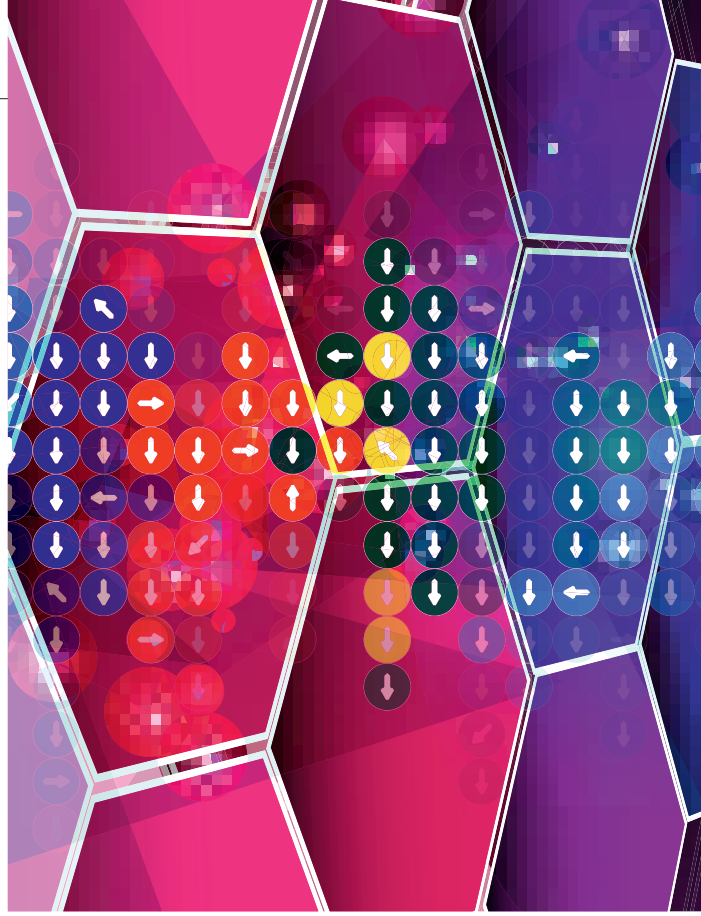

Software investitionssicher produzieren

Beständig ist nur der Wandel

Wenn Kunden ein Softwareprojekt beauftragen, ist von Anforderungen die Rede. Es wird über funktionale Anforderungen gesprochen und auch über nicht-funktionale. Worüber jedoch nicht gesprochen wird, ist Investitionssicherheit. Das ist doch seltsam, oder?

Mit den funktionalen Anforderungen wird beschrieben, was eine Software leisten soll, während die nicht-funktionalen Anforderungen etwas über die Qualität des Systems aussagen. Funktionale Anforderung könnte beispielsweise sein, ein Shopsystem zu entwickeln. Nicht-funktional könnte gefordert sein, dass das System eine gewisse Anzahl von Transaktionen pro Minute bewältigen kann oder dass das Antwortverhalten eine maximale Dauer nicht überschreiten darf. Über all diese Dinge sprechen Auftraggeber und Auftragnehmer. Worüber jedoch nicht gesprochen wird, ist die Investitionssicherheit. Die setzt der Auftraggeber voraus. Er erwartet vom Auftragnehmer, dass die Software in nachhaltiger Weise produziert wird, so dass über viele Jahre hinweg Korrekturen und Ergänzungen vorgenommen werden können. Für den Auftraggeber ist die Investitionssicherheit von enormer Bedeutung und trotzdem wird nicht darüber gesprochen. Wie kann das sein?

WEB-TIPP:
www.ccd-school.de

Investitionssicherheit durch Wartbarkeit?

Ein gängiger Begriff in diesem Zusammenhang ist Wartbarkeit. Software soll wartbar sein. Doch die im Begriff Softwarewartung steckende Vorstellung führt uns nicht zum Ziel der Investitionssicherheit. Wartung ist etwas proaktives; bevor etwas kaputt geht, werden periodisch Wartungstätigkeiten durchgeführt, damit erst gar nichts kaputt geht. Durch Wartung soll sichergestellt werden, dass die ursprünglich vereinbarte Leistung über die gesamte Lebenszeit unverändert erbracht wird. Im Maschinenbau ist das eine gute Sache. Im Bereich der Softwareentwicklung geht es aber um etwas fundamental anderes: Software soll sich auf Dauer verändern lassen. Investitionsschutz bedeutet hier Wandelbarkeit. Software soll eben nicht so bleiben wie sie ist, sondern sich über viele Jahre verändern lassen, sowohl im funktionalen wie im nicht-funktionalen Sinn. Wartungstätigkeiten gibt es bei Software im Betrieb. Dort wird pro-aktiv sichergestellt, dass die Software ohne Probleme betrieben werden kann, in dem Festplattenplatz, Prozessorauslastung und andere Ressourcen beobachtet werden.

