Agil heißt nicht „undokumentiert“

16 Februar 2018 | **Insight Zühlke** | **Jochen Reber**
Lesezeit: 4 Minutes

Im Vergleich zur klassischen Softwareentwicklung spielt die Dokumentation bei der agilen Softwareentwicklung eine geringere Rolle und ist oftmals nur in Form von User Stories vorhanden. Das kann bei langlaufenden Softwareprojekten dazu führen, dass es keinen Überblick über die Funktionalität und die Abläufe der Software gibt – ein großes Problem. Wie sollte man also die Dokumentation bei solchen Projekten handhaben?

„Working software over comprehensive documentation“ – dieser Satz aus dem Agilen Manifest hat es zu einer gewissen Berühmtheit gebracht. Weit weniger bekannt ist der Zusatz, der danach kommt. Er besagt, dass auch die Dokumentation wichtig ist – wenn auch weniger wichtig als funktionierende Software („while there is value in [documentation], we value [working software] more.“ http://agilemanifesto.org/). Gerade bei langfristigen Software-Projekten geht jedoch das Eine nicht ohne das Andere.

In klassischen Prozessmodellen (z.B. V-Modell oder RUP) galt die Devise, dass eine vollständige Detailspezifikation in Form von eines Lasten- und Pflichtenhefts vorliegt, bevor implementiert wird. So soll gewährleistet werden, dass sowohl das „Was?“ als auch das „Wie?“ verständlich und nachlesbar hinterlegt sind (z.B. um die Anforderungen für einen Auftragnehmer aus der Sicht des Auftrags zu dokumentieren).

Agile Softwareentwicklung braucht weniger Dokumentation

In der Welt der agilen Softwareentwicklung wird mehr Wert auf die persönliche Kommunikation und die Entwicklung von funktionierender Software gelegt. Die einzige Form der fachlich-inhaltlichen Dokumentation bilden oft User Stories. Diese sind knapp geschrieben und dienen als Aufforderung zur Kommunikation. Sie dokumentieren das Delta zum Ist-Zustand und verzichten meistens auf eine Beschreibung des Kontexts. Dieser ist den Projektbeteiligten in der Regel bestens bekannt. Das kann auf lange Sicht zu Problemen führen, etwa wenn es einmal auf das Gesamtverständnis für ein Softwaresystem und auf unterschiedliche Perspektiven ankommt (Endnutzer, Architekt, Administrator, Produktmanagement, Entwickler, etc.).

User Stories reichen nicht immer

Liegt die Dokumentation nur in Form von User Stories vor, so muss man für die gesuchte Information sämtliche dieser Stories chronologisch rückwärts nachverfolgen. Unter Umständen kann es sogar schlichtweg unmöglich sein, unter hunderten und tausenden von User Stories die relevanten zu finden und daraus den benötigten Kontext abzuleiten.

Falls z.B. einer, in die Jahre gekommenen, agil entwickelten Software neue Features hinzugefügt werden sollen, bedarf es unter Umständen einer mehrwöchigen Analyse-Phase, um zu verstehen, wie das System überhaupt funktioniert. Eine Code Analyse funktioniert nur effizient bei lokalen Features. Und eine Expertenbefragung kommt spätestens dann nicht mehr in Frage, wenn von den ursprünglichen Projektmitgliedern niemand mehr erreichbar ist.

Auch agil entwickelte Software profitiert von Dokumentation

Ein vereinfachtes Beispiel anhand einer agil entwickelten Personensuchfunktion: Die Entwicklung der Suche beginnt mit dem einfachsten Fall, einer genauen Personennamensuche. Anschließend erfolgt auch eine Suche nach Geburtsort und nach Geburtsdatum, dann werden Namenskombinationen (zuerst Nachname, dann Vorname, nur mittlerer Name, etc.) ermöglicht, und daraufhin Sonderzeichen und schließlich eine fehlertolerante Suche unterstützt. Für die Antwort auf die Frage „Ist es möglich, eine Suche mit einem falsch geschriebenen und mit Sonderzeichen versehenen Geburtsort durchzuführen?“ müssten sämtliche User Stories gelesen und verstanden oder der Code analysiert werden, da es keine Gesamtdokumentation gibt.

Neben der Entwicklung gibt es noch andere Bereiche, die auf eine Dokumentation angewiesen sind: Für die Erstellung des Benutzerhandbuchs, für den Support oder für Marketing-Material etwa ist ein Verständnis des Systems erforderlich. Daher ist es wichtig, dass es für Systeme mit hoher Komplexität und langer Lebensdauer auch eine Anforderungsdokumentation außerhalb von User Stories gibt.

Wie Dokumentieren in der agilen Softwareentwicklung?

Die grundsätzliche Beschreibung, was ein System leistet, und warum es das leistet, ist für die Nachvollziehbarkeit und Änderbarkeit eines Systems essentiell. Die Struktur der Dokumentation hängt dabei vom System ab. Beispiele für wesentliche Elemente, die der Dokumentation bedürfen, sind: Zweck des Systems, fachliche Funktionalität, unterstützte Nutzergruppen, Grobbeschreibung der Workflows, Auflistung und Kurzbeschreibung der Komponenten und ihrer Interaktionen und Schnittstellen sowie Außenschnittstellen des Systems.

Dabei geht es nicht um eine Detaildokumentation. In manchen Fällen mag dies notwendig sein (z.B. für die Außenschnittstellen), aber in vielen Fällen reicht eine Grobbeschreibung aus. Hier hilft der gesunde Menschenverstand dabei, zwischen dem Aufwand zur Pflege der Dokumentation und dem Nutzen für die Leser abzuwägen. Die konkrete Erfassung der Dokumentation kann mit unterschiedlichen Mitteln erfolgen, z.B. einem Anforderungsmanagement-Tool , einem UML-Tool, einem internen Wiki, oder als Prosa-Beschreibung in Word.

User Stories mögen den Bedarf für die laufende und folgende Entwicklung decken. Sie sind als Planungswerkzeug und für die entwicklungsbegleitende Kommunikation gedacht und hierfür auch gut geeignet. Allerdings ist ein Minimum an darüber hinaus gehender Dokumentation sowohl für die langfristige Pflege als auch für unterstützende Prozesse notwendig, auch in agilen Projekten. Sie erleichtert einem Software-Entwicklungsteam darüber hinaus auch die Einarbeitung neuer Mitglieder, was sich positiv auf Produktivität und Teamdynamik auswirkt.