mehr zum thema:
www.mbohlen.de/blog/artikel/projektvertraege-ohne-tauziehen/

PROJEKTVERTRÄGE: SCHLANKE REGELUNGEN ERSPAREN TAUZIEHEN

Umfragen unter Benutzern von IT-Systemen ergeben: Nur zwanzig Prozent der Funktionen eines IT-Systems werden „immer" oder „oft" benutzt, fast zwei Drittel hingegen „selten" oder „nie". Diese Zahlen haben den Umgang mit Anforderungen in IT-Projekten verändert: Lernen im Projekt ist ausdrücklich erlaubt, Änderungen sind willkommen. Diese Situation stellt neue „Anforderungen an die Anforderungen". Der Artikel zeigt, wie Anforderungen heute aussehen können, damit man mit ihnen sehr schlanke Projektverträge gestalten kann. „Money for nothing, changes for free" sind hier die zunächst unglaublich klingenden Stichworte. Auftraggeber und Entwicklungsorganisation ziehen am selben Strang, Änderungsanforderungen kosten ab jetzt kein zusätzliches Geld mehr und brauchen deshalb nicht mehr verhandelt zu werden.

der autor

Matthias Bohlen
(E-Mail: mbohlen@mbohlen.de)
ist unabhängiger Coach für Softwareentwicklungsteams, die hochproduktiv werden wollen. Er ist Mitglied des OBJEKTspektrum-Redaktionsteams.

Anforderungen heute

Alistair Cockburn hat als Gedankenstütze für die Gliederung von Anforderungen das Modell eines Segelschiffs benutzt (siehe Abb. 1, vgl. [Coc00]): Die Mastspitze stellt das übergeordnete Projektziel dar. Die Segel repräsentieren die zentralen, leicht zu merkenden Geschäftsziele, die der Auftraggeber mit dem neu zu schaffenden System erreichen möchte. Auf der Wasseroberfläche befinden sich die Ziele der späteren Benutzer. Unter Wasser schließlich findet man die Systemfunktionen, die später identifiziert werden, wenn Geschäftsziele und Benutzerziele klar sind.

Die Ziele der Benutzer sind zunächst das Wichtigste, denn eine neue Software muss so gestaltet sein, dass die Benutzer sie begeistert bedienen. Wie können Sie das erreichen? Ich lade Sie ein, zunächst eine bestimmte Art und Weise der Anforderungsdokumentation zu verwenden, die ich hier beschreibe. Es ist eine von vielen Möglichkeiten, die sich sehr gut bewährt hat. Wenn Sie möchten, machen Sie mit!

Stellen Sie sich vor, das neue System, ein *Online-Konferenzmanagementsystem*, existiert bereits, und ein Benutzer erzählt, welche Ziele er damit erreichen kann:

- Als *Sprecher* kann ich einen *Beitrag einreichen*, der von Gutachtern bewertet wird.
- Als *Gutachter* kann ich einen *Beitrag bewerten*, damit der Vorsitzende weiß, ob der Beitrag angenommen werden soll.

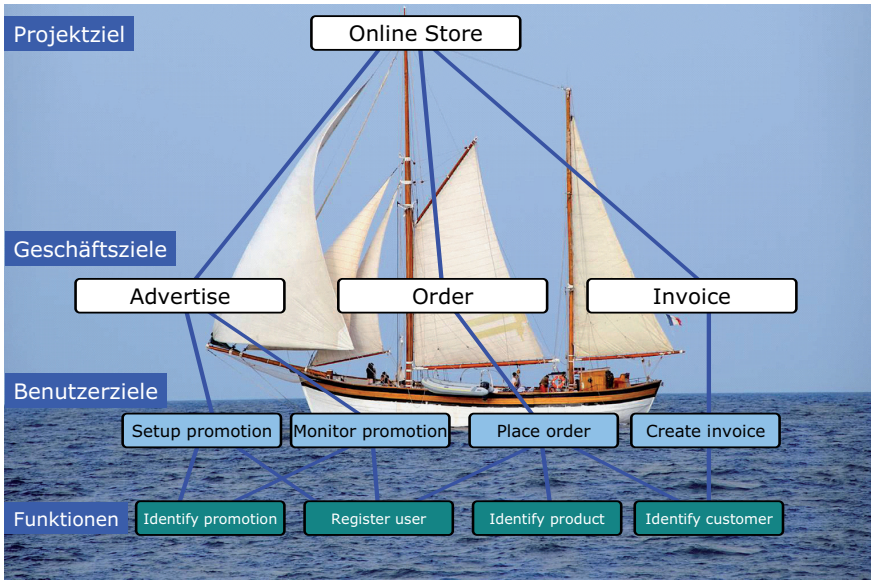

Abb. 1: Ein Schiff als Gliederungsmodell für Anforderungen.

