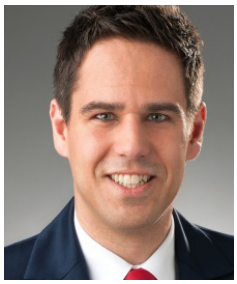

Philip Stolz
(philip.stolz@hood-group.com)
ist Senior Consultant bei der HOOD GmbH. Er arbeitet bei Agile-by-HOOD als Trainer, Coach und Berater mit Leidenschaft und viel persönlichem Engagement mit Menschen, Teams und Organisationen auf ihrem Weg zu mehr Agilität.

Bertil Muth
(bertil.muth@hood-group.com)
ist Senior Consultant bei der HOOD GmbH. Sein Schwerpunkt liegt in den Bereichen agiles Coaching und Consulting.

Markus Eberhardt
(markus.eberhardt@hood-group.com)
ist Senior Consultant bei der HOOD GmbH. Sein Schwerpunkt ist Beratung, Training und Coaching für Requirements-Engineering, sowohl im traditionellen als auch im agilen Umfeld.

# Vom „Big Picture“ zur Umsetzung

Populäre agile Frameworks wie Scrum setzen voraus, dass der Umgang mit Anforderungen beherrscht wird. Aber kann man das wirklich als gegeben voraussetzen? Was hilft in der Entwicklung konkret, damit ein Produkt entsteht, das dem Kunden gefällt – und man nicht einfach nur schneller als bisher entwickelt? Wie stellt man sicher, dass der Blick auf das „große Ganze“ trotz vieler kleiner User Stories und häufiger Änderungen nicht verloren geht – ohne am Anfang der Entwicklung eine umfangreiche Spezifikation zu schreiben? Dieser Artikel demonstriert an einem Beispiel, wie der erfolgreiche Umgang mit Anforderungen in einem agilen Umfeld aussehen kann.

## Fiktives Beispiel: Streamingportal

Als fiktives Beispiel in diesem Artikel dient das Unternehmen Guillemets Cash (im Nachfolgenden «$» genannt). «$» ist ein Anbieter von Finanzdienstleistungen. Das Unternehmen will zukünftig in den Bereich Internetmarketing einsteigen.

«$» hat bereits mehrere Versuche unternommen, im Bereich Internetmarketing Fuß zu fassen. Obwohl in der Vergangenheit genug Fachexpertise im Unternehmen vorhanden war, konnte nie zum richtigen Zeitpunkt ein lauffähiges System präsentiert werden, das dem Stand der Technik entsprochen hätte.

Dem Management von «$» ist heute klar: Um mit dem Thema Internetmarketing erfolgreich zu sein, muss es neue Wege beschreiten. Ein neues Produkt soll den Weg ebnen – ein Streamingportal für Musikstücke und Filme, in die Werbung direkt integriert ist. Dieses neue Produkt soll mit Scrum entwickelt werden [SuSch13].

## Die Produktvision: Wo geht die Reise hin?

Damit allen Beteiligten klar ist, in welche Richtung die Entwicklung gehen soll, entschließt sich der Product Owner, eine gemeinsame Vision des neuen Produktes im Unternehmen zu etablieren. Zu diesem Zweck kommt das Product Vision Board von Roman Pichler [Pich10] zum Einsatz, ein Format zur Beschreibung und Visualisierung einer Produktvision (siehe Abbildung 1).

## Der Scope des Systems: Use-Case-Modell als „Big Picture“

Nachdem sich die Beteiligten auf die Produktvision geeinigt haben, beginnt das Produktmanagement gemeinsam mit Fachexperten damit, den Scope des Systems [Hood08] zu erarbeiten. Es stellt sich die Frage, welche Funktionalitäten die Soft-

| VISION STATEMENT | Nur das, was Sie wirklich wollen! | | |
|---|---|---|---|
| ZIELGRUPPE | BEDÜRFNISSE | PRODUKT | WERT |
| Produkt- / Dienstleitungs-anbieter | Konsumenten zum Kauf bringen | Streaming Portal für Musik & Videos | große Anzahl an Anbietern |
| Konsumenten jeder Altersgruppe | Musik & Videos Kostenlos konsumieren | Personalisierte Werbung | große Anzahl an Konsumenten |
| | Keine Zeit für Werbung verschwenden | Integrierte Werbung in Musik und Videos | hoher Konsum an Werbung |

