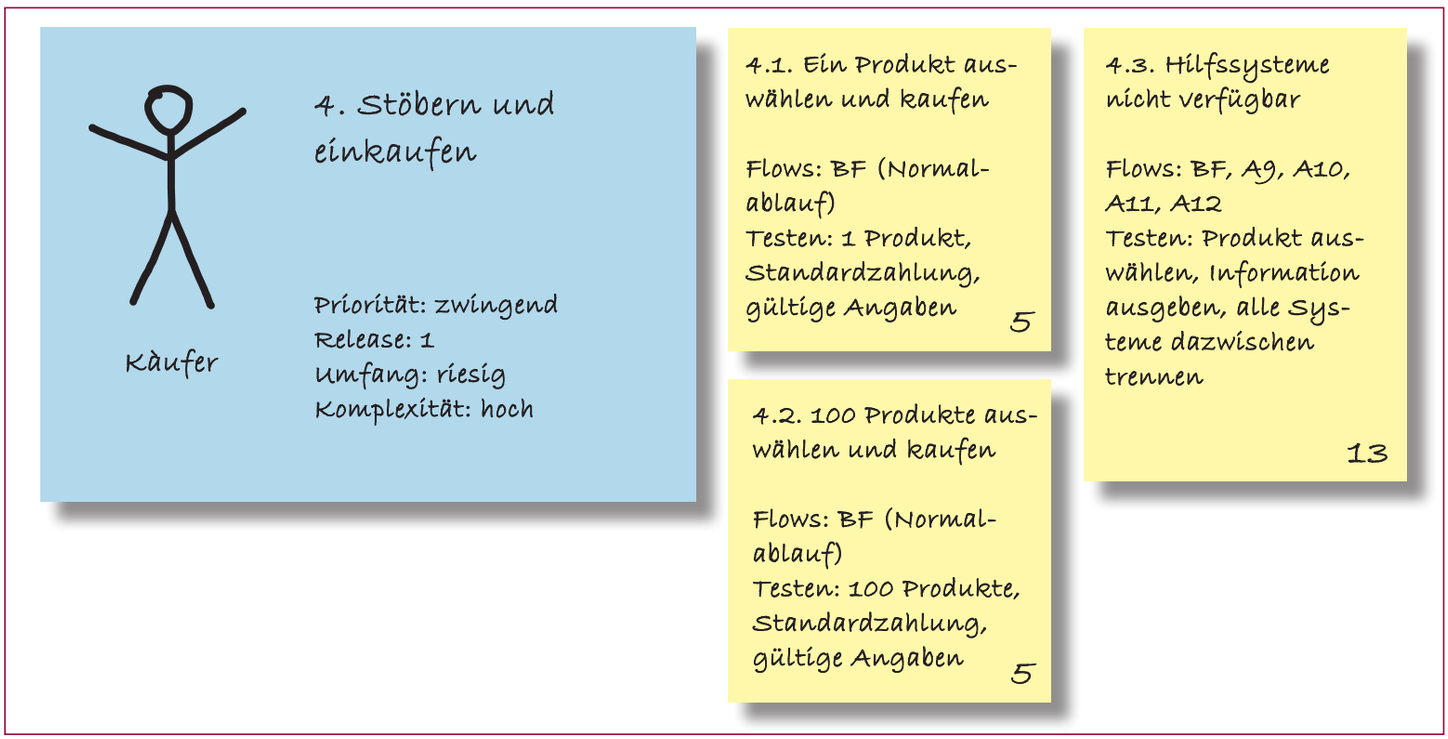
Auslaufmodell Use Case?

Die Definition von Anforderungen mittels Use Case ist mit agiler Softwareentwicklung etwas ins Abseits geraten. Doch das muss nicht heissen, dass sich die beiden Konzepte nicht kombinieren liessen. Alexander Gautschi, Stefan Meier



Ein Use Case und drei Slices, je auf einem Notizzettel aufgezeichnet. Quelle: Use-Case 2.0, Ivar Jacobson

Solche Geschichten ereignen sich regelmässig in der Projektarbeit: «Herr Müller, bitte stellen Sie die Anforderungen bis Ende nächster Woche fertig!» «Aber Herr König, für die Spezifikation aller Use Cases brauche ich noch mindestens vier Wochen …» «Dann verwenden Sie eine andere Technik, ich kann es mir nicht leisten, so viel Aufwand in unnötige Dokumentationen zu stecken, die erstens niemand liest, und zweitens zu kompliziert sind.»

Dieses fiktive Beispiel zeigt die Problematik: Use Cases (auf Deutsch: Anwendungsfälle) werden so weit überformalisiert, dass letztlich weder die Kunden (Stakeholder) noch die Entwickler viel Nutzen daraus ziehen. Doch das Problem von Herrn Müller kann gelöst werden.