- Als *Vorsitzender* kann ich die eingereichten *Beiträge auf Gutachter verteilen*, damit ich Bewertungen bekomme.
- Als *Vorsitzender* kann ich *Beiträge auflisten* und nach Bewertung sortieren, um das Meeting des Programm-Komitees vorzubereiten.

Fragen Sie die zukünftigen Benutzer nach ihren Zielen und schreiben Sie die Ziele immer in folgender Form auf:

- *Deutsch:* <Rolle> kann <etwas tun>, damit <Ziel>
- *Englisch:* <Role> can <do what> so that <why>[1])

Mit dieser normierten Zusammenfassung beginnt eine User-Story, d. h. eine einzelne Anforderung, die ein Benutzer an das neue System hat (vgl. [Coh04]) (obige Beispiele sind der Entwicklung des Online-Konferenzmanagementsystems „SplendiCon" (vgl. [Spl09]) entnommen). Notieren Sie die Story auf einer nicht zu großen Karteikarte (siehe Abb. 2).

Die Zahl links oben gibt die ID an, unter der Sie im Anforderungsmanagementsystem, Bug-Trracker oder Wiki-gestützten Akzep-

[1]) *In der Literatur findet man oft „As a <role>, I want to <do what> so that <why>". Doch die Form „<Role> can..." ist kürzer und passt besser auf eine Karte.*

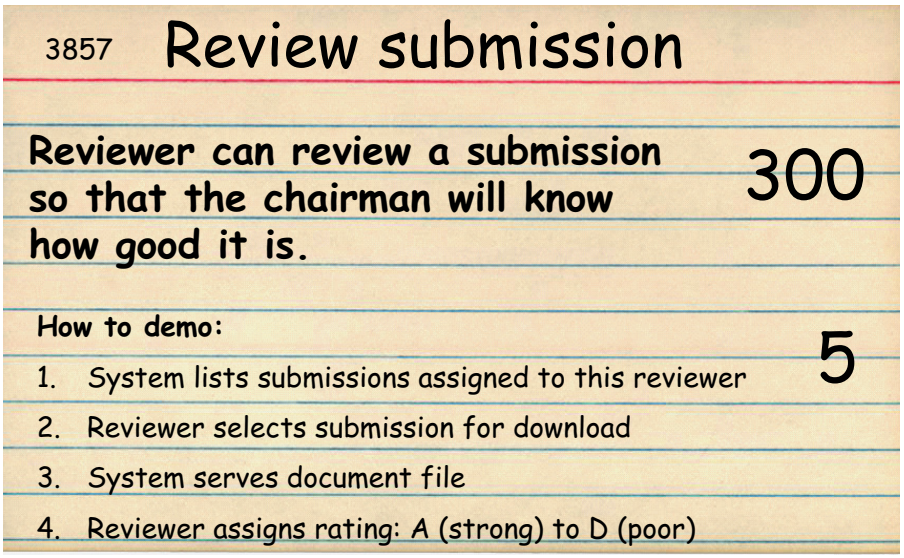

Abb. 2: Karteikarte mit User-Story.

tanztestsystem weitere Details hinterlegt haben. Der Titel der Karte hat immer die Form „etwas tun". Der Satz unter dem Titel ist die normierte Zusammenfassung. Weiter unten finden Sie den wichtigen Abschnitt „How to demo". Er beschreibt, was zu tun ist, um die Story am lauffähigen System vorzuführen, wenn sie fertig implementiert ist. Der Abschnitt ist damit ein Akzeptanztest. Die Zahl rechts oben ist der so genannte Geschäftswert (*Business Value*), eine dimensionslose Zahl, nach der man die Karten im Stapel sortieren kann. Sie gibt an, wie wertvoll es für den Kunden ist, wenn diese Karte auch umgesetzt wird. Die Karten mit dem höchsten Geschäftswert sollen vom Entwicklungsteam in der Regel immer als erste umgesetzt werden. Dabei ist „300" nicht unbedingt doppelt so wichtig wie „150", sondern lediglich wichtiger.

