Kapitel 5
Scrum

Im Gegensatz zu den bisher diskutierten agilen Prozessmodellen ist Scrum eine agile Projekt*management*methode. Nicht die Zusammenarbeit des Teams, sondern der *Ablauf des Projekts* steht im Vordergrund. Auffallend sind die in Scrum festgelegten Projektrollen Product Owner, Team und Scrum Master, insbesondere das Fehlen einer „klassischen" Rolle, der des Projektmanagers. Neben den Aufgaben der Projektrollen regelt Scrum den Projektablauf in Form von *Sprints*, also Entwicklungszyklen von wenigen Wochen.

Scrum unterscheidet sich von den bisher diskutierten agilen Prozessmodellen, denn es ist eine agile Projekt*management*methode. Fragen der Zusammenarbeit und der Teamstruktur stehen nicht im Vordergrund, sondern der Ablauf des Projekts. Trotzdem ist es eine sehr interessante agile Methode mit zunehmender Verbreitung. Scrum folgt der Idee des *Lean Managements*, des „schlanken" Managements, wie es insbesondere bei Toyota verfolgt wird[74] (Gloger 2009).

Scrum wurde in den frühen 1990er Jahren von Ken Schwaber und Jeff Sutherland entwickelt, die Grundgedanken sind in (Schwaber 2004) niedergelegt. Auffallend sind die in Scrum festgelegten Projektrollen

- Product Owner
- Team
- ScrumMaster

und damit insbesondere das Fehlen einer „klassischen" Rolle, der des Projektmanagers. Neben den Aufgaben der Projektrollen regelt Scrum den Projektablauf in Form von *Sprints* – Entwicklungszyklen von wenigen Wochen.

5.1 Scrum – die Projektrollen

Im Folgenden sollen die in Scrum zwingend geforderten Projektrollen näher beleuchtet werden.

[74] auch wenn Ken Schwaber immer wieder betont, dass Scrum seinen Ursprung *nicht* in der Lean Production hat

E. Hanser, *Agile Prozesse: Von XP über Scrum bis MAP*, eXamen.press,
DOI 10.1007/978-3-642-12313-9_5, © Springer-Verlag Berlin Heidelberg 2010

5.1.1 Product Owner

Der *Produktverantwortliche* übernimmt die Sichtweise des Endkunden. Er steuert die Software-Entwicklung und arbeitet eng mit dem Entwicklungsteam zusammen. Ähnlich wie der Kunde in XP (Kap. 3) ist er für das Entwicklungsteam ständig verfügbar. Im Gegensatz zum XP-Kunden bestimmt er im Projekt alleine „wohin die Reise geht“[75] (Pichler 2008).

Die Rolle des Product Owners ist bivalent. Er darf nicht die „Unschuld“ des Kunden annehmen und sich neue Features wünschen, sondern wird auch sofort auf die Realisierungsmöglichkeit achten. Es besteht somit die Gefahr, dass tolle Ideen nicht realisiert werden, weil der Product Owner nicht an ihre Realisierungsmöglichkeit glaubt.

Im Einzelnen hat der Product Owner folgende Aufgaben:

5.1.1.1 Anforderungsmanagement

Der Product Owner erfasst die Anforderungen des Kunden an die Software und beschreibt sie in Form eines Produktkonzepts[76] und des *Product Backlogs.*[77] Ein traditionelles Scrum Product Backlog enthält alle Arten von Anforderungen oder Tätigkeiten, die im Zusammenhang mit dem Projekt erledigt werden müssen.[78] Ziel ist immer, einen Mehrwert für den Kunden zu erzeugen. Der Product Owner priorisiert und schätzt die Anforderungen als Basis der Arbeit des Teams. Das Product Backlog „lebt“. Im Verlauf des agilen Prozesses wird es ständig aktualisiert und erweitert.

5.1.1.2 Release Management

Der Product Owner ist verantwortlich für den Erfolg des Projekts. Er alleine entscheidet über Funktionalität, Kosten und Termine. Somit ist er auch verantwortlich für die Auslieferung der diversen Releases.[79] Er erstellt also einen Release Plan und passt ihn ggf. an. Der Product Owner ist gut beraten, dies in Absprache mit dem Team zu machen. Die Aufwandschätzungen aus dem Team sind die Basis für seine Planung.

---

[75] und wird damit zum „single wringable neck“, wie diese Rolle bei Yahoo! bezeichnet wird (Pichler 2008)

[76] Beantwortung von Fragen wie: Was wollen wir als Firma mit dem Produkt erreichen? Was ist der Zweck des Produkts?

[77] *engl.* Auftragsbestand

[78] Für den Product Owner ist es am einfachsten, sich zuerst auf Anforderungen zu konzentrieren, die einen Mehrwert für den Kunden erzeugen. Diese Anforderungen sind aus (Kap. 3.3.1) als *User Stories* (XP) bekannt, siehe dazu auch (Wirdemann 2009).

[79] Versionen

5.1.1.3 Kommunikation

Der Product Owner muss beständig mit dem Team und mit allen beteiligten Interessengruppen (Stakeholdern), insbesondere beim Endkunden, kommunizieren. Da der Product Owner eng mit dem Team zusammenarbeiten muss, kann man aus der Sicht des Teams von Innen- und Außenwirkung sprechen. In der Innenwirkung steht der Product Owner dem Team idealerweise pro Tag eine entsprechende Zeitdauer zur Verfügung (mindestens eine Stunde) während er den Rest des Tages zumindest gut erreichbar ist.[80] In der Außenwirkung kommuniziert der Product Owner insbesondere mit den Endkundengruppen, um ein möglichst frühes Feedback über neue Funktionalität der Software zu bekommen.

Es ist wichtig, den Product Owner mit genügend Vollmachten und Zeit auszustatten, um die o.g. Aufgaben zu erfüllen. Product Owner ist ein Vollzeit-Job[81]! Nur so kann ein Erfolg des Projekts ermöglicht werden.