*Abb. 1: Die Produktvision*

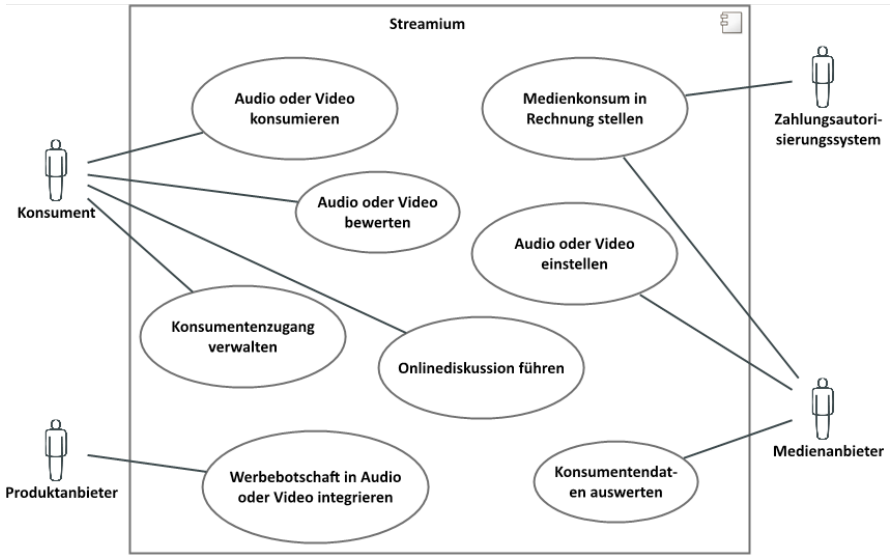


*Abb. 2: Use-Case-Modell als „Big Picture"*

ware bieten soll, die den Zielgruppen einen Mehrwert liefert.

Dazu erstellt der Product Owner zusammen mit weiteren Stakeholdern aus dem Unternehmen in einem Workshop ein High-Level-Use-Case-Modell ([PoRu11], siehe Abbildung 2). Dieses initiale Modell zeigt grafisch die Nutzerrollen beziehungsweise externen Systeme als Akteure, die bestimmte Systemfunktionalitäten verwenden.

## Vom Big Picture zur Releaseplanung: Leichtgewichtige Beschreibung der Use Cases

Dem Product Owner ist klar, dass sich in der Regel nicht alle Use Cases aus dem Use-Case-Modell vollständig in einem ersten Release umsetzen lassen. Zu groß sind zu diesem Zeitpunkt die Unklarheiten, sowohl was die Anforderungsdetails als auch die technische Umsetzung betrifft.

Nach mehreren Gesprächen mit den Stakeholdern entscheidet sich der Product Owner für folgendes Vorgehen bei der Releaseplanung [Jac11]: Für jeden der Use Cases wird zunächst ein mit den Fachexperten und Entwicklern abgestimmter „Regulärer Ablauf" entwickelt, das heißt, ein Durchlauf durch den Use Case, der dem Nutzer das gewünschte Ergebnis bringt.

Dieser „Reguläre Ablauf" wird nur stichpunktartig dokumentiert, denn viele Details sind noch unklar oder könnten sich noch während der Entwicklung ändern. Die alternativen Abläufe werden bereits genannt, aber nicht ausformuliert (siehe Abbildung 3).

| Audio oder Video konsumieren | Audio oder Video einstellen |
|---|---|
| Kurzbeschreibung:<br>Als Konsument möchte ich jederzeit beliebige Musikstücke und Videos abspielen können, damit ich das konsumieren kann, worauf ich gerade Lust habe. | Kurzbeschreibung:<br>Als Medienanbieter möchte ich Musikstücke oder Videos online einstellen können, damit ich eine breitere Masse an Konsumenten erreiche. |
| Regulärer Ablauf:<br>1. Nutzer identifizieren<br>2. Musikstücke/Videos durchstöbern<br>3. Musikstück oder Video abspielen | Regulärer Ablauf:<br>1. Nutzer identifizieren<br>2. Musikstück oder Video hochladen<br>3. Musikstück oder Video attributieren<br>4. Freigabe durch «$» |
| Alternative Abläufe:<br>A1 Stichwortsuche<br>A2 Audio/Video vorschlagen<br>A3 Audio/Video zufällig | Alternative Abläufe:<br>A1 Automatische Attributierung über verfügbare Online-Informationen<br>A2 Freigabe durch Peer-Review |

*Abb. 3: Stichpunktartige Beschreibung der Use Cases*

## Von der Releaseplanung zur Entwicklungsplanung: Story Mapping

Das Scrum Team hat sich darauf geeinigt, in Zwei-Wochen-Sprints zu entwickeln. Das heißt: Alle zwei Wochen muss eine lauffähige, getestete Software zu Verfügung stehen. Nach sechs dieser Sprints soll dann das erste Release zur Verfügung stehen.

