

#### 4.2.1 Das Product Backlog ist ein lebendes Dokument

Das Product Backlog verändert sich über die gesamte Projektlaufzeit: Neue Anforderungen werden aufgenommen, existierende Anforderungen modifiziert und verfeinert oder gestrichen. Außerdem kann sich die Priorität der Anforderungen ändern. Nicht nur die Software, auch das Product Backlog wächst somit iterativ-inkrementell: Gestartet wird mit einem grobgranularen Backlog, das von Sprint zu Sprint weiter detailliert und verfeinert wird. Das Product Backlog entspricht somit nicht einer traditionellen Anforderungsspezifikation oder dem Pflichtenheft: Die Anforderungen sind zu Beginn eines Scrum-Projekts weniger detailliert und vollständig beschrieben verglichen mit traditionellen Anforderungsdokumenten. Erst am Ende des Scrum-Projekts sind alle umgesetzten Anforderungen präzise und detailliert festgehalten.

Da das Product Backlog lebt und Anforderungsänderungen der Normalfall sind, kennt Scrum kein *Change-Request*-Verfahren. Oft ist es aber nützlich, Änderungen im Product Backlog zu dokumentieren, z.B. durch eine Versionierung des Backlog. So lassen sich diese zu einem späteren Zeitpunkt leichter nachvollziehen.

#### 4.2.2 Die Einträge sind priorisiert

Alle Einträge im Product Backlog sind priorisiert. Der Product Owner entscheidet darüber, wonach und wie die Anforderungen priorisiert werden. Üblicherweise wird das Product Backlog nach Nutzen, Risiko und Kosten priorisiert, siehe Abschnitt 4.7. Das Team unterstützt den Product Owner insbesondere bei der Identifizierung und Berücksichtigung technischer Risiken.

#### 4.2.3 Die Einträge weisen einen unterschiedlichen Detaillierungsgrad auf

Hochpriore Anforderungen werden detaillierter und präziser beschrieben als niedrpriore. Dies stellt sicher, dass die wichtigen Anforderungen präzise formuliert sind. Zugleich wird der Aufwand zur Erstellung des Product Backlog und der Bestand an vorgehaltener Information minimiert. Dies bedingt, dass Anforderungen über die gesamte Projektdauer verfeinert werden: Teil der Aufgaben des Product Owner ist die systematische Verfeinerung der im nächsten Sprint umzusetzenden Anforderungen, vgl. Abschnitt 6.3.2.

#### 4.2.4 Die Einträge sind abgeschätzt

Alle Einträge im Product Backlog sind abgeschätzt. Dies ermöglicht dem Product Owner, eine Kosten-Nutzen-Betrachtung für die Anforderungen durchzuführen. In Scrum werden Schätzgrößen wie Punkte (*story points*) eingesetzt, um die Auf-

#### 4.3 Das Produktkonzept

wände zur Umsetzung von Anforderungen zu bestimmen, siehe Abschnitt 5.5. Das Team, das für die Umsetzung der Einträge verantwortlich ist, schätzt dabei stets die Anforderungen ab.

#### 4.2.5 Die Form des Product Backlog

Das Product Backlog kann in Form von Karteikarten oder elektronisch als Tabelle vorgehalten werden. Da das Release-Reporting in Scrum auf dem Product Backlog basiert, verwenden Scrum-Projekte häufig ein elektronisches Backlog. Tabelle 4–1 zeigt ein Beispiel für ein Product Backlog in tabellarischer Form.

Priorität	Thema	Beschreibung	Akzeptanzkriterien	Aufwand
1	Kalender	Als Basisanwender möchte ich eine Besprechung anlegen.	Teste das Eingeben ungültiger Werte, z.B. Endzeit liegt vor Startzeit.	1
2	Kalender	Als Basisanwender möchte ich eine Besprechung stornieren.	Teste, dass dieselbe Besprechung nicht zweimal gelöscht werden kann.	3
3	Kalender	Als Basisanwender möchte ich eine Besprechung ändern.	Teste, ob die Änderungenpersistiert wurden.	2
...				