5.1.2 Team

Das Team entwickelt („liefert“) das Produkt, also die Software. Seine Zusammensetzung ist entscheidend für das Gelingen des Projekts. Dabei geht es nicht nur um die fachliche Qualifikation, sondern auch um die Teampsychologie, also das „Zusammenpassen“ der Teammitglieder. Dies soll in Kap. 6 und 7 näher beleuchtet werden. Wichtig ist, potenziellen Teammitgliedern die Möglichkeit zu geben, sich für interessante Projekte zu bewerben. Damit ist zumindest sichergestellt, dass Interesse vorhanden ist.[82]

Scrum geht nicht weiter auf die Rollenverteilung im Team ein. Scrum ist, wie schon erwähnt, eine Managementmethode. Es ist jedoch klar, dass alle Rollen oder besser gesagt *Fähigkeiten*, die für eine erfolgreiche Projektentwicklung notwendig sind, vorhanden sein müssen. Auch hier sei auf Kap. 6 und 7 verwiesen.

Ein Team muss folgende Eigenschaften haben:

5.1.2.1 Selbstorganisiert und klein

Innerhalb eines Sprints (also einer Iteration) organisiert sich das Team selbst. Ausschließlich die Teammitglieder entscheiden, welche Aufgaben gelöst werden müssen, um ein Sprint-Ziel zu erreichen, und wer für die Aufgabe verantwortlich zeichnet.[83] Das Team hat keinen Teamleiter, also auch keinen klassischen

---

[80] Aus der Erfahrung des Autors empfiehlt sich die Verfügbarkeit über ein „rotes Telefon“, ein Mobiltelefon ggf. mit Geheimnummer, welche nur dem Team bekannt ist.

[81] Sollte das nicht möglich sein, muss er wenigstens 50% seiner Zeit für das Projekt freigestellt werden (Pichler 2008).

[82] Google ermutigt Entwickler, sich für Projekte zu bewerben!

[83] Das Team ist *empowered* (bevollmächtigt).

Projektleiter. Dies hat zum einen mit der Größe des Teams zu tun: Ein ideales Scrum-Team hat sieben ± zwei Mitglieder (Schwaber 2004). Zum anderen kann gemäß Scrum auf den Teamleiter verzichtet werden, weil das Team seine tägliche Arbeit im Rahmen eines „Korsetts" von Berichten und Besprechungen durchführt und somit „Leitplanken" für die Projektdurchführung hat.[84] Bedingung für diese Arbeitsweise ist die Vollzeitmitgliedschaft der Mitglieder im Team. Agile Techniken erfordern ein Team, das sich als Einheit versteht. Dieses Teamverständnis kann sich nicht aufbauen, wenn einige Teammitglieder von Projekt zu Projekt „hüpfen".

Auch der Arbeitsplatz des Teams muss den Anforderungen genügen: Ein agiles Team muss *osmotisch* kommunizieren (Cockburn 2005). Dazu gehört räumliche Nähe. Das Team *muss* ein gemeinsames Büro haben[85] (Abb. 5.1).

Osmotische Kommunikation läuft in einer solchen räumlichen Situation typischerweise wie folgt ab:

- Ein Teammitglied holt sich einen Kaffee, verlässt seinen Arbeitsplatz und schlendert an anderen vorbei. Automatisch und ohne eigenes Zutun („osmotisch")

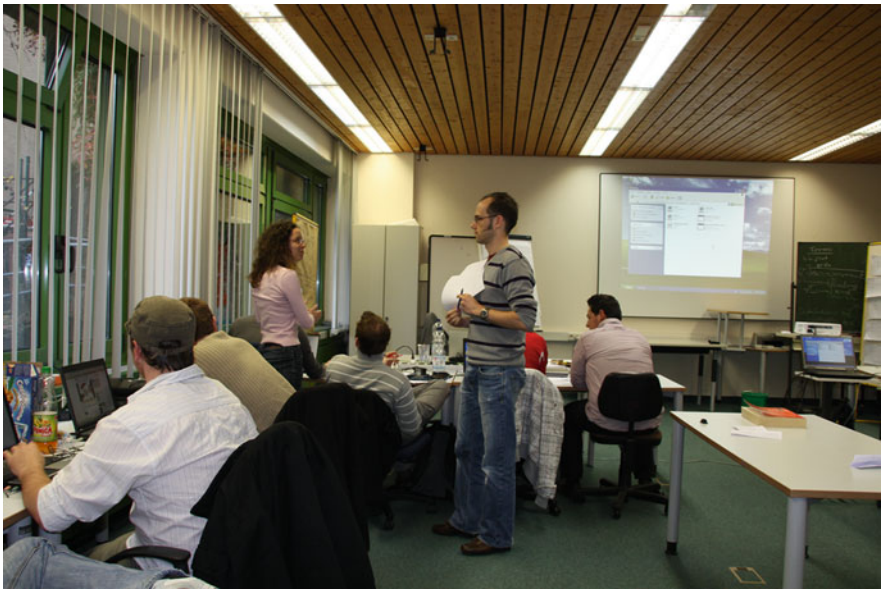

Abb. 5.1 Team im gemeinsamen Büro

[84] Es sind dies: Sprint-Backlog, Sprint-Burndown-Bericht und die Daily Scrum-Besprechung, siehe auch Kap. 5.2.

[85] Nach der Erfahrung des Autors ist diese Forderung nicht verhandelbar und kann auch nicht durch Videokonferenzen ersetzt werden. Der Verlust an osmotischer Kommunikation wäre einfach zu hoch.

schnappt es Gesprächsfetzen der anderen auf. Entweder sind diese uninteressant oder sie betreffen die Arbeit des Teammitglieds. Das Teammitglied schaltet sich in die Diskussion ein.
- Durch die räumliche Nähe hört man ohne Absicht Gesprächsfetzen der nebenan sitzenden Entwickler mit. Betrifft einem das Thema, schaltet man sich in die Diskussion ein.

Osmotische Kommunikation geschieht immer „im Vorbeigehen". Man glaubt nicht, wie viele wichtige Dinge man auf diese Weise erfährt!

Eine Anmerkung zu Großraumbüros: Der Teambereich sollte durch Stellwände abgegrenzt werden. Es empfiehlt sich, nicht mit dem Platz zu sparen. Nach den Stellwänden sollte nicht sofort der erste Schreibtisch folgen, sondern es sollte ausreichend „Luft" um den Teambereich herum sein. Ansonsten wird das Team ständig durch Passanten gestört, die *über* die Wände hinweg sprechen.