Um Wandelbarkeit und damit Investitionssicherheit herzustellen, muss ein Team Software so produzieren, dass es ständig mit der Wandelbarkeit der Software in Kontakt kommt.

Stefan Lieser,
freiberuflicher Berater und Trainer für Softwareentwicklungsprozesse und Clean Code Development

Wandelbarkeit!

Durch Wandelbarkeit wird die Investition des Auftraggebers geschützt. Wandelbare Software macht es möglich, auf Dauer Änderungen am System vorzunehmen, mit mehr oder weniger gleichem Aufwand. Ob ein Feature zu Beginn der Entwicklung realisiert wird oder erst nach vielen Jahren, darf keinen nennenswerten Unterschied machen. Da das System über die Jahre gewachsen ist, wird der Aufwand naturgemäß leicht ansteigen. In Projekten ohne Blick auf die Investitionssicherheit werden Änderungen jedoch irgendwann faktisch un-

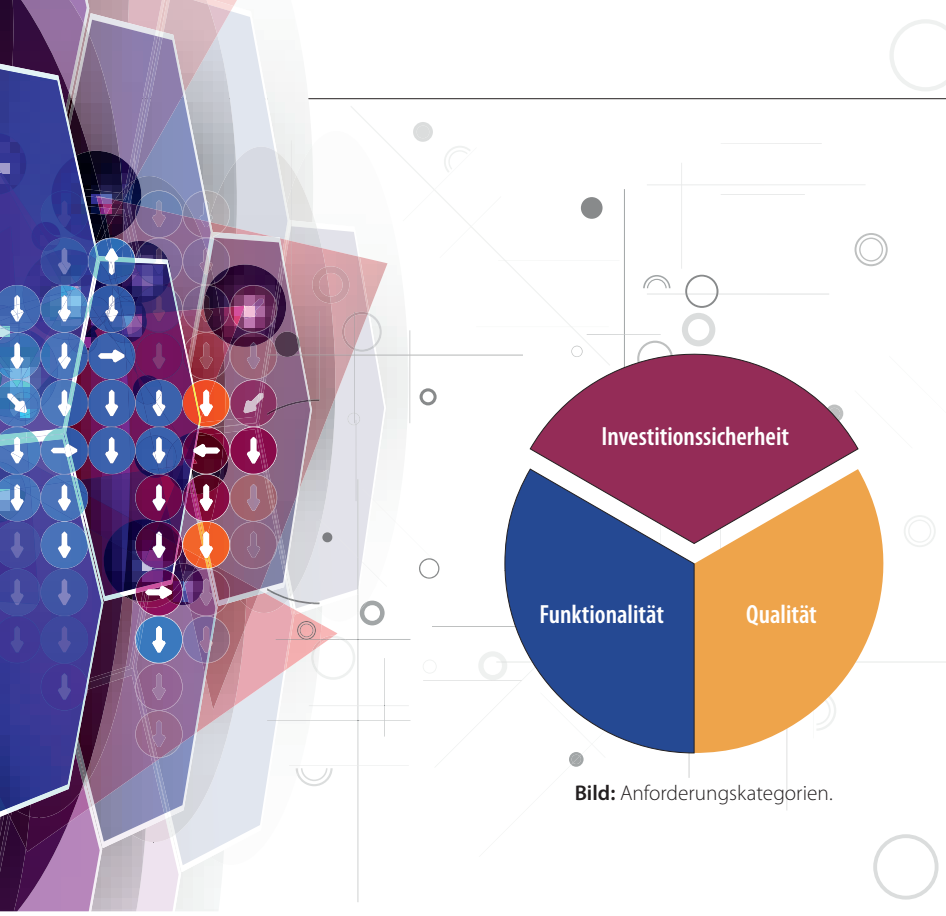

Bild: Anforderungskategorien.

möglich. Ferner leidet dann die Korrektheit. Hohe Aufwände für Tests sind die Folge. Investitionssicherheit bedeutet also nicht nur, Änderungen zu gleichbleibenden Kosten sondern auch unter Beibehaltung der Korrektheit.