Tab. 4–1 Das Product Backlog als Tabelle

Ich erfasse Anforderungen zunächst bevorzugt auf Karteikarten, insbesondere wenn Anforderungen in Workshops mit Interessenvertretern, Endkunden und dem Team ermittelt werden. Dies ermöglicht allen Beteiligten, aktiv an der Anforderungsbeschreibung mitzuwirken. Sind die wichtigsten Anforderungen erfasst, so überführe ich diese in eine elektronische Form.

Unabhängig von der Form Ihres Product Backlog sollten Sie seine Struktur so einfach wie möglich halten. Tabellarische Product Backlogs, die ein mehrfaches Scrollen nach rechts notwendig machen, enthalten oft zu viele Informationen. Beginnen Sie mit den in Tabelle 4–1 aufgeführten Spalten. Zusätzliche Spalten können Sie bei Bedarf zu einem späteren Zeitpunkt immer noch hinzufügen.

#### 4.3 Das Produktkonzept

##### 4.3.1 Von der Produktidee zum Product Backlog

Stellen Sie sich vor: Sie haben eine fantastische Idee für ein neues Produkt, die Sie mithilfe von Scrum realisieren wollen. Wie aber erstellen und füllen wir nun am besten das Product Backlog? Und wie können wir wirtschaftliche Ziele für die

Produktentwicklung frühzeitig formulieren? Insbesondere für innovative und komplexe Softwaresysteme ist es häufig nicht zielführend, sofort das Product Backlog aufzufüllen. Denn dies führt oft dazu, dass wir uns in der Menge der Anforderungen verlieren und wenig zielgerichtet Anforderungen aufnehmen und verfeinern. Das Product Backlog wird so unnötig aufgebläht.

Anstatt dessen ist es sinnvoll, zunächst ein Produktkonzept zu erstellen. Das Produktkonzept leitet sich aus der Idee für das Produkt bzw. die neue Softwareversion ab und ist die Grundlage für die Erstellung des Product Backlog, vgl. [Morgan&Liker 2006]. Dies veranschaulicht Abbildung 4–3.



**Abb. 4–3** Von der Produktidee zum Product Backlog

Das Produktkonzept beschreibt das zu befriedigende Kundenbedürfnis zusammen mit wesentlichen Leistungsmerkmalen und Zielgrößen der Software bzw. des Produkts. Diese beinhalten Alleinstellungsmerkmale und wichtige Funktionen. Das Produktkonzept beschreibt den Mehrwert, den das Produkt für Kunden und Anwender kreiert. Darüber hinaus kann es sinnvoll sein, zusätzlich eine Produkt-Roadmap zu erstellen, die zeigt, wie sich das Produkt über verschiedene Versionen und Releases hinweg entwickelt und sich in das entsprechende Produktpotfolio einbettet.

Der Product Owner erstellt das Produktkonzept unter Einbezug von Team, Kunden und Interessenvertretern wie Marketing, Service und Vertrieb, vgl. [Morgan&Liker 2006]. Er ist somit in die notwendigen Innovations- und Marktforschungsaktivitäten frühzeitig involviert. So wird eine Kontinuität vom Konzept bis zum fertigen Produkt erzielt. Übergaben und Informationsverluste sowie Wartezeiten, die bei einer strikten Trennung von Innovations- und Produktmanagementaktivitäten auftreten, werden vermieden.