5.1.2.2 Multidisziplinär und autonom

Die Teammitglieder müssen alle Fähigkeiten abdecken, die zum Erreichen der Projektziele erforderlich sind. Software-Architekten müssen genauso vorhanden sein wie Datenbankspezialisten, Programmierer, Qualitätssicherer, Tester, Dokumentierer usw. Wichtig ist, dass diese Spezialisten nicht auf ihrer traditionellen Projektrolle beharren. Grundsätzlich muss jedes Teammitglied jede Aufgabe im Projekt übernehmen (können) und damit auch sein Wissen an die anderen weitergeben.

Das Scrum-Team muss unabhängig sein. Ist ein Sprint-Ziel erst einmal definiert, muss das Team in der Lage sein, das Ziel ohne externe Abhängigkeiten oder Hilfe von außen zu erreichen. Ist dies nicht gewährleistet, dann muss die Zusammensetzung des Teams verbessert werden, indem externer Sachverstand ins Team geholt wird.

5.1.3 ScrumMaster

Der ScrumMaster (*Change Agent*) ist der Prozessverantwortliche. Er ist der *Freund* des Teams. Er coacht das Team, wenn es um die Einhaltung des Prozesses geht. Insbesondere bei mit Scrum unerfahrenen Teams hilft er, den Prozess zu etablieren. Macht er seine Arbeit gut, macht er sich idealerweise überflüssig! Die Erfahrung zeigt jedoch, dass es nicht soweit kommt, da Teams dazu neigen, den gewählten Prozess (manchmal bis zur Unkenntlichkeit) abzuändern (was der Autor grundsätzlich nicht für beängstigend hält, siehe Kap. 6 und 7). Deswegen muss der ScrumMaster immer wieder eingreifen und die Korrektheit des Prozesses überprüfen. Bei erfahrenen Teams kann er allerdings die Zeit, die er mit dem Team verbringt,

langfristig reduzieren. Der Beziehung zwischen ScrumMaster und Team ist eine Eins-zu-Eins-Beziehung. Der ScrumMaster arbeitet beim Team.

Die Aufgaben des ScrumMasters sind:

5.1.3.1 Scrum einführen und Product Owner und Team unterstützen

Der ScrumMaster etabliert den Prozess im Team. Er identifiziert Hindernisse wie verkrustete Denkweisen oder „veraltete“ Arbeitsweisen. Er hilft dem Team, Scrum kennen zu lernen und damit Erfahrungen zu sammeln. Er erklärt dem Team die Abläufe im Prozess und unterstützt es dabei. Er beantwortet auftauchende Fragen zum Prozess. Wichtig ist es auch, das Team gegen Einflüsse von außen zu schützen. Keinesfalls dürfen Mitglieder von den Vorgesetzten für andere Aufgaben abgezogen werden (auch nicht temporär). Ziel ist, das Team in Ruhe arbeiten zu lassen.

Der ScrumMaster unterstützt auch den Product Owner. In der Regel ist auch dieser anfangs ungeübt und braucht Hilfe. Wichtig ist zu verhindern, dass der Product Owner in die Rolle des Projektleiters verfällt. (Diese Gefahr besteht allerdings auch für den ScrumMaster!) Der Product Owner muss zwar eng mit dem Team zusammenarbeiten, setzt das Team jedoch zu einem Scrum Sprint an, darf er es nicht mehr stören. Das muss der ScrumMaster verhindern.

5.1.3.2 Hindernisse beseitigen

Der ScrumMaster muss Abweichungen vom Prozess-Soll, die *Impediments* angehen (Gloger 2009). Impediments hindern das Team daran, effizient zu sein. Diese Blockaden müssen gelöst werden. Der ScrumMaster notiert sich alle Abweichungen in einer eigenen Liste, dem *Scrum Impediment Backlog*. Diese muss er nach und nach abarbeiten. In der Regel können komplexere Blockaden nicht in kurzer Zeit gelöst werden.

Impediments können Probleme in der Teampsychologie sein (wenn also z.B. Teammitglieder nicht miteinander auskommen), können sich aber auch aus falsch verstandenen Scrum-Rollen ergeben, z.B. dem Product Owner, der versucht als Projektleiter zu fungieren. Auch ein fehlender oder ungeeigneter Product Owner wird ganz schnell zu einem essenziellen Problem. Hier ist das Fingerspitzengefühl des ScrumMasters gefragt, die Situation zu entschärfen und einen geeigneten Product Owner zu finden.

5.1.3.3 Dem Team dienen

Der ScrumMaster führt das Team. Jeder Coach weiß, dass er alleine aufgrund seines Wissensvorsprungs eine Führungsfunktion hat. Diese darf jedoch nicht ausgenutzt werden. Der ScrumMaster sollte einen kollegialen Führungsstil pflegen. Seine Aufgabe ist es, dem Team Hindernisse aus dem Weg zu räumen, nicht Aufgaben anzuweisen bzw. im Team zu verteilen. Dies obliegt alleine dem Team selbst.

5.1.4 Weitere Scrum-Rollen

In der Literatur findet man manchmal weitere Scrum-Projektrollen,[86] die zum Verständnis des Prozesses nützlich sind, aber auch in „klassischen“ Prozessmodellen vorkommen:

5.1.4.1 Der Kunde

Der Kunde beauftragt die Produktentwicklung. Er ist budgetverantwortlich, sowohl bei interner als auch bei externer Projektvergabe. Er entspricht der Rolle des Kunden, wie wir sie aus anderen agilen Methoden kennen, übernimmt aber keine Projektverantwortung.

5.1.4.2 Der Anwender

Der Anwender ist der spätere Benutzer der Software. Üblicherweise hat er bereits Erfahrungen mit der Vorgänger-Software (falls eine solche vorhanden ist) und kann dem Team wertvolle Tipps bezüglich der *Usability* (Nutzbarkeit) geben. Team und Product Owner tun gut daran, solche Tipps zu beherzigen und in die Sprint-Planung einzubeziehen, weil die Akzeptanz der Software damit ungemein steigt.[87] Nach dem Sprint wird der Anwender die Realisierung seiner Tipps prüfen.