Lassen Sie die zukünftigen Benutzer innerhalb eines Anforderungsworkshops solche Story-Karten schreiben und mit Nadeln an eine Wand heften. Achten Sie darauf, dass die normierte Form (Titel, Zusammenfassung, „How to demo" usw.) eingehalten wird. Sortieren Sie den Kartenstapel dann nach Geschäftswert, nehmen Sie ihn mit und gehen Sie mit den Karten zu den Entwicklern. Erklären Sie, wie die Karten gemeint sind, und klären Sie mit Hilfe von kurzen Diskussionen die Essenz jeder Story ab. Lassen Sie die Entwickler schätzen, wie hoch der Aufwand für die Realisierung einer Story ist. Schreiben Sie diese Story-Points rechts unten auf die Karte. Ein Story-Point ist ein relatives Maß für die Komplexität einer Story: Eine Story mit 8 Punkten ist dabei mehr als doppelt so komplex wie eine Story mit 3 Punkten. Die Story-Points dienen zur Verfolgung des Projektfortschritts. Wenn Sie iterativ vorgehen, lassen Sie Ihr Entwicklungsteam messen, wie viele Storys am Ende einer Iteration wirklich funktionsfähig an den Kunden übergeben werden konnten. Addieren Sie die Punkte dieser fertigen Storys, dann erhalten Sie eine Zahl, die angibt, wie viel geschätzte Arbeit in dieser Iteration tatsächlich geleistet werden konnte. Wenn Sie das mit der Summe der Story-Points vergleichen, die auf den noch nicht umgesetzten Story-Karten stehen, haben Sie einen Begriff dafür, wie lange das Entwicklungsteam noch dafür brauchen wird.

Story-Points sind abstrakte Einheiten, ohne Bezug zur Zeitachse. Das ist ein Vorteil, weil Menschen die relative Größe von Dingen besser schätzen können als die absolute. Manchmal wird es als Nachteil empfunden, dass man den Bezug zur Zeitachse erst nachträglich herstellen muss, durch Messung am Ende einer Iteration. Dafür ist das Verfahren selbst-kalibrierend gegen Schätzfehler. Mike Cohn hat das Schätzverfahren für diese Story-Points in seinem Buch sehr gut und ausführlich beschrieben (vgl. [Coh05]).

Hier die Qualitätskriterien für gut geschriebene Story-Karten. Von Bill Wake (vgl. [Wak03]) stammt das Akronym **INVEST** für diese Kriterien. Eine gute Story weist folgende Eigenschaften auf:

- *Independent:* Sie ist unabhängig von anderen Storys realisierbar.
- *Negotiable:* Sie lässt Raum für Abstimmungen, z. B. für die Gestaltung der Benutzungsoberfläche.
- *Valuable:* Sie hat einen geschäftlichen Wert.
- *Estimatable:* Sie hat einen gut schätzbaren Aufwand.
- *Small:* Die Entwickler können mehrere Storys pro Iteration umsetzen.
- *Testable:* Man kann gut testen, ob die Story implementiert ist oder nicht.

Newkirk und Martin empfehlen zusätzlich noch *Constraint*-Karten, um Beschränkungen und andere nicht-funktionale Anforderungen an das neue System festzuhalten (vgl. [New01]). Die zentralen Sätze darauf könnten beispielsweise lauten:

- Das System muss in der Spitze bis zu 50 Benutzer gleichzeitig unterstützen.
- Das System muss mit der bestehenden Datenbank XY zusammenarbeiten.
- 80 % aller Datenbankabfragen müssen die Ergebnisse in weniger als zwei Sekunden auf den Bildschirm liefern können.

Wenn möglich, schreiben Sie auch für die *Constraints* ein „how to demo", also einen Akzeptanztest, der beweist, dass die nicht-funktionale Anforderung eingehalten wird.

