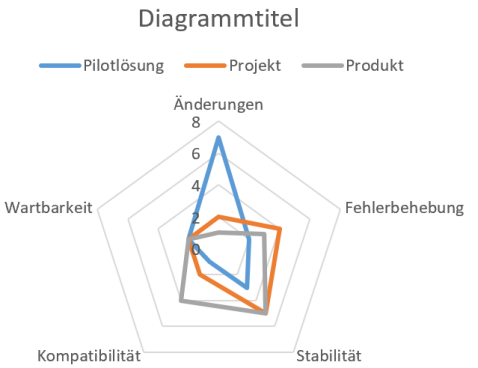

Abb. 3: Gewichtung von Qualitätskriterien in Relation zur Projektart

Anhand der jeweiligen strategischen Ausrichtung werden Projektschablone und QA-Module gewählt und an den Projektbedarf angepasst. Teams sind somit in der Lage, sowohl Unternehmens-Standards einzusetzen und dennoch das eigene Projektsetup zu gestalten. Zum Beispiel für die Pilotversion auf eine offene „DoR“, für das Kundenprojekt auf die Mindestanforderungen an Spezifikationsartefakte und für die Service-Komponente auf eine weitere Teststufe „Contract“ [Cl14] Wert zu legen.

7 Fazit und Ausblick

Die Lösung versteht sich als Basis, als „Methoden-Bibliothek“, die beliebig erweitert oder zugeschnitten werden kann. In diesem Sinne ermöglicht die Methodik allen Teammitgliedern, an der Gestaltung und Nutzung der Bibliotheks-Elemente mitzuwirken.

In Anlehnung an den Titel dieses Beitrages liegt es in der Hand der Beteiligten, kein Flickenwerk, sondern eine abgestimmte QA Strategie zu gestalten, die auf das jeweilige Projekt strategisch zugeschnitten ist.

8 Literaturverzeichnis

[Be01] Beck, K. et.al.: Principles behind the Agile Manifesto; Principle #12; https://agilemanifesto.org/principles.html; Stand: 16.06.2021

[Cl14] Toby Clemson, Testing Strategies in a Microservice Architecture, https://martinfowler.com/articles/microservice-testing/. Stand: 15.06.2021

[Li09] Liggesmeyer P. (2009) Software-Messung. In: Software-Qualität. Spektrum Akademischer Verlag. https://doi.org/10.1007/978-3-8274-2203-3_7. Stand: 13.06.2021

[SS20] Schwaber, K.; Sutherland, J.: The Scrum Guide, S. 3&11, 2020