Das Trennen der Rollen Kunde und Anwender ist nicht unproblematisch. In klassischen, nicht agilen Projekten entspricht es der Realität, andere agile Methoden[88] versuchen hingegen, einen Kunden zu finden, der auch die Rolle des Anwenders hinreichend gut ausfüllen kann. Somit kann er auch die Machbarkeit eines Wunsches sofort abschätzen und wird damit für das Team verlässlicher.

5.1.4.3 Das Management

Das Management sorgt für die Rahmenbedingungen, innerhalb derer ein Projekt durchgeführt wird. Meist löst es die vom ScrumMaster im Projektumfeld identifizierten Probleme.

5.1.5 Gefahr durch Rollenmissbrauch

Da in Scrum die klassische Rolle des *Projektleiters* fehlt, besteht insbesondere in unerfahrenen Scrum-Teams die Gefahr, dass Product Owner oder ScrumMaster die

---

[86] siehe z.B.(Gloger 2009). Die Rollen werden allerdings bei (Schwaber 2004) nicht erwähnt.

[87] Die Erfahrung des Autors zeigt allerdings, dass in der Regel nicht alle Wünsche des Anwenders berücksichtigt werden können. Manchmal widersprechen sich Wünsche, oft sind sie auch im Rahmen des Budgets nicht realisierbar. Dieser Konflikt entsteht durch das Trennen der Rollen Kunde und Anwender.

[88] z.B. XP (Kap. 3) und MAP (Kap. 7)

Rolle übernehmen und der „gelebte“ Prozess somit von Scrum abweicht. Problematisch ist, dass mit Einführung von Scrum meist ehemalige Projektleiter die Rolle des ScrumMasters übernehmen. Damit ist die Gefahr eines „Rückfalls“ in alte Denkmuster naturgemäß groß. Bei unerfahrenen Product Ownern besteht die Gefahr, dass sie die Autonomie des Teams verletzen.

Bei Einführung von Scrum muss also darauf geachtet werden, dass Product Owner oder ScrumMaster ihre Rollen nicht missbrauchen. Es empfiehlt sich, Product Owner, ScrumMaster und Team im Verlauf des Projekts immer wieder durch externe Scrum-Experten zu coachen.

5.2 Scrum – der Prozess

Im Weiteren soll das Zusammenspiel der Scrum-Rollen beleuchtet werden: Wie sieht der Ablauf eines Scrum-Projekts aus?

5.2.1 Scrum-Flow – Überblick

Der Ablauf des Scrum-Prozesses ist einfach gehalten (Abb. 5.2). Am Anfang steht die *Vision* des Product Owners mit einer kurzen Beschreibung des Produkts, einer Abschätzung des Aufwands und möglicher Meilensteine.[89] Letztlich muss die Frage geklärt sein, ob es sinnvoll ist, das Projekt anzugehen, bevor man in den eigentlichen Scrum-Prozess eintritt. Wenn diese Business-Entscheidung gefällt ist, wird das *Product Backlog* vom Product Owner gefüllt.[90] Hier werden alle Anforderungen an die Software festgehalten und priorisiert.

Erstellt wird die Software vom Team in Iterationszyklen, den sog. *Sprints*. Alle Sprints dauern gleich lang, üblicherweise wählt man maximal 30 Tage für einen Sprint. Für jeden Sprint wird zwischen Product Owner und Team ein Endtermin vereinbart, der eingehalten werden muss. Zu Beginn des Sprints findet eine *Sprint-Planungssitzung* statt, in der das Team das *Sprint Backlog* erstellt. In diesem Planungsdokument wird festlegt, welche Anforderungen für das nächste Inkrement der Software abgearbeitet werden sollen und welche Aktivitäten dazu notwendig sind. Die Planung des Sprints geschieht entsprechend den Vorgaben im Product Backlog. Danach beginnt der Sprint, also die eigentliche Implementierung des nächsten Inkrements der Software. Hier entwickelt das Team ohne Störung von außen das neue Software-Inkrement mit den im Sprint Backlog festgelegten Features.

Jeden Tag am selben Ort zur selben Zeit findet eine Teambesprechung statt, die *Daily Scrum* genannt wird. Wichtig ist, dass diese Sitzung kurz gehalten wird. Neben dem Team *muss* der ScrumMaster anwesend sein. Der Product Owner hingegen

---

[89] Hier handelt es sich naturgemäß nicht um seriöse Schätzungen, sondern um *Ziele*, die man gerne erreichen würde. Sie müssen später vom Team durch Aufwandschätzungen untermauert werden (oder eben auch nicht).

[90] Hier kann er bereits vom Team unterstützt werden. Die Verantwortung bleibt aber beim Product Owner.

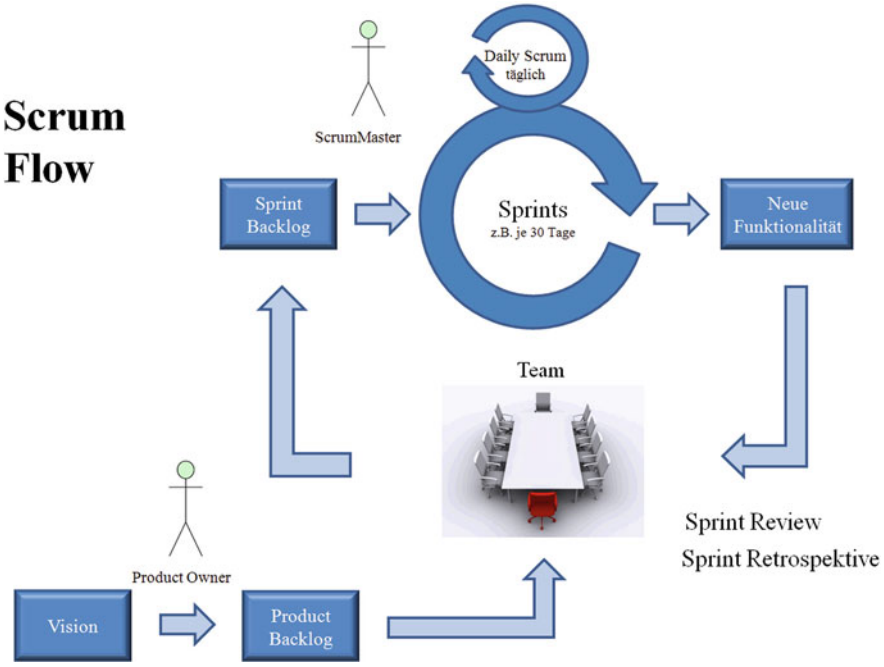

Abb. 5.2 Scrum Flow

sollte anwesend sein. Betroffene Gäste sind erlaubt, haben aber kein Rederecht. Das Team hat Gelegenheit, die Arbeit zu koordinieren und Probleme anzusprechen und entweder gleich zu lösen oder wenigstens an den ScrumMaster oder den Product Owner weiterzugeben.

