Jama software®

WHITEPAPER

Anforderungen testen

Aus Karl E. Wiegers, Testing the Requirements, Microsoft Press, 2006.

Jemand fragte mich einmal, wann man mit dem Testen von Software beginnen sollte. Ich antwortete: „Sobald Sie Ihre erste Anforderung geschrieben haben."

Nur mithilfe des Pflichtenhefts lässt sich kaum veranschaulichen, wie ein System tatsächlich arbeiten wird. Anforderungsbasierte Tests helfen, das erwartete Systemverhalten für die Projektbeteiligten greifbarer zu machen. Während Sie die Tests gestalten, offenbaren sich bereits viele Probleme mit den Anforderungen – lange bevor Sie sie auf einem Produktivsystem durchführen. Wenn Sie also mit der Testentwicklung anfangen, sobald die ersten Teile der Anforderungen feststehen, können Sie Probleme schnell und kostengünstig beheben.

Anforderungen und Tests

Tests und Anforderungen lassen sich synergetisch nutzen. Sie stehen für ergänzende Ansichten eines Systems. Die unterschiedlichen Perspektiven – in Form schriftlicher Anforderungen, von Diagrammen, Tests, Prototypen usw. – geben Ihnen ein besseres Verständnis des Systems.

Agile Entwicklungsmethoden für Software haben oft einen Fokus auf Tests zur Nutzerfreigabe anstelle detaillierter funktionaler Anforderungen. Ein Systemverständnis aus der Testperspektive ist wertvoll, stellt aber nur einen Teil des Wissens um Anforderungen dar.

Wenn Sie (funktionale) Blackbox-Tests entwickeln, bekommen Sie eine deutliche Vorstellung davon, wie sich das System unter bestimmten Bedingungen verhalten soll. Vage und unklar formulierte Anforderungen werden Ihnen sofort auffallen, da Sie die erwartete Systemreaktion nicht beschreiben können. Und wenn Business-Analysten, Entwickler und Kunden die Tests zusammen durchlaufen, kommen sie bei einer gemeinsamen Vorstellung davon an, wie das Produkt funktionieren soll.

Eine persönliche Erfahrung hat mir gezeigt, wie wichtig es ist, die Anforderungsspezifikation gedanklich mit dem Testen zu verbinden. Charlie ist unser UNIX-Skriptguru. Ich habe ihn mal gebeten, eine einfache E-Mailschnittstelle als Ergänzung für ein kommerzielles System zur Fehlerverfolgung zu entwickeln. Dafür habe ich ein Dutzend funktionale Anforderungen erstellt, um zu beschreiben, wie die E-Mailschnittstelle funktionieren soll. Charlie war begeistert. Er hatte schon viele Skripte geschrieben, doch Anforderungen in Schriftform kannte er nicht. Unglücklicherweise hatte ich ein paar Wochen gewartet, bevor ich die Tests für diese E-Mailfunktion schrieb. Natürlich hatte sich in einer der Anforderungen ein Fehler eingeschlichen. Ich fand ihn, weil das Bild, das ich mir von der Arbeitsweise der Funktion gemacht hatte und in etwa zwanzig Tests dargestellt wurde, mit einer der Anforderungen nicht übereinstimmte. Die fehlerhafte Anforderung hatte ich korrigiert,

