cassini
GUIDING AHEAD

Interpretation einer Story Point Schätzung

Product Owner in der Pflicht

Autor: Peter Kalvelage

Stand: 10.11.2014

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Aufwandsschätzungen in der Softwareentwicklung

Die Schätzung von Arbeitsaufwänden in der Softwareentwicklung stellt schon lange ein viel diskutiertes Thema dar. Schließlich bilden sie häufig die Grundlage umfangreicher Projektpläne und Vertragsinhalte. Dennoch bildet Software ein intangibles Gut, welches im Laufe der Entstehung und Existenz viele Veränderungen durchläuft und dabei ständig neue Technologien beinhalten kann. Es ist schwer, die notwendigen Tätigkeiten[1] präzise zu beziffern, zumal davon ausgegangen wird, dass sich die Anforderungen im Laufe der Zeit jederzeit ändern können. Durch diese Variabilität in den zu betrachtenden Inhalten ist man seit dem Beginn des Jahrtausends dazu übergegangen, alternativ zur Expertenschätzung in konkreten Personentage vielmehr in einer relativen Währung namens „Story Points“ zu schätzen[i]. Die Intention dahinter ist, der Exaktheitsillusion von konkreten Schätzwerten in einer schwer planbaren Domäne entgegenzuwirken bzw. den zeitlichen Aufwand für Schätzungen einzugrenzen.

Nachfolgend wird das Schätzen anhand von Story Points näher betrachtet. Es soll damit ausdrücklich keine Wertung zu anderen Schätzformaten abgegeben werden. Vielmehrwird eine Unterstützung geboten, wenn Story Point Schätzungen angewendet werden sollen. Dabei wird ein Vorgehen beschrieben, welches sich im Laufe der letzten Jahre bewährt hat und einen Umgang mit den teils doch nebulösen Story Points erleichtert. Wir möchten damit primär agilen Product Ownern eine Hilfestellung im Umgang mit ihren Anforderungen geben, nachdem sie ein erstes Feedback seitens der Entwicklung erhalten haben.

2 Einflussfaktoren auf Story Point Werte

Das ursprüngliche Verständnis der Story Points, wie Mike Cohn sie populär gemacht hat, entspricht einer numerischen Maßeinheit basierend auf Umfang, Unsicherheit und Komplexität einer abzuschätzenden Software-Technologieaufgabe. Diese Aspekte werden dabei nicht zwangsläufig anteilig einbezogen, sondern der jeweilige Einfluss variiert. Abbildung 1 ist hier missverständlich, denn der für die jeweilige Aufgabe wichtigste Aspekt dominiert dabei immer. Zudem werden die verschiedenen Aufgaben zueinander ins Verhältnis gesetzt, um durch eine solche Sortierung die Genauigkeit zu verbessern.

Eine direkte zeitliche Komponente spielt bei der Vergabe von Story Points im Übrigen keine Rolle. Der Schätzende sollte nicht ermuntert werden, den kognitiven Übertrag durchzuführen: „Ich benötige für diese Aufgabe zwei Tage, also vergebe ich fünf Story Points“. Die Ableitung von Planwerten erfolgt zu einem späteren Zeitpunkt und wird weiter unten beschrieben.

[1] Nachfolgend wird öfter von Tätigkeiten oder Aufgaben gesprochen. Im agilen Umfeld sind dies in der Regel technologische Tätigkeiten, welche durch User Stories beschrieben werden.

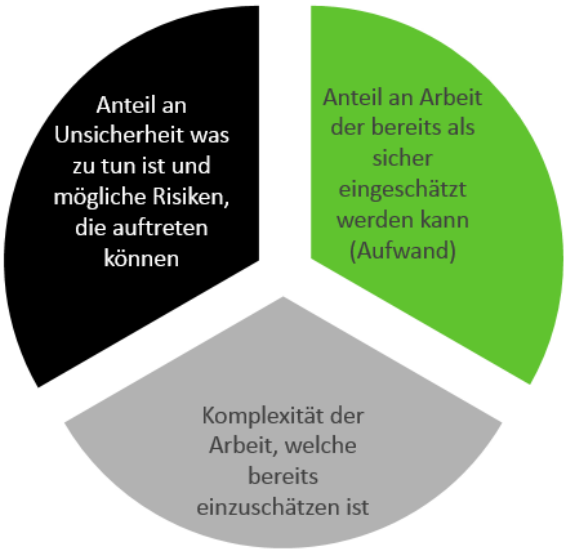

Abbildung 1: Einflussfaktoren auf einen Story Point Wert

Abgebildet werden die Ergebnisse meist auf eine angepasste Fibunacci-Folge, in der Regel mit den Werten 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100 und ∞. Sinn dieser Werte ist es, dass die nächst-höhere Zahl merklich größer als die vorherige ist und die Entscheidungen bewusster getroffen werden müssen. Die Werte können über eine ganze Reihe von Methoden wie zum Beispiel Planning Poker, Bucket Estimation oder Magic Estimation[ii] ermittelt werden.