Schnell wird klar: Jeden Use Case in zwei Wochen fertig zu bekommen, wird unter den gegebenen Rahmenbedingungen nicht möglich sein. Die Use Cases müssen also so „dünn" geschnitten werden, dass das Entwicklungsteam innerhalb eines Sprints etwas umsetzen kann, was den Zielgruppen einen Wert liefert. Das Scrum Team trifft sich und erstellt eine sogenannte Story Map [Patt14] für die Use Cases (siehe Abbildung 4).

Die oberste Zeile der Story Map gibt die groben „Aktivitäten des Users" an. Hier hat der Product Owner der Einfachheit halber die Use Cases eingetragen. Jede umrahmte Zeile der Story Map definiert einen sinnvollen Durchlauf durch den gesamten Geschäftsprozess. Jede grüne Karte repräsentiert eine User Story. Je weiter unten sich eine der grünen Karten befindet, desto geringer ist die Dringlichkeit ihrer Umsetzung – sprich: Was oben steht, wird zuerst umgesetzt.

Bevor das Entwicklungsteam Funktionalitäten wie das „Einsprechen" von Werbeslogans während des Abspielens von Liedern entwickelt, muss erst das Fundament stehen: Das Abspielen von Medien muss funktionieren, einschließlich der Integration von Werbebotschaften. Diese grundlegende Funktionalität wird auch als „Walking Skeleton" [Cock] bezeichnet: Die Software läuft, aber es ist noch kein „Fleisch" daran.

Mit dem „Walking Skeleton" wird für die Stakeholder erstmals Nutzen erkennbar und bewertbar und es wird Vertrauen aufgebaut, dass das Entwicklungsteam die benötigten technischen Aspekte der Produktentwicklung ausreichend beherrscht, um künftig kontinuierlich wertvolle Software zu liefern.

## Das Product Backlog: Priorisierung der Entwicklungstätigkeiten

Das Product Backlog in Scrum ist eine geordnete Liste. In unserem Beispiel enthält das Product Backlog die User Stories, die während des Story Mappings identifiziert wurden. Um von der zweidimensionalen Story Map auf das eindimensionale Back-

log zu kommen, werden die Stories entsprechend der geplanten Implementierungsreihenfolge sortiert. Diese Priorisierung ist in Abbildung 4 in Form einer Durchnummerierung dargestellt.

## Backlog Refinement: Die kontinuierliche Pflege des Backlogs im Scrum Team

Die geordneten User Stories alleine reichen nicht, um die Software zu entwickeln. Die User Stories dienen nicht der Dokumentation, sondern der Kommunikation. Die Details werden vom Scrum Team geklärt, und zwar kurz bevor es an die Umsetzung der jeweiligen User Story geht. Ein solches Detail könnten im Beispiel die Festlegung von Akzeptanzkriterien sein (siehe Abbildung 5). In der Regel werden solche Detailfragen maximal ein bis zwei Sprints im Voraus geklärt, bei Zwei-Wochen-Sprints also maximal zwei bis drei Wochen im Voraus.

Außerdem muss spätestens zu Beginn jedes Sprints klar sein, wie viele User Stories voraussichtlich umgesetzt werden können. Um nicht auf den letzten Drücker die Details klären zu müssen, ist die Klärung der Details und die Aufwandsschätzung etwas, das kontinuierlich während der Sprints stattfindet – im sogenannten Backlog Refinement. Das Backlog Refinement sollte bis zu 10 Prozent der Zeit im Sprint einnehmen und kann zum Beispiel in einer fest reservierten Zeit, der „Story Time", umgesetzt werden.

## Anforderungen im Sprint Planning

Im Sprint Planning wird das „Was" und „Wie" für den kommenden Sprint festgelegt, also was im Sprint umgesetzt werden und wie die Umsetzung erfolgen kann. Durch das Sprint Planning soll ein gemeinsames Verständnis der Anforderungen sichergestellt werden. Bei der Diskussion der möglichen Umsetzung werden zudem die Akzeptanzkriterien aktualisiert. Nicht selten erwachsen aus diesen Diskussionen neue Anforderungen, die bisher nicht bedacht worden sind, die aber wesentlich für den Erfolg des Produktes sind (siehe Abbildung 6).