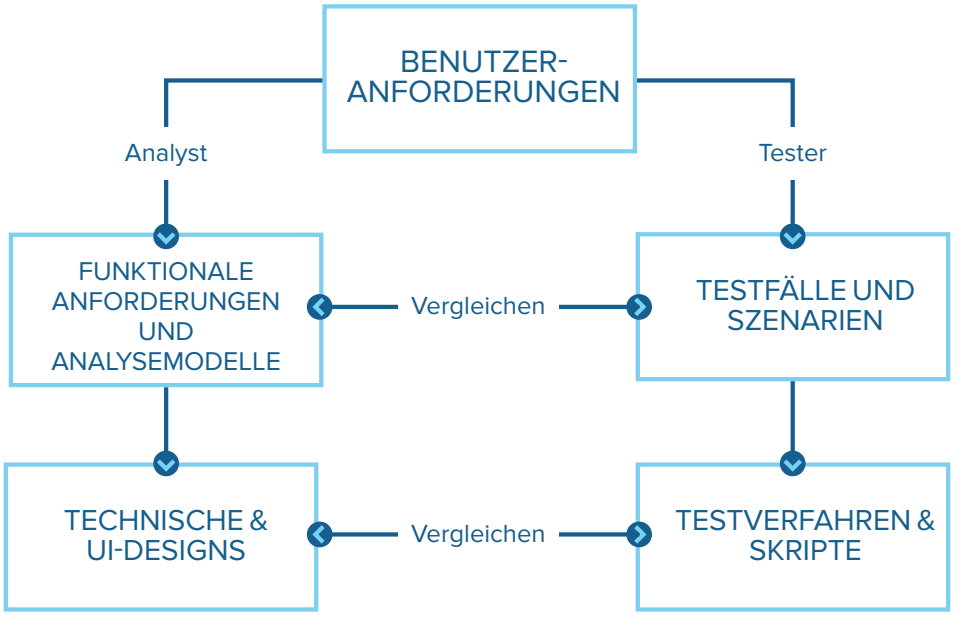

Abbildung 1. Entwicklung und Test aus einer Hand

noch bevor Charlie seine Implementierung abschloss. Und als er das Skript geliefert hat, war es fehlerfrei. Ein kleiner Sieg – aber kleine Siege summieren sich.

Konzeptionelle Tests

Solange Sie noch keine ausführbare Software geschrieben haben, sich also noch in der Anforderungsphase befinden, können Sie Ihr System natürlich nicht testen. Sie können aber bereits mit der Ableitung konzeptioneller Tests aus Anwendungsfällen oder User Stories beginnen. Mit den Tests können Sie dann funktionale Anforderungen, Analysemodelle und Prototypen bewerten. Die Tests sollten den normalen Ablauf des Anwendungsfalles abdecken, genauso wie alternative Abläufe und die Ausnahmen, die Sie bestimmt haben, während Sie die Anforderungen ermittelten und analysierten.

Veranschaulichen wir uns das an einem Beispiel: Stellen wir uns ein Tracking-System für Chemikalien vor. In diesem System gibt es den Anwendungsfall „Bestellung ansehen“. Der Nutzer ruft eine Bestellung aus der Datenbank auf und lässt sich die Details dazu anzeigen. Einige der konzeptionellen Tests unter diesem System wären die folgenden:

- Der Nutzer gibt die Bestellnummer zur Ansicht ein, die Bestellung ist vorhanden, der Nutzer gab sie auf. Erwartetes Ergebnis: Anzeige der Bestelldetails.
- Der Nutzer gibt die Bestellnummer zur Ansicht ein, die Bestellung liegt nicht vor. Erwartetes Ergebnis: Anzeige der Nachricht „Entschuldigen Sie, ich kann Ihre Bestellung nicht finden.“
- Der Nutzer gibt die Bestellnummer ein, die Bestellung ist vorhanden, der Nutzer gab sie nicht auf. Erwartetes Ergebnis: Anzeige der Nachricht „Entschuldigen Sie, das ist nicht Ihre Bestellung, Sie können sie daher nicht einsehen.“