Am Ende des Sprints steht das *Sprint Review*. Hier wird das entstandene Software-Inkrement vom Product Owner geprüft und abgenommen. Dabei werden nur vollständige und fehlerfreie Arbeitsergebnisse akzeptiert. Der Sprint schließt mit der *Sprint-Retrospektive*, die dem Team Gelegenheit gibt, über den Ablauf des Sprints nachzudenken, insbesondere über die Zusammenarbeit im Team und über Verbesserungsmaßnahmen. Diese bilden die Grundlage für den nächsten Sprint.

5.2.2 Sprint – Details

5.2.2.1 Sprintlänge

Üblicherweise werden in iterativen Prozessmodellen die Features der Software definiert, die im nächsten Inkrement enthalten sein sollen. Der Aufwand für ihre Implementierung wird spätestens jetzt vom Team geschätzt, und ggf. werden Features zurückgestellt, um die Iteration nicht zu lange andauern zu lassen. In größeren Projekten sind Iterationslängen von zwei bis drei Monaten keine Seltenheit. Von Iteration zu Iteration schwankt die zeitliche Länge, weil man die Realisierung

der konkreten Features der Software in den Vordergrund stellt und letztlich (aus Entwicklersicht) keinen Vorteil in einer gleichbleibenden Iterationslänge sieht.

Anders ist dies in Scrum: Scrum-Sprints haben *feste* Iterationslängen. Ken Schwaber beschreibt das Einführen von festen Sprintlängen als Reduktion der Komplexität des Projekts für das Team: Das Team muss nur eine feste Zeitdauer überblicken, um zum nächsten Ziel zu kommen. Schritt für Schritt wird es mit der zunehmenden Komplexität im Projekt fertig (Schwaber 2004). Dabei hat sich eine fixe Sprintlänge, die 30 Tage nicht überschreiten sollte, bewährt (*Timeboxing*). Das Team vereinbart gemäß der Sprintlänge einen Endtermin mit dem Product Owner. Dieser Endtermin *muss* eingehalten werden. Allerdings darf das Team auf dem Weg zum Endtermin auch nicht gestört werden.

5.2.2.2 Anfang des Sprints – die Sprint-Planungssitzung

Zu Beginn des Sprints findet die Sprint-Planungssitzung statt. Der Product Owner muss ein *Sprint-Ziel* definieren. Es muss ein Schritt in Richtung Projektziel sein und einen Nutzen für das Projekt bringen. Der Product Owner muss das Sprint-Ziel so definieren, dass es für das Team klar erkenntlich ist. Das Team verpflichtet sich zur Umsetzung des Ziels und der damit verbunden Features der Software. Nach der Sitzung kann der Product Owner für den laufenden Sprint keine Änderungen des Ziels mehr vornehmen. Es gilt also, das Ziel mit Bedacht zu wählen.

Der Product Owner definiert entsprechend dem Product Backlog und dem zuvor erstellten Releaseplan[91] die Anforderungen an das Produkt, die im aktuellen Sprint realisiert werden sollen. Dabei spielt die Priorisierung einer Anforderung im Product Backlog naturgemäß eine große Rolle. Wichtig ist auch, dass im Vorfeld geklärt wurde, ob diese Anforderungen problemlos zu erfüllen sind.[92] Dies kann durch den Product Owner und ausgewählte Teammitglieder erfolgen. Dazu gehört auch eine vernünftige Abschätzung des Aufwands für die Realisierung einer Anforderung.

In der Sprint-Planungssitzung erstellt das Team gemäß Sprint-Ziel und vom Product Owner ausgewählten Anforderungen das Sprint-Planungsdokument, das *Sprint Backlog*. Hier werden alle Features des nächsten Inkrements der Software identifiziert, die im Sprint realisiert werden sollen. Ebenso werden alle Aktivitäten identifiziert, die notwendig sind, um das Inkrement zu realisieren. Es empfiehlt sich auch hier ein iteratives Vorgehen – eine schrittweise Verfeinerung der durchzuführenden Aktivitäten und der Schätzung des benötigten Aufwands bei gleichzeitiger Präzisierung der vom Product Owner „bestellten" neuen Features der Software.[93]

---

[91] Der Releaseplan dient der Information, wann welches Feature der Software zu erwarten ist (Kap. 5.3.3).

[92] Sollten die Zweifel technischer Natur sein, muss ggf. vorab ein technischer Prototyp erstellt werden. (Vorsicht, das Erstellen eines Prototyps muss als zusätzliche Anforderung in das Product Backlog eingetragen werden).

[93] Aus der Erfahrung des Autors empfiehlt sich hier der Einsatz von User Story Cards (bekannt von XP). Siehe hierzu auch Kap. 5.3.2 und Kap. 3.

Sowohl Product Owner, als auch Team und ScrumMaster müssen in der Planungssitzung die ganze Zeit anwesend sein: Der Product Owner, um dem Team Rede und Antwort stehen zu können, der ScrumMaster, um die Selbstorganisation des Teams zu überwachen. Die Planungssitzung endet, wenn die dafür angesetzte Zeit (oft nicht mehr als ein Tag) abgelaufen ist. Als Ergebnis der Sprint-Planungssitzung muss ein realistisches Sprint Backlog vorliegen. Die Anforderungen, zu deren Realisierung sich das Team verpflichtet hat, müssen mitsamt den zugehörigen Aktivitäten definiert und abgeschätzt sein (Kap. 5.3.2).