## Dokumentieren während des Sprints

Die Dokumentation des Produktes ist auch im agilen Umfeld eine Notwendigkeit. Deshalb sollte die erforderliche Produktdokumentation jeweils im Sprint für das entwickelte Produkt-Inkrement erfol-

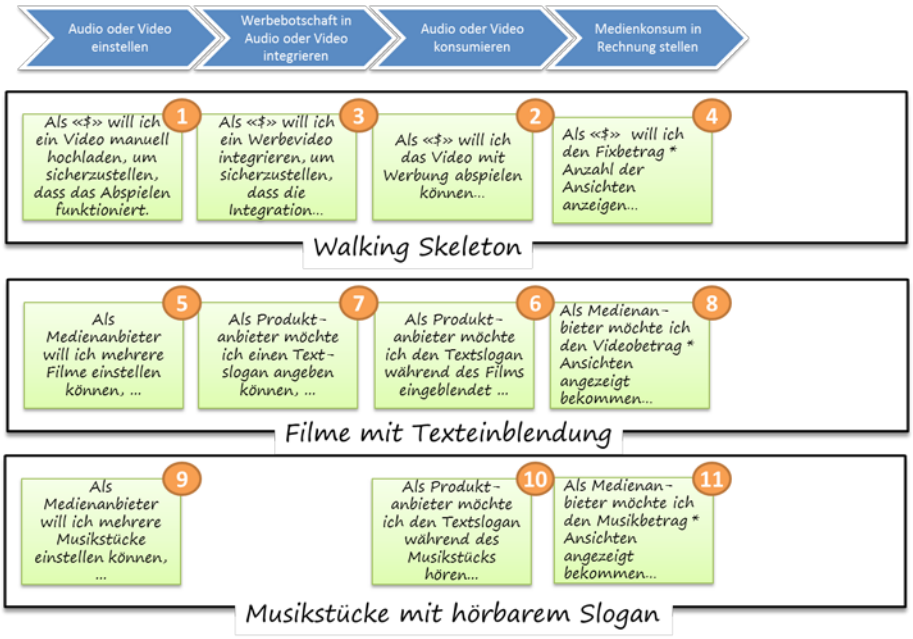


*Abb. 4: Exemplarischer Ausschnitt einer Story Map*

Als Medienanbieter will ich mehrere Filme einstellen können, damit die Konsumenten möglichst viele meiner Filme konsumieren.

**Akzeptanzkriterien:**

- Ein Medienanbieter kann nacheinander verschiedene Filme einstellen.
- Zu jedem Film kann der Medienanbieter folgende Daten angeben:
  - Titel (verpflichtend)
  - Erscheinungsjahr (verpflichtend)
  - Regisseur (verpflichtend)
  - Drehbuchautor (optional)

*Abb. 5: Details aus der Story Time*

Als Medienanbieter will ich mehrere Filme einstellen können, damit die Konsumenten möglichst viele meiner Filme konsumieren.

**Fragestellung aus dem Sprint Planning:**
Wer hostet die Filme? «$» oder der Medienanbieter?

**Neues Akzeptanzkriterium**
Der Benutzer darf nicht länger als 3 Minuten mit dem System beschäftigt sein, um einen Film einzustellen. D. h. nach spätestens 3 Minuten muss er einen weiteren Film einstellen können.

*Abb. 6: Ergänzung von Akzeptanzkriterien im Sprint Planning*

Als Medienanbieter will ich mehrere Filme einstellen können, damit die Konsumenten möglichst viele meiner Filme konsumieren.

**Akzeptanzkriterium:**
ANGENOMMEN
ich bin als Medienanbieter „Mediaprov" angemeldet
UND
in meiner Filmliste befindet sich der Titel „Dr. Mabuse, der Spieler"
WENN
ich die Menüoption „Film einstellen" wähle
UND
ich als MedienURL „http://guillemetcash/2001.wmv" angebe
UND
ich folgende Attributierung wähle: Titel=„2001: Odyssee im Weltraum" Erscheinungsjahr=„1968" Regisseur=„Stanley Kubrick"
UND
auf den Knopf „einstellen" drücke
DANN
bekomme ich die Meldung „Es befinden sich 2 Filme in Ihrer Filmliste"
UND
die Filmliste enthält die folgenden Titel: „Dr. Mabuse, der Spieler", „2001: Odyssee im Weltraum"

*Abb. 7: Akzeptanzkriterienbeispiel in der Beschreibungssprache Gherkin*

gen. Aus Sicht des Requirements-Engineerings sollte das Entwicklungsteam im Sprint mindestens die Testfälle zur Überprüfung der umgesetzten Anforderungen dokumentieren.