#### Das Produktkonzept in der schlanken Produktentwicklung

Sobald eine Produktidee als attraktiv bewertet wurde, wählt Toyotas Management einen geeigneten Chief Engineer aus und beauftragt diesen mit der Erstellung des Produktkonzepts [Morgan&Liker 2006]. Bestandteil von Toyotas Produktkonzept sind übergreifende *product performance benchmarks* [Morgan&Liker 2006]. Diese fassen die wesentlichen Eigenschaften des zu entwickelnden Produkts zusammen [Reinertsen 1997]. Der Chief Engineer des ersten Lexus definierte beispielsweise als zentrale Eigenschaften des Fahrzeugs: »1. außergewöhnliche funktionale Performance (Leistung, Emotion, Geräuschpegel, Aerodynamik) und 2. elegantes Erscheinungsbild« [Morgan&Liker 2006, S. 124]. Der erste Lexus setzte die Vorgaben konsequent um und wurde ein durchschlagender Erfolg auf dem US-Markt.

#### 4.3.2 Qualitative Marktforschung

Für Toyotas Chief Engineer ist qualitative Marktforschung ein Muss. Dies kann das zeitweilige Einziehen bei einer Familie in Kalifornien oder ausgiebige Testfahrten inklusive Gespräch mit potenziellen Kunden in Vertriebsfilialen beinhalten [Liker 2003]. Quantitative Marktforschung wie die Erhebung von Kundenbedürfnissen in Form von Fragebögen und deren Auswertung mithilfe statistischer Verfahren hat auch in Scrum nach wie vor ihre Berechtigung. Als Product Owner sollten Sie jedoch stets auch den direkten Kontakt zu den Endkunden suchen, um deren Bedürfnisse möglichst genau zu verstehen.

Hierbei können Ihnen Innovationsspiele (*innovation games*) helfen [Hohmann 2007]. Innovationsspiele sind leicht anzuwendende kollaborative Techniken der qualitativen Marktforschung. Sie ermöglichen, spielerisch Produktideen und Anforderungen zu generieren und dabei den Endkunden eng einzubeziehen. *Prune the Product Tree* erlaubt beispielsweise, das Produkt an die Marktbedürfnisse anzupassen. *Spider Web* erlaubt dem Product Owner, die Beziehung des zu entwickelnden Produkts zu anderen Produkten der Kunden zu verstehen. Innovationsspiele generieren nicht nur wichtige Informationen, sondern machen auch noch Spaß!

#### 4.3.3 Nutzen des Produktkonzepts

Das Produktkonzept ist kein verpflichtender Bestandteil von Scrum. Es bietet jedoch folgende Vorteile:

- Es erlaubt die Eruierung des Marktpotenzials, die Formulierung wirtschaftlicher Ziele und ermöglicht so die Entscheidung über eine Durchführung des Scrum-Projekts.

■ Es bildet die Grundlage für die zielgerichtete und effiziente Erstellung des Product Backlog.

■ Es ermöglicht eine einheitliche Ausrichtung aller Beteiligten und Interessenvertreter inklusive Management, Product Owner, Team, Marketing, Vertrieb, Service und Endkunden.

#### 4.3.4 Kurz und knapp

Halten Sie das Produktkonzept bewusst kurz und knapp und achten Sie darauf, dass es leicht verständlich ist. Fokussieren Sie sich auf den Mehrwert und die Merkmale, die den wirtschaftlichen Erfolg des Produkts entscheidend beeinflussen. Widerstehen Sie der Versuchung, zu viele Informationen und zu viel Detail in das Konzept aufzunehmen. Meist ist es schwerer, die wichtigsten Produkteigenschaften kompakt zu beschreiben, als detaillierte, längere Spezifikationen zu erstellen. Ohne eine Fokussierung auf das Wesentliche laufen wir aber Gefahr, den Wald vor lauter Bäumen nicht mehr zu sehen. Die einheitliche Ausrichtung der Beteiligten geht so verloren. [Reinertsen 1997, S. 174 ff.] bemerkt hierzu treffend: »Manch einer behauptet, dass das Kaufmotiv für ein Produkt viel zu komplex sei, um es in ein oder zwei Sätzen beschreiben zu können. Unsere langjährige Marktforschungserfahrung zeigt jedoch genau das Gegenteil: Erfolgreiche Produkte besitzen meist einen eindeutigen und klaren Mehrwert.«