5.2.2.3 Daily Scrum

Jeden Tag am selben Ort zur selben Zeit findet eine Teambesprechung statt, die *Daily Scrum* genannt wird. Sie darf nur *15 Minuten* dauern (Schwaber 2004),[94] das Team und sein ScrumMaster *müssen* anwesend sein. Die Anwesenheit des Product Owners ist nicht zwingend erforderlich. Wenn es seine Zeit erlaubt, sollte er allerdings anwesend sein. So kann er aus erster Hand erfahren, was das Team umtreibt. Allerdings sind diese Sitzungen nicht dazu gedacht, dass der Product Owner versucht, den Ablauf des Sprints zu beeinflussen. Gäste sind im Daily Scrum erlaubt, insbesondere alle vom Projekt Betroffenen, sie haben aber kein Rederecht.

Das Ziel des Daily Scrums ist es, die Arbeit des Teams zu koordinieren (Selbstorganisation!) und auftauchende Probleme anzusprechen. Dazu muss das Sprint Backlog aktualisiert sein. Das Team muss im Daily Scrum über den Status quo des Projekts informiert sein! In Anbetracht der Kürze des Daily Scrums sollte jedes Teammitglied nur Antwort auf folgende drei Fragen geben (Schwaber 2004):

- Was habe ich seit dem letzten Daily Scrum für das Projekt getan?
- Was will ich bis zum nächsten Daily Scrum für das Projekt tun?
- Was hindert mich daran, so effektiv wie möglich zu arbeiten?

Der ScrumMaster ist verantwortlich, dass diese Fragen von jedem Teammitglied beantwortet werden, ohne dass das Team in (endlose) Diskussionen verfällt. Etwas psychologisches Geschick ist angebracht um zu erkennen, wo es um persönliche Befindlichkeiten Einzelner und wo es um wirkliche Probleme geht, die dann in der Regel auch andere Teammitglieder behindern. Kleinere Probleme kann das Team ggf. nach dem Daily Scrum gleich lösen. Komplexere Probleme müssen aufgezeigt und an ScrumMaster oder Product Owner weitergegeben werden. Es ist wichtig, dass diese die gemeldeten Probleme *ernst* nehmen und wissen, dass ungelöste Probleme die Leistung des Teams in der Regel mindern.

[94] Der Autor hat gute Erfahrungen mit *Standup Meetings* gemacht, also Sitzungen, die im Stehen (oft an der Kaffeemaschine) abgehalten werden. Solche Sitzungen bleiben in der Regel kurz!

5.2.2.4 Ablauf des Sprints

Scrum ist eine reine Managementmethode und macht folglich dem Team keine Vorgaben bezüglich der angewandten Entwicklungspraktiken. Es ist allerdings sinnvoll, Praktiken, die man von anderen agilen Methoden kennt, wie z.B. von XP (Kap. 3), anzuwenden.

Wichtig ist, dass das Team im Sprint *ohne Störung* von außen entwickeln kann. In der Zeit des Sprints sind die Anforderungen eingefroren. Sie dürfen nicht geändert werden. Nur so hat das Team eine Chance, den anvisierten Endtermin zu halten. Am Endtermin ist der Sprint *fertig*, die Ergebnisse müssen dem Product Owner präsentiert werden.

5.2.2.5 Ende des Sprints – Sprint Review

Am Ende des Sprints steht das Sprint Review. Der Product Owner bekommt die Möglichkeit, das entstandene Software-Inkrement zu prüfen und abzunehmen. Die Sprint Review-Sitzung ist auf maximal vier Stunden beschränkt (Timeboxing, Schwaber 2004). Wichtig ist, dass der Product Owner nur vollständige und fehlerfreie Arbeitsergebnisse akzeptiert. Auch fast fertige oder fehlerhafte Ergebnisse werden nicht akzeptiert, unvollständige Resultate gelten als nicht erledigt. Solche Ergebnisse werden z.B. im Product Backlog festgehalten und Gegenstand des nächsten Sprints. Ziel ist, sie möglichst schnell (im nächsten Sprint) in Ordnung zu bringen, um das aktuelle Inkrement der Software zu vervollständigen.

Das vom Team entwickelte Inkrement der Software muss auf einem zentralen Integrationsrechner in Form einer Live-Demo präsentiert werden. Der Product Owner sollte selbst Tests an der Software durchführen können und sich von der (aus seiner Sicht) korrekten Funktionsweise der Software überzeugen. *Nur der Product Owner* entscheidet, ob das Ergebnis akzeptiert wird oder nicht! Auf keinen Fall darf der Product Owner einzelne Ergebnisse auf lokalen Rechnern akzeptieren. Der Autor hat die Erfahrung gemacht, dass die Integration solcher Ergebnisse auf einem Rechner keine Trivialität ist (Kap. 6.4) und in der Regel nur mit entsprechendem Aufwand zu realisieren ist. Somit gehört die Integration klar mit zum Sprint und muss am Ende erfolgreich abgeschlossen sein.

Neben Product Owner und Team muss auch der ScrumMaster am Sprint Review teilnehmen. Seine Aufgabe ist es, die Sitzung zu moderieren. Interessant ist es außerdem, Endanwender der Software dabeizuhaben. Von ihnen können wertvolle Tipps und Ideen für die Weiterentwicklung der Software kommen. Es erhöht die Akzeptanz der Software ungemein, wenn der Endanwender regelmäßig das Wachsen der Software mitverfolgen kann.

5.2.2.6 Sprint Retrospektive

Die Sprint Retrospektive schließt direkt an das Sprint Review an. Auch diese Besprechung ist in ihrer Dauer beschränkt. Ken Schwaber fordert eine maximale Länge von drei Stunden (Schwaber 2004). Die Sprint Retrospektive gibt dem Team

Gelegenheit, über den Ablauf des Sprints nachzudenken, insbesondere über die Zusammenarbeit im Team und über notwendige Verbesserungsmaßnahmen. Diese bilden die Grundlage für den nächsten Sprint.[95] Oftmals geht um die Einführung, Abschaffung oder Veränderung agiler Praktiken, die das Team für sinnvoll oder eben nicht mehr für sinnvoll hält (Kap. 6).