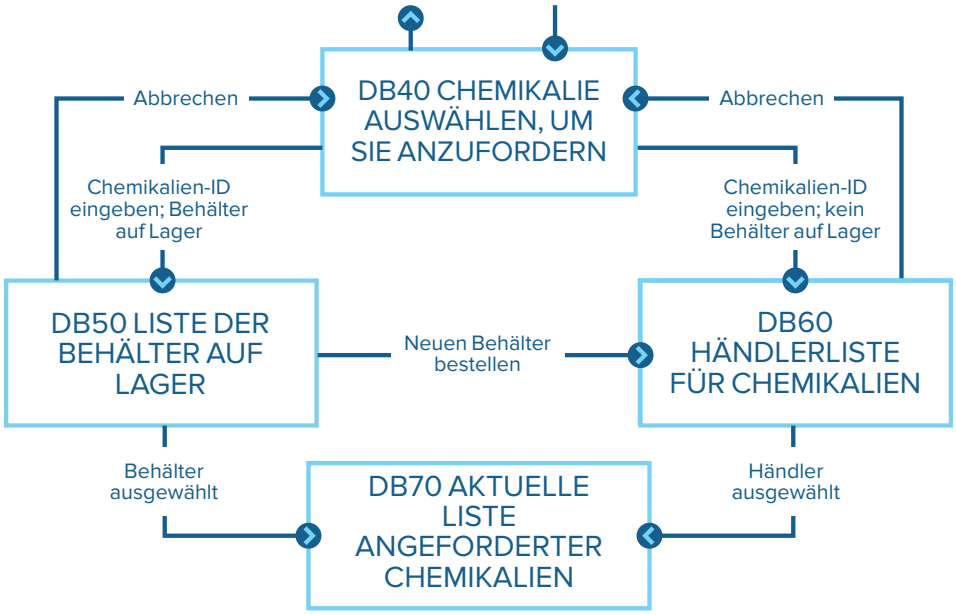

Abbildung 2. Ausschnitt der Dialoglandkarte für den Anwendungsfall „Chemikalie anfordern."

Ilm Idealfall schreibt ein Business-Analyst die funktionalen Anforderungen und ein Tester beginnt mit den Tests ab einem gemeinsamen Ausgangspunkt, den Benutzeranforderungen, wie in Abbildung 1 dargestellt. Unklare Benutzeranforderungen und unterschiedliche Interpretationen führen zu abweichenden Ansichten, die von den funktionalen Anforderungen, Modellen und Tests dargestellt werden. Finden Sie diese Abweichungen, decken Sie die Fehler auf. Während die Entwickler die Anforderungen schrittweise in die Benutzeroberfläche und das technische Design übertragen, können Tester auf die frühen konzeptionellen Tests zurückgreifen und in detaillierte Verfahren verwandeln.

Ein Beispiel

Um das Thema Anforderungstests konkreter zu machen, schauen wir uns an, wie bei einem Anwenderbeispiel des Tracking-Systems für Chemikalien, das Pflichtenheft, die Analyse-Modellierung und frühe Testfall-Generierung verbunden wurden. Im Folgenden stehen ein Anwendungsfall, funktionale Anforderungen, ein Teil einer Dialoglandkarte und ein Test. Alle Elemente beziehen sich auf die Aufgabe, eine Chemikalie zu bestellen.

Der Anwendungsfall

Ein grundsätzlicher Anwendungsfall für dieses System lautet „Chemikalie anfordern". Dieser Anwendungsfall umfasst einen Pfad, der dem Benutzer ermöglicht, einen chemischen Behälter anzufordern, der im Chemielager bereits vorhanden ist. Diese Möglichkeit würde dem Unternehmen Kosten ersparen: Statt einen neuen Behälter kaufen zu müssen, wird ein vorhandener wiederverwendet. Hier ist die Beschreibung des Anwendungsfalls:

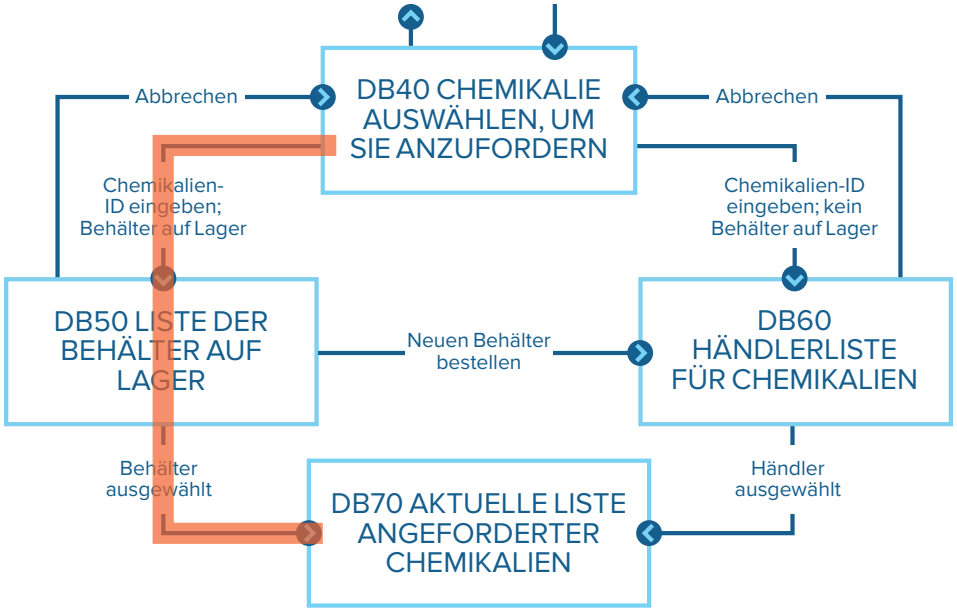

Abbildung 3. Nachverfolgung eines Tests auf der Dialoglandkarte für den Anwendungsfall „Chemikalie anfordern."