Ein altbewährtes Modell

Das Prinzip des Use Cases wurde von Ivar Jacobson entwickelt, damals primär, um Aktionsabläufe von komplexen Telekommunikationssystemen zu beschreiben. An der «OOPSLA '87»-Konferenz stellte er diesen Ansatz erstmals öffentlich vor. Im selben Jahr gründete Jacobson eine Firma, um einen unterstützenden Softwareentwicklungsprozess zu entwickeln («Objectory»).

Mit der Einführung der UML Notation (Unified Modeling Language) rückten die Use Cases weiter ins Zentrum des allgemeinen Interesses: als einfaches Modellierungselement für die Darstellung der Kundenanforderungen auf einer detaillierten Ebene. Diese Position wurde noch verstärkt durch die Einführung des Rational Unified Process (RUP) im Jahre 1997. Hierbei standen die Use Cases im Zentrum des «4+1 View Model of Architecture» des Informatikers Philippe Kruchten.

Das 1994 erschienene Sachbuch «The Object Advantage» des UML-Mitbegründers Ivar Jacobson erweiterte den Einsatz des Use Case auf die Modellierung von Geschäftsprozessen. Damit hielt der Use Case Einzug ins Reengineering und ins Modellieren von Grosssystemen. Mit der Veröffentlichung des UML 2.0 wurden nur minimale Änderungen am Use-Case-Konstrukt eingeführt – dessen Einsatz hatte sich bewährt und ist auch heute noch weitgehend gleich in der aktuellen UML-Version definiert.

Funktionalität statt Dokumentation

In den letzten Jahren haben sich agile Vorgehensweisen in der Softwareentwicklung stark verbreitet. Damit einher ging auch eine Verschiebung von Prioritäten. Unter anderem wurde die Wichtigkeit einer funktionierenden Software über die Wichtigkeit der Dokumentation der Anforderungen gestellt. So

Alexander Gautschi ist selbstständiger Berater mit langjähriger Erfahrung und kennt die Gründer von RUP und Use Cases noch persönlich.

Stefan Meier ist Managing Consultant und Head Requirements bei der SwissQ Consulting AG.

steht etwa im Agile Manifesto: «Working Software over comprehensive Documentation.»

Die am weitesten verbreitete agile Entwicklungsmethode ist Scrum. Meist werden in Scrum User Stories zur Dokumentation von Anforderungen verwendet. Im Vergleich zu einer herkömmlichen Spezifikation mittels Use Case sind die User Stories sehr leichtgewichtig. Der Trend hin zu «Just-in-Time Specification» ist klar erkennbar, das heisst, nur dann Aufwand in eine Spezifikation zu stecken, wenn danach unmittelbar die Anforderungen umgesetzt werden. Somit werden Fehlinvestitionen verhindert, die entstehen, wenn alle Anforderungen im selben Umfang spezifiziert werden, viele jedoch nie in dieser Form realisiert werden.

Agilität mit Use-Case 2.0

2011 haben Jacobson, Spence und Bittner die Erfahrungen und Erkenntnisse aus Use Cases im mittlerweile mehr als 20-jährigen Einsatz zusammengefasst und ein modernes Einsatzkonzept unter dem Namen «Use-Case 2.0» zusammengestellt, das auch die Bedürfnisse des agilen Umfeldes zu befriedigen vermag. Dabei sollten die bewährten Aspekte von Use Cases beibehalten werden. Im neuen Ansatz sind Use Cases immer noch Use Cases, sie werden nach wie vor mit UML-Diagrammen gezeichnet und mit Geschichten (Szenarien) beschrieben. Zu jedem Use Case gehören die entsprechenden Testfälle (Acceptance Test Driven Development). Das Neue am Use-Case-2.0-Ansatz sind die Use-Case Slices. Ein Use-Case Slice umfasst ein oder mehrere Szenarien (inklusiv deren Testfälle), die einen klaren Wert für den Stakeholder darstellen und eine Grösse besitzen, die es erlaubt, sie in einem Sprint vollständig umzusetzen. Ein Use-Case Slice wird in seiner Vollständigkeit entworfen, implementiert und getestet. Für diese Planungseinheiten gelten selbstverständlich die bewährten Mittel von Priorisierung und Risikomanagement.

Der Umfang der Spezifikation dieser Slices richtet sich nach den konkreten Projektbedürfnissen und kann beliebig schlank sein. Eine bewährte Methode sind kleine Kärtchen an Pinnwänden, die auf der Vorderseite eine kurze Beschreibung der Szenarien enthalten, auf der Rückseite die dazugehörenden Testfälle, die auch als Abnahmekriterien für die Stakeholder dienen. Dadurch ist jederzeit transparent, wie der Stand der Umsetzung eines Szenarios (Kärtchens) ist. Ebenfalls ist damit möglich, bereits sehr früh Teile des Systems auszuliefern und Geschäftswert zu generieren. Die Software entsteht somit inkrementell, immer getrieben durch Prioritäten aus dem Business, und dies über den ganzen Produktlebenszyklus hinweg.