Eine solche Karte allein ist aber nicht alles. Eine komplette Anforderung besteht nach Ron Jeffries (vgl. [Jef01]) aus *den drei C*:

- Card
- Conversation
- Confirmation

Das bedeutet: Erst, wenn Sie die Karte ausführlich mit den späteren Benutzern besprochen haben (*Conversation*) und eine Bestätigung in Form eines Akzeptanztests haben (*Confirmation*), ist die Anforderung komplett. Dann haben Sie die beste Grundlage für einen vertrauensgeprägten Projektvertrag.

Von den Anforderungen zum Projektvertrag

Den Kartenstapel, den Sie in der Anforderungsanalyse gewonnen haben, können Sie als Grundlage für die Gestaltung des Projektvertrages nutzen. Gerade bei Projekten der Öffentlichen Hand kommt es immer wieder zu folgendem Problem, das Sie mit dieser Gestaltungsweise leicht lösen können. Hier ein leicht übertriebenes *Anti-Beispiel*:

- Kunde und Dienstleister schließen einen Werkvertrag. Der Werkvertrag

schreibt fest, dass im Projekt ein Werk aufgrund der in den Ausschreibungsunterlagen genannten Anforderungen erstellt werden muss.
- Kunde und Dienstleister lernen im Laufe des Projekts hinzu und stellen fest, dass die ursprünglich gestellten Anforderungen geändert werden müssen.
- Für die Änderungen werden Änderungsanforderungen (*Change Requests*) geschrieben, die in regelmäßigen Meetings diskutiert, bewertet und verabschiedet werden müssen. Dadurch vergrößert sich der Aufwand und die Kosten des Projekts steigen.
- Um das zu aufzufangen, rechnet der Kunde von vornherein mit diesem Effekt und versucht, den Preis des Werkes im Ausschreibungsverfahren ohnehin schon einmal auf die Hälfte zu drücken.
- Der Dienstleister kennt den Effekt natürlich auch und versucht, den verlorenen Profit bei der Bearbeitung der Änderungsanforderungen wieder hereinzuholen. Er kann also eigentlich nicht an einer zügigen Abarbeitung der Änderungsanforderungen interessiert sein.
- Kunde und Dienstleister finden sich nun in einer Art Tauziehen auf verschiedenen Seiten des Seils wieder – eine wenig ergiebige Situation.

Jeff Sutherland erzählt immer wieder von einer IT-Dienstleistungsfirma, der diese Vorgehensweise zunehmend auf die Nerven ging. Sie wollten sich auf eine Ausschreibung bewerben und waren durch Schätzungen zu dem Ergebnis gekommen, dass der Preis, den sie realistisch fordern müssten, etwa 10 Mio. Euro betragen würde. Der Vertriebsmann des Dienstleisters rief den Mitarbeiter des zukünftigen Kunden an, der die Ausschreibung verantwortete. Das Gespräch spielte sich in etwa so ab:

- **DIENSTLEISTER:** „Wir schätzen, dass der Aufwand bei ca. 10 Mio. Euro liegen wird und möchten wissen, ob wir damit in einer von Ihnen als realistisch gesehenen Größenordnung liegen."
- **KUNDE:** „Ich kann Ihnen nur sagen, dass Sie schlechte Karten haben, wenn Sie mehr als 5 Mio. Euro verlangen."
- **DIENSTLEISTER:** „Mit oder ohne Änderungsanforderungen?"
- **KUNDE:** „Wie bitte?"
- **DIENSTLEISTER:** „Ich verstehe Sie nun so, dass Sie mit 5 Mio. rechnen und jede Änderungsanforderung, die Sie im Laufe des Projekts stellen, extra bezahlen möchten."
- **KUNDE:** „Im Prinzip richtig. Über die Änderungsanforderungen werden wir natürlich jedes mal verhandeln müssen."
- **DIENSTLEISTER:** „Aus Erfahrung können wir sagen, dass sich die Anforderungen im Mittel über den gesamten Projektverlauf zu 66 % ändern werden. Also bezahlen Sie voraussichtlich am Ende 5 Mio. mal 1,66, also 8.3 Mio. Euro, und haben jede Menge Verhandlungsarbeit während des Projekts. Wir bieten Ihnen eine Flatrate von 10 Mio. Euro an, inklusive aller Änderungsanforderungen. Sie sparen den ganzen Ärger mit den Änderungsanforderungen ein und bekommen die Änderungen kostenlos."
- **KUNDE** (*staunendes Schweigen*): „Wie machen Sie das?"
- **DIENSTLEISTER:** „Unser Vertragsmodell basiert auf dem Scrum-Vorgehen. Sie bekommen alle zwei bis vier Wochen lauffähige Software vorgeführt und können nach jeder Vorführung entscheiden, die Anforderungen für die nächsten zwei bis vier Wochen anzupassen. Die geänderten Anforderungen werden geschätzt und dem Stapel von Anforderungen (*Backlog*) hinzugefügt. Dafür fällt die unwichtigste Anforderung, die sich ganz unten im Stapel befindet, heraus. Oben auf dem Stapel liegen immer die wichtigsten Anforderungen, die auch zuerst realisiert werden sollen."
- **KUNDE:** „Das klingt auf den ersten Blick faszinierend. Ich weiß jedoch nicht, ob Sie damit durch unseren Ausschreibungsprozess kommen."
- **DIENSTLEISTER:** „Machen wir es so: Ich schicke Ihnen zwei Angebote: Ein herkömmliches über 5 Mio. Euro plus kostenpflichtiges Änderungsmanagement und ein modernes, wie eben beschrieben, zu einer Flatrate von 10 Mio. Euro und „Changes for free". Sie entscheiden dann selbst, welches von beiden Sie annehmen wollen."
- **KUNDE:** „OK, ich bin gespannt."
- **DIENSTLEISTER:** „Ich gebe Ihnen noch mehr. Sie haben bei uns das Recht, das Projekt jederzeit unterwegs abzubrechen, sobald Sie glauben, dass die Software bereits das tut, was Sie brauchen."
- **KUNDE:** „Abbrechen, also einfach raus aus dem Vertrag?"
- **DIENSTLEISTER:** „Richtig. Sie bekommen die lauffähige Software und bezahlen uns 20 % des Geldes, das Sie dadurch einsparen, weil wir so gut gearbeitet haben, dass Sie die Anwendung früher in Betrieb nehmen können. 80 % des Geldes behalten Sie."
- **KUNDE:** „Da muss ich erst überlegen, ob wir das wollen."
- **DIENSTLEISTER:** „Wenn nicht, lassen Sie das Projekt einfach normal weiterlaufen. Sie bekommen dann mehr Funktionalität. Einverstanden?"
- **KUNDE:** „Gut, schicken Sie Ihre Angebote. Wir melden uns."

Der Kunde entschied sich zum Schluss für diesen Dienstleister und für das 10 Mio. Euro Angebot. Das Projekt wurde ein Erfolg. Dieser Kunde und dieser Dienstleister blieben nicht die einzigen – mehr und mehr Unternehmen interessieren sich für dieses agile Vertragsmodell.

Wie konnte das funktionieren? Während des Projekts wurden geänderte Anforderungen ebenfalls als Story- und *Constraint*-Karten erfasst und auf den existierenden Kartenstapel gelegt. Am unteren Ende des Stapels wurden jeweils so viele Karten entfernt, dass deren Aufwandssumme dem Aufwand auf der neu hinzugefügten Karte entsprach (siehe Abb. 3).

Dadurch ergeben sich „Changes for free": Änderungsanforderungen vergrößern den Aufwand des Projekts nicht. Kunde und Dienstleister haben beide das Interesse, das Projekt im bisherigen Rahmen weiterzuführen. Außerdem benötigt der Dienstleister den *Change-Request*-Prozess nicht als Einnahmequelle.

Der Dienstleister bot im obigen Gespräch dem Kunden zusätzlich das Recht zum vorzeitigen Abbruch des Projekts an. Wie funktionierte das? Der Dienstleister arbeitet mit seinem Entwicklungsteam immer die wichtigsten Storys vom oberen Ende des Stapels ab, also die mit dem höchsten Geschäftswert. Erfahrungen zeigen, dass die Kurve des geschaffenen Geschäftswerts über die Zeit einen nicht-linearen Verlauf hat, den man für dieses Vertragsmodell ausnutzen kann (siehe Abb. 4).