Der Besteller bestimmt die gewünschte Chemikalie, indem er ihren Namen oder ihre ID-Nummer eingibt. Das System lässt den Besteller daraufhin zwischen einem neuen oder benutzten Behälter aus dem Lagerraum oder einem neuen Behälter eines Händlers wählen.

Die funktionale Anforderung

Hier ein paar Funktionen, die mit dem Anwendungsfall verbunden sind:

1. Wenn im Lagerraum Behälter für die bestellte Chemikalie vorhanden sind, soll das System eine Liste der verfügbaren Behälter anzeigen.
2. Der Benutzer kann so einen der angezeigten Behälter wählen oder anfragen, einen neuen bei einem Händler zu bestellen.

Die Dialoglandkarte

Eine Dialoglandkarte liefert die Übersicht über die Architektur einer Benutzerschnittstelle in Form eines Zustandsübergangsdiagramms. Die obenstehende Abbildung 2 zeigt einen Teil der Dialoglandkarte für den Anwendungsfall „Chemikalie anfordern", der sich auf diese Funktion bezieht. Die Felder stellen Anzeigen der Benutzeroberfläche dar (in diesem Fall Dialogfenster), die Pfeile Navigationspfade von einer Anzeige zur anderen.

Test

Für diesen Anwendungsfall gibt es mehrere Ausführungen, zahlreiche Tests zu normalem Ablauf, Alternativen und Ausnahmen sind denkbar.

Der folgende Test basiert auf dem Pfad, der zur Anzeige der verfügbaren Behälter im Chemikalienlager führt:

Geben Sie im Dialogfenster DB40 eine gültige Chemikalien-ID ein; im Chemikalienlager sind zwei Behälter mit dieser Chemikalie. Das Dialogfenster DB50 erscheint, mit den beiden Behältern. Wählen Sie den zweiten Container. DB50 schließt sich und Behälter Nummer zwei wird am unteren Rand der Liste aktuell angeforderter Chemikalien im Dialogfeld DB70 hinzugefügt.

Die Testleitung für das Tracking-System, Ramesh, hat mehrere Tests wie diesen geschrieben. Und alle basieren auf seinem Verständnis davon, wie der Benutzer mit dem System interagieren könnte, um eine Chemikalie anzufordern. Solch abstrakte Tests lassen sich unabhängig von Implementierungsdetails durchführen. Sie beziehen keine bestimmten Interaktionen ein, also auch nicht das Anklicken von Schaltflächen oder dergleichen. Während die Benutzeroberfläche weiterentwickelt wurde, hat Ramesh diese abstrakten Tests in spezifische Testabläufe verfeinert.

Wenn Sie den Ausführungspfad für jeden Test im Modell verfolgen, können Sie falsche oder fehlende Anforderungen finden, Fehler in der Dialoglandkarte korrigieren und die Tests verfeinern. Um diese Anforderungen zu testen, glich Ramesh sie zunächst mit den funktionalen Anforderungen ab. Anschließend hat er geprüft, ob jeder Test mit dem bestehenden Anforderungskatalog durchgeführt werden konnte. Zudem stellte er sicher, dass zumindest ein Test alle funktionalen Anforderungen abdeckte. Eine solche Zuordnung offenbart in der Regel Versäumnisse. Anschließend verfolgte Ramesh den Ausführungspfad für jeden Test auf der Dialoglandkarte mit einem Textmarker. Die rote Linie in Abbildung 3 (vgl. Seite 8) zeigt, wie der vorangegangene Stichprobentest auf die Dialoglandkarte übergeht. Wenn Sie den Ausführungspfad für jeden Test im Modell verfolgen, können Sie falsche oder fehlende Anforderungen finden, Fehler in der Dialoglandkarte korrigieren und die Tests verfeinern. Stellen wir uns vor, dass alle Tests auf diese Weise ausgeführt wurden. Und die Navigationszeile, die von DB50 bis DB60 reicht, mit der Bezeichnung „Neuen Behälter bestellen“ in Abbildung 2 (dargestellt auf Seite 6) wurde nicht markiert. Dafür gibt es zwei mögliche Interpretationen:

- Die Navigation von DB50 zu DB60 ist kein zulässiges Systemverhalten. Der Business-Analyst muss diese Zeile aus der Dialoglandkarte entfernen. Wenn das Pflichtenheft für die Software eine Anforderung enthält, die den Übergang festlegt, muss der Business-Analyst diese Anforderung ebenfalls entfernen.
- Die Navigation ist ein zulässiges Systemverhalten, aber es fehlt der Test, der das Verhalten aufzeigt.

Wenn ich eine solche Unterbrechung finde, kenne ich noch nicht die richtige Interpretation. Ich weiß jedoch, dass alle

Ansichten der Anforderungen – Pflichtenheft für die Software, Modelle und Test – übereinstimmen müssen.

Angenommen, ein anderer Test besagt, dass der Benutzer Schritte unternehmen kann, um direkt von DB40 auf DB70 zu wechseln. Die Dialogkarte enthält jedoch keine Navigationszeile dafür, sodass der Test nicht „ausgeführt“ werden kann. Auch hier gibt es zwei mögliche Interpretationen:

- Die Navigation von DB40 zu DB70 ist kein zulässiges Systemverhalten, also ist der Test falsch.
- Die Navigation von DB40 zu DB70 ist eine zulässige Funktion, aber in der Dialogkarte und vielleicht auch im Pflichtenheft für die Software fehlt die Anforderung, die den Test ermöglicht.

In diesen Beispielen brachten Business-Analyst und Tester Anforderungen, Analysemodelle und Tests zusammen, um fehlende, fehlerhafte oder unnötige Anforderungen zu erkennen, bevor Code geschrieben wurde. Wenn ich diese Technik anwende, finde ich immer Fehler an allen Punkten, die ich miteinander vergleiche.

Berater Ross Collard sagt dazu: „Anwendungsfälle und Tests funktionieren auf zweierlei Weise gut zusammen: Wenn die Anwendungsfälle für ein System vollständig, genau und klar sind, ist die Ableitung der Tests unkompliziert. Wenn etwas an den Anwendungsfällen nicht stimmt, helfen Tests, sie zu bereinigen.“ Er hat meine volle Zustimmung. Das konzeptionelle Testen von Software-Anforderungen ist eine wirkungsvolle Technik. Unklarheiten und Fehler der Anforderungen werden damit schon früh im Projekt erkennbar. Kosten und Zeitplan sind so stets unter Kontrolle.

ÜBER KARL WIEGERS

Karl Wiegers ist ein Experte in Software-Entwicklung und hat darin jahrzehntelange Erfahrung. Er ist weltweit als Trainer und Berater für Software-Entwicklung, -Verwaltung und Prozessoptimierung tätig. Darüber hinaus hat Karl Wiegers fünf Bücher zu IT-Themen wie Software-Anforderungen und mehr als 175 Artikel verfasst. 1997 gründete er die Beraterfirma Process Impact. Davor war er 18 Jahre für Eastman Kodak in verschiedenen Positionen tätig gewesen, unter anderem als Forschungswissenschaftler für Fotografie, Software-Entwickler und Leiter der Prozess- und Qualitätsoptimierung im Software-Bereich. Karl Wiegers hat unterschiedliche Teams geleitet: kleine Entwicklergruppen, das Web-Entwicklungsteam von Kodak und eine Abteilung von 500 Software-Ingenieuren, die eingebettete und hostgestützte Digital-Imaging-Software-Produkte entwickelt haben.

ÜBER JAMA SOFTWARE

Jama Software bietet die führende Plattform für Anforderungs-, Risiko- und Testmanagement. Teams, die komplexe Produkte, Systeme und Software entwickeln, verbessern mit Jama Connect und branchenorientierten Dienstleistungen die Umlaufzeiten, erhöhen die Qualität, reduzieren Nacharbeiten und minimieren den Aufwand für den Compliance-Nachweis. Der wachsende Kundenstamm von Jama ist führend in der modernen Entwicklung und umfasst mehr als 600 Unternehmen, darunter SpaceX, Boston Scientific, Lyft, Deloitte, Alight, Samsung und Caterpillar.