Investitionssicherheit ist implizit gefordert

Kunden formulieren ihren Wunsch nach Investitionssicherheit nicht explizit und dennoch ist es die Aufgabe der Softwareentwickler, nachhaltig zu produzieren. Eine Parallele: Im Supermarkt habe ich Anforderungen an Margarine. Sie soll vielleicht leicht salzig schmecken und sich gut verstreichen lassen. Ich stelle aber nicht explizit bei jedem Einkauf die Anforderung, dass die Margarine meine Gesundheit nicht gefährden darf. Und trotzdem tut sie das nicht, weil dafür implizit gesorgt ist. Ähnliches gilt für Software: Softwareentwickler müssen implizit Investitionssicherheit herstellen. Andernfalls drohen hohe finanzielle Schäden, denn dass Software weiter entwickelt werden muss ist unstrittig. Anpassungen an neue oder geänderte gesetzliche Vorgaben wären da zu nennen. Oder auch der Druck, der sich aus Wettbewerb ergibt. Liefert der Konkurrent ein neues Feature, muss nachgezogen werden, sonst ist man langfristig vom Markt.

Was tun?

Neben einem veränderten Bewusstsein brauchen wir dazu vor allem Konzepte, mit denen sich Wandelbarkeit zuverlässig herstellen lässt. Die Herausforderung beginnt schon damit, dass sich Wandelbarkeit nicht messen lässt. Durch Inaugenscheinnahme des Quellcodes lässt sich nicht mit Gewissheit sagen, ob die Software wandelbar ist. Abwesenheit von Wandelbarkeit lässt sich manchmal erkennen. Abhängigkeiten wären hier zu nennen. Abhängigkeiten innerhalb der Struktur von Software erschweren die Wandelbarkeit, weil sich Änderungen so auf viele Bereiche der Software ausdehnen, statt isoliert zu sein. Ein weiteres Problem stellt die Vermischung von Aspekten dar. Gibt es keine klaren Zuständigkeiten, sind also die Aspekte in der Struktur nicht sauber getrennt, wirken sich Änderungen auf mehr Bereiche der Software aus, als von der Sache her erforderlich. Damit werden die Änderungen kompliziert und aufwändig.

Wandelbarkeit muss gelebt werden

Um Wandelbarkeit und damit Investitionssicherheit herzustellen, muss die Wandelbarkeit ständig herausgefordert werden. Ein Team von Softwareentwicklern muss Software so produzieren, dass sie ständig mit der Wandelbarkeit der Software in Kontakt kommen. Sie müssen rechtzeitig erkennen können, dass die geschaffenen Strukturen plötzlich nicht mehr wandelbar sind, um sofort gegensteuern zu können.

Für die Praxis bedeutet das, dass Teams alle zwei Tage ein Inkrement liefern müssen. Ein Inkrement zeichnet sich dadurch aus, dass der Product Owner dazu sein Feedback geben kann. Mit Product Owner bezeichnen wir die Rolle, die dafür zuständig ist, dem Team die Anforderungen verbindlich zu nennen und die fertigen Arbeitsergebnisse verbindlich abzunehmen. Feedback kann der Product Owner nur geben, wenn das Inkrement einen Durchstich durch die üblichen Aspekte von Benutzerschnittstelle, Domänenlogik und Ressourcenzugriff darstellt. Die Anforderungen werden durch das Team also in einem technischen Sinne vertikal zerteilt. Dem Product Owner wird nach zwei Tagen die Realisierung eines Ausschnitts der gesamten Anforderungen übergeben, der einen kleinen Teil Benutzerschnittstelle, einen kleinen Teil Domänenlogik und einen kleinen Teil Ressourcenzugriff enthält. Das gelieferte Ergebnis kann er installieren und ausführen und kann Feedback dazu geben.

Beständig ist nur der Wandel

Die kurze Iterationsdauer von zwei Tagen führt zu regelmäßigem Feedback durch den Product Owner. Für die Softwareentwickler ergibt sich aber noch ein weiterer Effekt, der im Sinne der Wandelbarkeit im Vordergrund steht: das Team lässt sich alle zwei Tage von neuen Anforderungen überraschen und ist somit alle zwei Tage gefordert, die Wandelbarkeit unter Beweis zu stellen. Unsere Erfahrung aus der Begleitung vieler Teams zeigt, dass tendenziell zu viel vorausgeschaut wird, als zu wenig. Insofern sollen die kurzen Iterationen ganz bewusst dazu genutzt werden, jeweils nur das nötigste zu tun, um den kleinen Ausschnitt von Anforderungen umzusetzen. Schon zwei Tage später wird sich dann zeigen, ob die Wandelbarkeit hergestellt wurde, denn nur dann lassen sich weitere Anforderungen bequem umsetzen.

Stefan Lieser