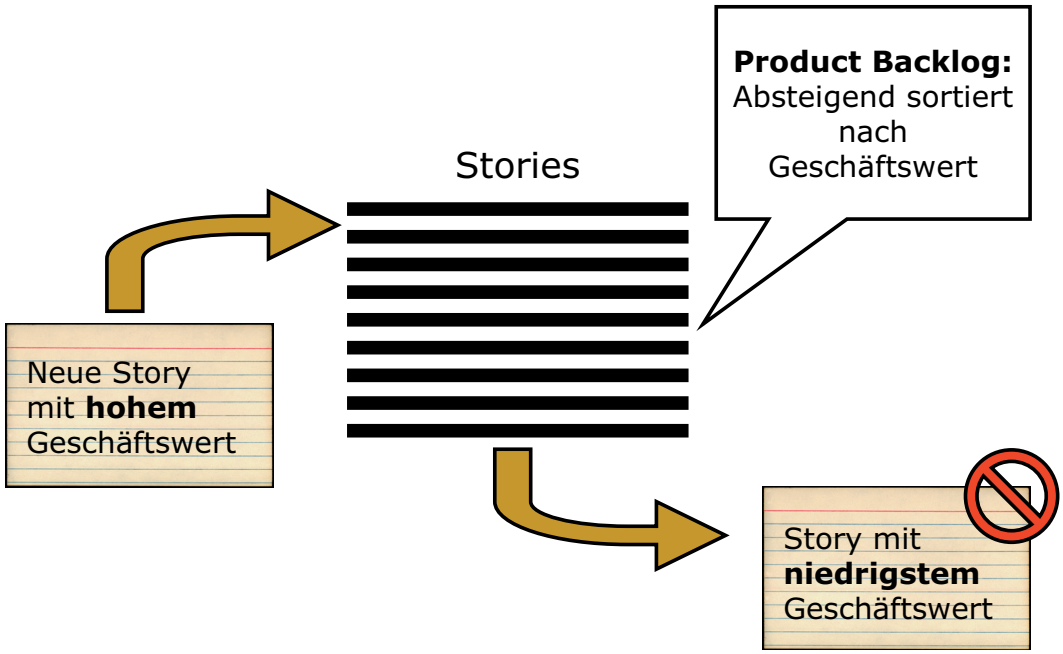

Abb. 3: Kostenneutrale Abwicklung von Änderungsanforderungen.

Am Anfang des Projekts wird etwas weniger Geschäftswert pro Woche erzeugt, weil das Team erst seine Entwicklungsumgebung aufbauen, die Sprache des Kunden erlernen und sonstige Vorbereitungen treffen muss. Dann gibt es einen Zeitraum, in dem die Kurve steil ansteigt, die hoch produktive Phase des Projekts. Danach flacht die Kurve ab, weil die weniger wichtigen Storys realisiert werden und kleine Feinschliff-Verbesserungen an den hochwertigen, bereits realisierten Storys vorgenommen werden.

Der Kunde könnte sich nun entscheiden, mit 80 % des Geschäftswerts zufrieden zu sein und das Projekt nach der Hälfte der Zeit abbrechen (in Abb. 4 etwas übertrieben dargestellt). Dadurch spart er die Hälfte der Kosten (5 Mio. Euro). Von dieser gesparten Hälfte gibt er 20 % als Prämie an den Dienstleister (1 Mio. Euro „Money for Nothing"), weil dieser den Geschäftswert so schnell erbracht hat. 80 % der gesparten Kosten (4 Mio. Euro) behält der Kunde selbst.

Der Dienstleister kann die Prämie dafür verwenden, in der entstehenden Pause zwischen den Projekten seine Leute zu schulen und in Ruhe ein neues Projekt für sein Entwicklungsteam zu akquirieren.

Gegenwind: die Risiken

Bisher wurde der Idealfall beschrieben – es gibt jedoch auch Risiken, an die man denken sollte.