Teilnehmer der Sprint Retrospektive sind zwingend das Team und der ScrumMaster. Die Teilnahme des Product Owners ist optional. Die Erfahrung zeigt aber, dass sein Verständnis für das Team mit der Teilnahme an der Sprint Retrospektive wächst. Die Aufgabe des ScrumMasters ist es, die vom Team vorgeschlagenen Veränderungen zu sammeln und zu notieren. Es ist nicht seine Aufgabe, dem Team Lösungen aufzuzeigen. Er hat also die Rolle eines Moderators inne! Das Team priorisiert die vorgeschlagenen Veränderungen und bespricht sie in dieser Reihenfolge. Veränderungen, die gemeinsam beschlossen werden, werden im nächsten Sprint umgesetzt. Das Ziel der Sprint Retrospektive ist die Optimierung des Prozesses. Angesprochene Probleme, die zu keinen Veränderungen führen, sind frustrierend und unterhöhlen die Akzeptanz der Sprint Retrospektive. Warum soll man auch Zeit für eine Verbesserungssitzung opfern, wenn keine Verbesserung dabei heraus kommt!

5.3 Scrum-Artefakte

Die wichtigsten Scrum-Artefakte (Dokumente) sind das Product Backlog, das Sprint Backlog und der Releaseplan mit Burndown Chart. Die Backlogs sind Planungsinstrumente, während der Releaseplan der Information dient, wann welches Feature der Software zu erwarten ist. Die Burndown Chart dient der Überwachung des Projektfortschritts.

5.3.1 Product Backlog

Das Product Backlog ist das zentrale Dokument zum Erfassen und Verwalten von Anforderungen. Das Product Backlog verwaltet *Items* – die zu liefernden Funktionalitäten. Ein typisches Product Backlog zeigt Abb. 5.3.

Nachdem die Funktionalitäten im Product Backlog gesammelt wurden,[96] werden sie priorisiert, und der Realisierungsaufwand wird geschätzt. Dies geschieht durch den Product Owner (idealerweise mit Hilfe des Teams).

In der Praxis hat sich auch die elektronische Darstellung des Product Backlogs bewährt. Dies kann mit Hilfe eines gängigen Bugtracking-Tools geschehen (Abb. 5.4). Wichtig dabei ist, dass das Product Backlog für alle Teammitglieder, den Product Owner und den ScrumMaster deutlich sichtbar, z.B. auf einen großen

[95] Dies ist natürlich keine Erfindung von Scrum, bei Crystal heißt dieses Meeting *Reflexions-Workshop* (Kap. 4).

[96] Eine Funktionalität ist üblicherweise eine Zeile im Product Backlog!

Priorität	Thema	Beschreibung	Akzeptanzkriterien	Geschätzter Aufwand [in Personentagen]
3	Eingabemaske Abonnenten	Die Eingabemaske muss um das Feld "Kampagne" erweitert werden.	Es dürfen nur gültige Kampagnen eingegeben werden.	1
1	Versandsystem	Versand muss unterbrechbar gemacht werden	Abgebrochener Versand muss an dieser Stelle wieder aufgenommen werden.	5
2	Anlegen neue Kampagne	Neue Maske "Kampagne" mit Zuordnung zu Newslettern (Listbox zur Auswahl)	Kampagne muss nach Anlegen sichtbar sein	3

Abb. 5.3 Typisches Product Backlog für ein Newsletter-Versandsystem

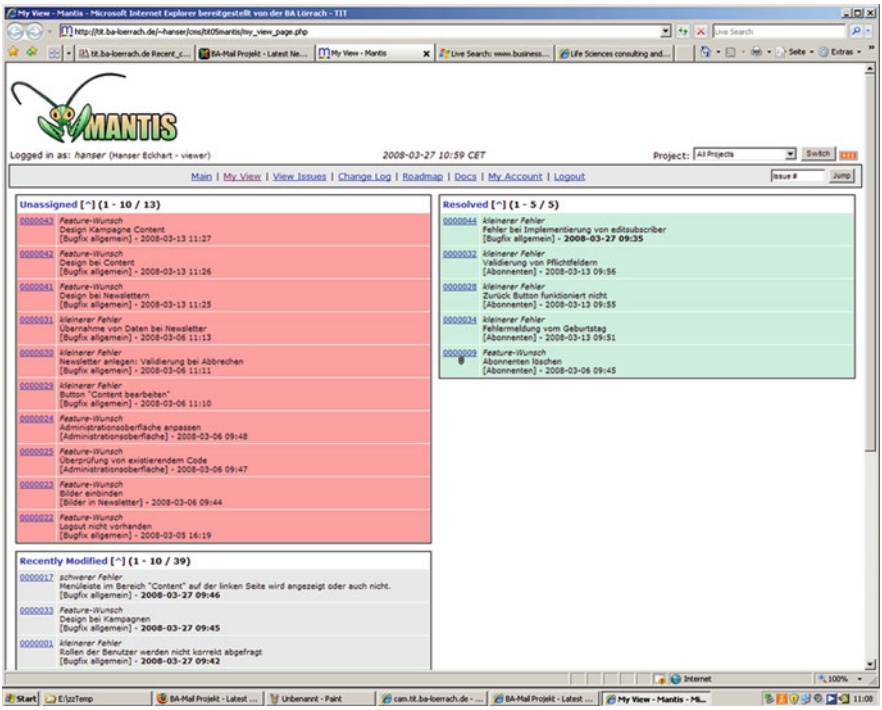

Abb. 5.4 Product Backlog in elektronischer Form

LCD-Schirm im Team-Büro dargestellt wird. Ist dies nicht der Fall, ist es nicht möglich, „schnell im Vorbeigehen“ einen Blick auf den Gesamtstand des Projekts zu werfen.[97] In elektronischer Form abgespeichert können die Items auf einfache Art und Weise beim Sprint in das Sprint Backlog übernommen werden.

[97] Das hat sich im studentischen Labor des Autors als klarer Nachteil herausgestellt. Das Team neigt dann dazu, die Projektsituation zu positiv zu beurteilen, wenn nicht *alle* Beteiligten ständig den aktuellen Gesamtstand vor Augen haben.