Aus diesem Grunde hat sich das Scrum Team bei «$» nach einigen Sprints dazu entschieden, den Ansatz „Specification by Example" zu nutzen [Adz11]. Das heißt, das Scrum Team notiert Akzeptanzkriterien der User Stories in einer Form, die es ermöglicht, diese per Automatismus in ausführbare Testfälle zu verwandeln (siehe Abbildung 7).

### Review: Das Event zur Erhebung von Anforderungen

Im Review stellt das Entwicklungsteam das Produkt-Inkrement dem Product Owner sowie weiteren Stakeholdern vor. Diese Vorstellung erlaubt es dem Scrum Team, kontinuierlich zu prüfen, ob es die Anforderungen der Stakeholder verstanden hat, und ein Produkt entsteht, das den Stakeholdern gefällt. Die Stakeholder begutachten, probieren und kommentieren das Produkt-Inkrement. Hier gilt es für das Scrum Team aufmerksam zu beobachten, wie die Stakeholder mit dem Produkt-Inkrement umgehen und welche Kommentare fallen. In diesem Kontext kann das Scrum Team auf Erhebungstechniken zurückgreifen, die aus dem Requirements-Engineering bekannt sind.

Der Product Owner von «$» lädt zu jedem Review Medienanbieter, Konsumenten und Produktanbieter ein. Das Scrum Team beobachtet im Review die beteiligten Stakeholder und kann dadurch nicht explizit formulierte Anforderungen erkennen. In unserem Beispiel konnte das Scrum Team frühzeitig herausfinden, dass Werbeeinblendungen in einem Film weitaus häufiger möglich sind, als es ursprünglich angenommen hat, bevor diese auf den Konsumenten aufdringlich wirken.

### Zusammenfassung

Das vorgestellte Beispiel bindet Praktiken des Requirements-Engineerings in das Scrum-Prozessrahmenwerk ein. Die Stärken des Requirements-Engineerings, wie zum Beispiel das Schaffen eines „Big Pictures" oder das Erheben von Anforderungen durch Beobachtung, kommen gut zur Geltung und unterstützen das agile Prozessrahmenwerk. Zusammenfassen lässt sich das mit der Aussage: „Bewährte Praktiken des Requirements-Engineerings sind auch im agilen Umfeld wertvoll – solange sich diese an den agilen Werten und Prinzipien orientieren".

Requirements-Engineering ist demnach keiner bestimmten Projektphase zuzuordnen, sondern findet kontinuierlich statt und wird von allen Beteiligten angewandt.

Mehr zu diesem Thema finden Sie unter http://www.hood-group.com/agile/training/certified-agile-requirements-specialist-cars/. ■

### Literatur & Links

[Adz11] G. Adzic, Specification by Example: How Successful Teams Deliver the Right Software, Manning, 2011

[Cock] A. Cockburn, Walking Skeleton, siehe http://a.cockburn.us/2102

[Hood] HOOD GmbH, Agile Requirements Cheat Sheet, 2015

[Hood08] C. Hood, et. al., Requirements Management: The Interfaces between Requirements Development and all other Systems Engineering Processes, Springer-Verlag, 2008

[Jac11] I. Jacobson, et. al., Use-Case 2.0: The Guide to Succeeding with Use Cases, eBook, Dezember 2011, siehe www.ivarjacobson.com/download.ashx?id=1282

[Patt14] J. Patton, User Story Mapping: Discover the Whole Story, Built the Right Product, O'Reilly, 2014

[Pich10] R. Pichler, Agile Product Management with Scrum, Addison-Wesley Professional, 2010

[PoRu11] K. Pohl, Ch. Rupp, Basiswissen Requirements Engineering, dpunkt.verlag, 2011

[SuSch13] J. Sutherland, K. Schwaber, Der Scrum Guide - Der gültige Leitfaden für Scrum: Die Spielregeln, Juli 2013, siehe http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-DE.pdf