Zunächst braucht es Vertrauen zwischen Kunde und Dienstleister. Ohne dieses Vertrauen ist die Gefahr groß, in alte Verhaltensweise zurückzufallen. Im einfachsten Fall würde das lediglich Nerven kosten, doch in der Regel führt so ein Rückfall auch zum Verlust der Kreativität auf beiden Seiten. Das bremst das Projekt stark ab und kann dazu führen, dass die Option „Money for Nothing" niemals ausgeübt werden kann – ein für beide Seiten unerfreuliches Szenario. Hier sollte man am besten einen erfahrenen agilen Coach einsetzen, der den Projektbeteiligten hilft, sich auf Vertrauen und Feedback zu konzentrieren.

Ein weiteres Risiko ist die Verführung, die in kostenlosen Änderungsanforderungen steckt. Ziel des Projekts ist es ja, ein lauffähiges Produkt herauszubringen. Wenn jedoch Änderungsanforderungen (weil sie ja kostenlos sind) in ungeordneter Weise benutzt werden, besteht die Möglichkeit, dass das Team zwischen sich

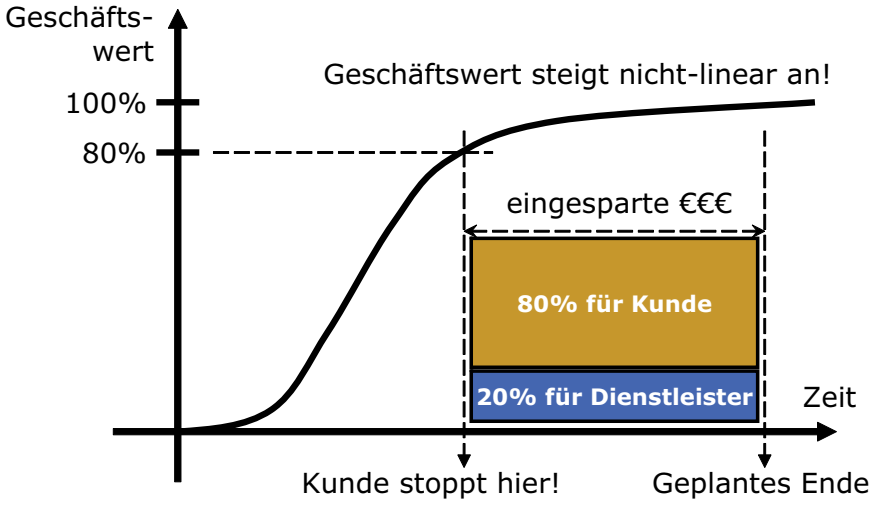

Abb. 4: Der Kunde kann bei genügend hohem Geschäftswert das Projekt mit Gewinn beenden.

ständig ändernden Zielen hin- und herspringt. In diesem Fall konvergiert das Projekt nicht mehr hin zu einem einheitlichen wertschaffenden Produkt. Diesen Effekt verhindert man am besten durch den Einsatz eines fähigen Produktmanagers (bei Scrum heißt dieser *Product Owner*), der auf die Konvergenz des Produktes achten muss.

Fazit

User-Storys und ein darauf basierendes agiles Vertragsmodell bringen Kunde und Dienstleister in dieselbe Interessenlage: Beide wollen das Projekt so bald wie möglich mit so hohem Geschäftswert wie möglich abschließen. Ein wesentlicher Konflikt ist damit von vornherein ausgespart. Die dadurch gewonnene Energie und Motivation können für das Projekt eingesetzt werden. ■

Literatur & Links

[Coc00] A. Cockburn, Writing effective Use Cases, Addison-Wesley 2000

[Coh04] M. Cohn, User Stories Applied: For Agile Software Development, Addison Wesley Professional 2004

[Coh05] M. Cohn, Agile Estimating and Planning, Prentice Hall PTR 2005

[Jef01] R. Jeffries, Essential XP: Card, Conversation, Confirmation, August 2001, siehe: http://xprogramming.com/xpmag/expcardconversationconfirmation/

[New01] J.W. Newkirk, R.C. Martin, Extreme Programming in Practice, Addison Wesley Professional 2001

[Sch07] K. Schwaber, What is Scrum?, White Paper 2007, siehe: www.scrumalliance.org/resources/227

[Spl09] SplendiCon-Konferenzmanagementsystem, siehe: www.splendicon.net

[Wak03] B. Wake, INVEST in Good Stories, and SMART Tasks, August 2003, siehe: http://xp123.com/xplor/xp0308/index.shtml