5.3.2 Sprint Backlog

Das Sprint Backlog ist eine Liste der Aktivitäten, die das Team innerhalb eines Scrum-Sprints abarbeiten muss. Es dient dem Team zur Organisation des Sprints. Das Sprint Backlog ist das Resultat der Sprint-Planungssitzung (Kap. 5.2.2). In der Planungssitzung teilt der Product Owner dem Team sein Sprint-Ziel mit. Er identifiziert die Anforderungen im Product Backlog, die das Team erfüllen muss. Das Team verpflichtet sich zum Erreichen des Sprint-Ziels und, damit verbunden, zur Umsetzung der vom Product Owner identifizierten Anforderungen aus dem Product Backlog.

Nachdem die dem Sprint-Ziel zugeordneten Anforderungen identifiziert wurden, werden sie in das Sprint Backlog aufgenommen.[98] Ihr Aufwand ist spätestens zu diesem Zeitpunkt üblicherweise in Personenstunden abgeschätzt. Er sollte vier bis 16 Stunden nicht überschreiten (Schwaber 2004). Sind aus dem Product Backlog Anforderungen übernommen worden, deren Aufwand größer ist, müssen sie in mehrere, kleinere (Teil-) Anforderungen zerlegt werden. Damit geht eine Präzisierung der Anforderungen einher. Kleine „Arbeitspakete“ können besser überwacht werden. Das Team weiß zu jedem Zeitpunkt über den Status jedes *Sprint Backlog Items* (also jedes Arbeitspakets) Bescheid.

Es ist sinnvoll, die Sprint Backlog Items als *User Stories* darzustellen.[99] Wie bei XP (Kap. 3) können diese dann als Einzelpapiere (z.B. Din A5 quer) an eine Pinwand gehängt werden. Das Sprint Backlog in Listenform enthält dann nur noch eine Kurzzusammenfassung der Items, um den Überblick zu wahren (Abb. 5.5).

Das Sprint Backlog ist ein „lebendes“ Dokument. Es wird täglich, idealerweise vor dem Daily Scrum, angepasst und spiegelt damit jederzeit den aktuellen Status des Sprints wider. Wird das Sprint Backlog als User Stories an einer Stellwand realisiert, kann das tägliche Update besonders einfach durchgeführt werden. Unerledigte User Stories sollten links und erledigte rechts an der Stellwand aufgehängt werden. Noch besser wäre es, zwei farblich unterschiedliche Stellwände zu benutzen.

5.3.3 Releaseplan und Burndown Chart

Für größere Projekte empfiehlt sich das Schreiben eines Releaseplans. Dies ist Aufgabe des Product Owners. Im Releaseplan wird die Abfolge der Sprints, also der Iterationen des Projekts, geplant. Inkrement für Inkrement nähert sich das Software-Produkt der endgültigen Lösung. Dem Releaseplan kann entnommen werden, wie viele Sprints benötigt werden, oder in welchem zeitlichen Rahmen das Projekt voraussichtlich stattfinden wird.

---

[98] Im Product Backlog ist der Aufwand für die einzelnen Anforderungen bereits geschätzt worden.

[99] Siehe auch (Wirdemann 2009).

Abb. 5.5 Sprint Backlog als User Stories an einer Stellwand

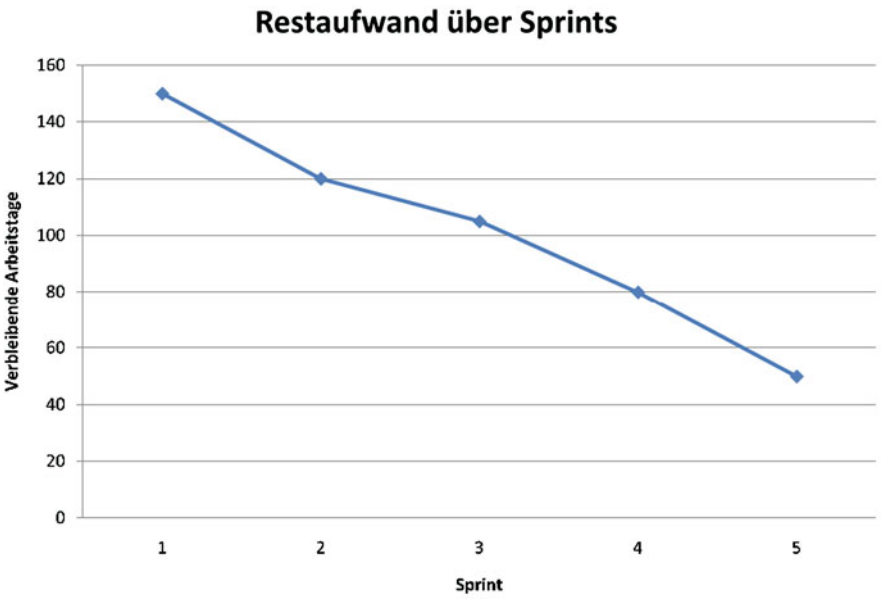

Abb. 5.6 Release Burndown Chart

Interessant ist die Überwachung der Releaseplanung mit Hilfe der *Release Burndown Chart.*[100] Hier wird pro Sprint der insgesamt verbleibende Aufwand im Projekt dargestellt (Abb. 5.6).

Eine Releaseplanung ist in agilen Projekten nicht unumstritten. Wie passt diese Gesamtplanung zu einem agilen Ansatz, der ja gerade den Anspruch hat, wechselnden Anforderungen, sog. „Moving Targets" (Kap. 3), folgen zu können. Die Antwort liegt in Scrum selbst: Während ein Sprint nicht unterbrochen werden darf, d.h. auch die Anforderungen des Sprints nicht verändert werden können (Kap. 5.2), kann das zwischen den Sprints durchaus passieren. Aus geänderten Anforderungen folgt zwangsläufig eine Anpassung des Releaseplans. Dies findet aber erst beim nächsten Sprint Beachtung (und eben nicht im aktuellen Sprint!).

Literatur

Cockburn A (2005) Crystal Clear – Agile Software-Entwicklung für kleine Teams. Bonn: MITP-Verlag

Gloger B (2009) Scrum – Produkte zuverlässig und schnell entwickeln. München: Hanser

Pichler R (2008) Scrum – Agiles Projektmanagement erfolgreich einsetzen. Heidelberg: dpunkt Verlag

Schwaber K (2004) Agile Project Management with Scrum. Washington, DC: Microsoft Press

Wirdemann R (2009) Scrum mit User Stories. München: Hanser

[100] *engl. burn down* = abbrennen