### 4.4 Inkrementelle Innovation

#### 4.4.1 Begriffsklärung

Scrum eignet sich ebenso für kurze Projekte, die wenige Wochen dauern, wie für Projekte, die erst nach mehr als einem Jahr Software ausliefern. Die Entscheidung, wie viel Funktionalität eine Softwareversion realisiert und wie lange ihre Entwicklung dauern darf, bleibt dabei dem Product Owner überlassen. Nichtsdestotrotz ist es empfehlenswert, die Anforderungen an eine Softwareversion möglichst minimal zu halten und so schnell wie möglich und häufig neue Funktionalität auszuliefern. Diese Vorgehensweise wird als inkrementelle Innovation bezeichnet, da Produktneuerungen schrittweise in den Markt eingeführt werden [Smith&Reinertsen 1998]. So werden »Big Bang«-Releases vermieden, die dem Anwender umfangreiche Funktionalität auf einen Schlag zur Verfügung stellen und lange Entwicklungszeiten bedingen. Da jeder Sprint in Scrum ein Produktinkrement erzeugt, macht Scrum die Anwendung von inkrementeller Innovation besonders leicht.

#### 4.4.2 Vorgehensweise

Um inkrementelle Innovation anzuwenden, wird für jede Softwareversion die kleinste Menge an vermarktbaren Merkmalen identifiziert, also der minimale Funktionalitätsumfang, der einen echten Mehrwert für Endkunden darstellt [Denne&Cleland-Huang 2004]. Google führt neue Produkte beispielsweise regelmäßig inkrementell in den Markt ein: Zunächst wird eine Betaversion den Endanwendern zur Verfügung gestellt. Diese wird dann je nach Anwenderfeedback rasch weiterentwickelt oder vom Markt genommen. Inkrementelle Innovation ist nicht nur für reine Softwareprodukte bzw. -lösungen möglich und vorteilhaft. [Smith&Reinertsen 1998] dokumentieren Produktentwicklungen mit Schwerpunkt auf Hardware- und Mechanikentwicklungen, die inkrementelle Innovation erfolgreich umgesetzt haben.

#### 4.4.3 Vorteile

Inkrementelle Innovation bietet folgende Vorteile, vgl. [Smith&Reinertsen 1998]:

- Reduzierung der Projektdauer und *time to market*: Funktionalität wird Endanwendern schneller zur Verfügung gestellt.
- Schnellerer *break-even* und verbesserter *cash flow*: Die Software wird schneller an Endanwender ausgeliefert, die Kunden bezahlen für die Software früher.
- Risikominimierung durch die Reduzierung der notwendigen Investition für eine Softwareversion. Dabei werden das investierte Geld und die investierte Zeit pro Softwareversion reduziert.
- Höhere Profitabilität (*return on investment, ROI*): Produkte, die mithilfe inkrementeller Innovation entwickelt und vermarktet werden, können einen wesentlichen höheren ROI aufweisen [Denne&Cleland-Huang 2004].
- Bessere Vorhersage der Kundenbedürfnisse und schnellere Rückmeldung über den Einsatz der Software beim Endanwender: Auf diese Weise kann auf Marktveränderungen schneller reagiert werden. Zudem fällt es leichter, die Anforderungen für eine Version stabil und das Product Backlog überschaubar zu halten.
- Kurze Releasezyklen erleichtern es, termingerecht zu liefern und im Notfall niederpriore Anforderungen auf die nächste Version zu schieben.

[Poppdieck 2003 und 2006] empfehlen übrigens allgemein, Projektdauer und Releasezyklen zu reduzieren (*deliver fast*), um die Wertschöpfung zu optimieren. Wie häufig neue Software in den Markt eingeführt wird und wie kurz die Entwicklungszyklen sind, hängt letztendlich von der Fähigkeit der Kunden ab, die