Mit obigem Ansatz kann der Use Case gewinnbringend auch in einem Scrum-Umfeld eingesetzt werden und bringt einige Vorteile. Ein Use-Case-Diagramm stellt den gesamten Scope eines Projekts dar. Die einzelnen Slices eines Use Cases können in unterschiedlichen Sprints umgesetzt werden, je nach Priorität und Beitrag zum Geschäftsnutzen. Ebenfalls stellen sie als Slices gebündelt die Planungseinheiten im Backlog dar. So entsteht stückweise das gesamte System, ohne dass man den Kontext verliert. Daneben werden die Stakeholder intensiv in den Kommunikationsprozess eingebunden, da der Fokus jeweils auf den Szenarien bleibt. Damit ist gewährleistet, dass die Anforderungen auch tatsächlich die Kundenbedürfnisse repräsentieren und nicht von Designer oder Entwickler «erahnt» werden müssen. Ebenfalls erleichtert die Erfassung der Testfälle bereits bei der Use-Case-Erstellung die Erfüllung der Definition of Done, die beschreibt, wann ein Slice fertig umgesetzt ist.

Auslaufmodell Use Case – oder doch nicht

Wie auch der neueste «Requirements Trends & Benchmarks 2013 Report» von SwissQ Consulting zeigt, ist die Use-Case-Beschreibung immer noch das mit 51 Prozent am häufigsten eingesetzte Spezifikationsmittel. Sie ist ein flexibles Instrument, da die Detaillierung der Spezifikation nur so weit vorgenommen wird, wie unbedingt nötig. Ein Use-Case-Diagramm zeigt jederzeit den kompletten Funktionsumfang auf einer für die Stakeholder verständlichen Flughöhe. Mit den Use-Case Slices werden die Vorteile einer agilen Entwicklung – kleine Einheiten, die vollständig umgesetzt werden – integriert. Durch den Zusammenhang von Use Case und Use-Case Slices behält man jederzeit den Überblick über den Systemumfang und die Zusammenhänge.

Wenn die Use-Case-Spezifikationen immer noch so dick wie der letzte «Brockhaus» ausfallen, ist es höchste Zeit, abzuspecken und den Use Cases wieder neues Leben einzuhauchen mittels den oben beschriebenen Ansätzen. Sie eignen sich nach wie vor, um einen schnellen Überblick über die Systemfunktionalität zu erlangen. Use Cases sind der Dreh- und Angelpunkt in der Kommunikation zwischen Stakeholder und Entwicklungsteam, besitzen Abnahmekriterien und Benutzerinteraktionen und stellen auch in einem agilen Projekt das Rückgrat des zu entwickelnden Systems dar. Mit dem Use-Case-2.0-Ansatz werden die Use Cases noch lange kein Auslaufmodell sein. <

DIE VERSCHIEDENEN USE CASES

In der Praxis werden Begriffe oftmals sehr unterschiedlich oder gar falsch verwendet. Die Unterscheidung zwischen «Business Use Case» und «System Use Case» ist primär eine Frage des Abstraktionsniveaus, also der Flughöhe.

Business Use Case: In der Geschäftsprozessanalyse werden Business Use Cases eingesetzt, um grobe Geschäftsszenarien zu beschreiben. Business Use Case beschreibt kundenorientierte Prozesse (Leistungserbringer) einer Firma oder Organisation, unabhängig davon, wie das Ganze oder Teile davon umgesetzt werden.

System Use Case (Use Case): wird eingesetzt, um die Automatisierung / Systemunterstützung eines Prozesses, Teilprozesses oder Prozessschritts zu beschreiben.

User Story: Die Unterscheidung zwischen User Story und Use Case wird in der Praxis kontrovers diskutiert. Wir sind der Ansicht, dass User Stories jeweils Teilausschnitte aus einem Use-Case-Szenario darstellen. Sie ist in Prosa verfasst und beinhaltet aus Anwendersicht (User) ein Szenario (Story), das für den Stakeholder noch einen Wert generiert, dabei aber so klein ist, dass es im jeweiligen Sprint (oder Iteration) realisierbar ist.

Storyboard: Skizziert ein Nutzerszenario anhand einer Abfolge von Illustrationen, Screens oder Bildern. Es wird eingesetzt, damit die Stakeholder einen Eindruck erhalten, wie sie künftig mit dem System konkret interagieren werden.

Im Gegensatz zu User Story und Storyboard fasst der Use Case alle Wege zusammen, die zur Erreichung eines fachlichen Zieles möglich sind. Dazu gehört auch die Beschreibung von Varianten, Fehler- und Ausnahmefällen. Die thematische Gruppierung von User Stories ermöglicht die formale Zusammenfassung dieser Informationen in einem einzigen Use Case. Die sogenannten Use-Case-Szenarien sind dann mehr oder weniger Abbilder der ursprünglichen User Stories.