3 Aussagekraft und Handlungsempfehlungen

Die nicht präzisen Einflussfaktoren auf Story Point Werte, welche Spielraum für Interpretationen lassen, stellen in vielen agilen Entwicklungsteams und auch in der agilen Community einen Grund großer Diskussion dar. Selbst Cohn befeuert die Diskussion durch widersprüchliche Blog-Artikel[iii] [iv]. Unseres Erachtens ist es dabei gar nicht relevant vorab zu definieren, welchen Anteil jetzt die Komplexität oder aber der Aufwand an einem Schätzwert haben sollte. Jede schätzende Person wird ihr eigenes mentales Model im Kopf haben und somit eine eigene Formel anwenden. Mathematisch betrachtet sind die Story Point Werte somit ein Ergebnis einer nicht-injektiven Funktion mit den drei genannten Faktoren als beeinflussende Komponenten. Abbildung 2 verdeutlicht diesen Umstand, indem vier Aufgaben mit jeweils acht Story Points geschätzt wurden, jedoch alle Schätzungen völlig unterschiedlich verlaufen sind.

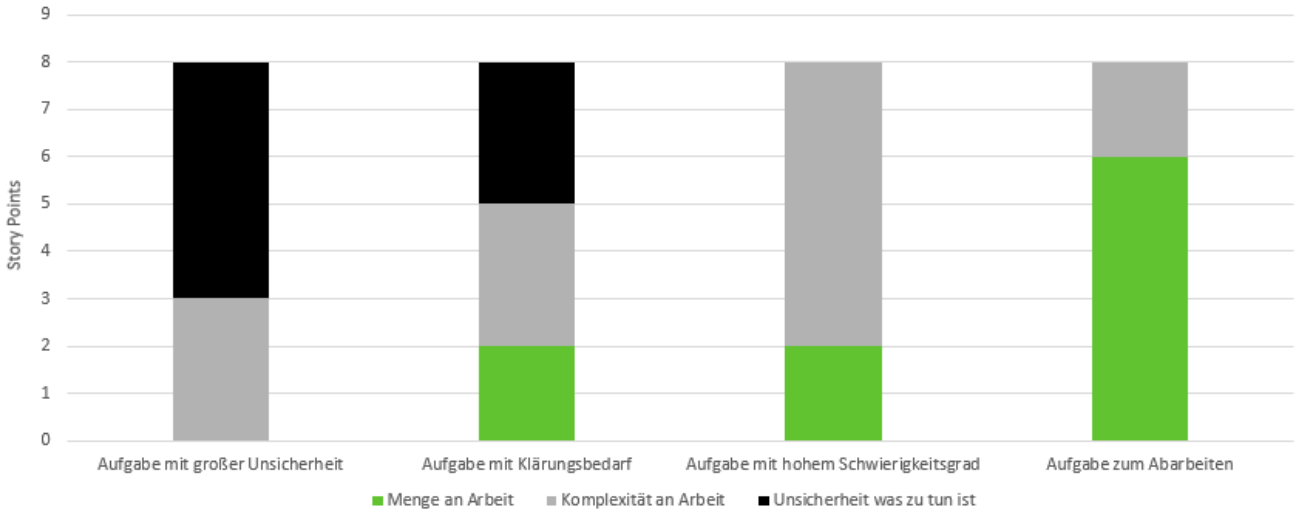

Abbildung 2: Zusammensetzung von vier verschiedenen "8er"-Aufgaben

Anstatt vorab darüber zu diskutieren, was in einen solchen Wert einfließen darf, muss es dem entsprechenden Anforderungsmanager, Business Analyst oder Product Owner im Scrum-Umfeld viel wichtiger sein zu erfahren, was real berücksichtigt wurde. Es gibt unseres Erachtens in dieser Situation zwei häufig anzutreffende Verhaltensweisen, von der eine schlecht und die andere destruktiv für kommende Schätzvorgängen ist. Destruktiv wäre es, mit Verhandlungen zu beginnen, ob die Aufgabe nicht doch „schneller“ gelöst werden kann und mit einer fünf zu bewerten ist. Dieses Verhalten führt die gesamte Schätzung ad absurdum, da die schätzenden Akteure sich nicht mehr ernst genommen fühlen und selbst wenn sie einer „Verkürzung“ zustimmen dürften, das Arbeitspaket dadurch nicht kleiner wird. Nicht ganz so schlimm, aber dennoch fahrlässig wäre hingegen der Schätzung nicht aufmerksam zu folgen und die Gründe für den Wert zu hinterfragen. Denn genau aus dieser Erkenntnis leiten sich die nächsten Handlungsschritte eines guten Anforderungsmanagers ab.

Das generelle Ziel muss sein, die Schätzergebnisse, die tendenziell weiter auf der linken Seite stehen, auf die rechte zu bewegen. Durch verschiedene Abstimmungen und den Ergebnissen daraus, wird sich meistens Unsicherheit und eventuell auch Komplexität eliminieren lassen. Daraus ergeben sich mit großer Wahrscheinlichkeit die beiden positiven Effekte, dass zum einen eine exaktere Einschätzung der Aufgabe vorgenommen werden kann und auch der Schätzwert sinkt. Abbildung 3 beschreibt, welches Verhalten in welcher Situation Sinn macht und wie wiederholte Schätzungen desselben Arbeitspaketes danach aussehen könnten.

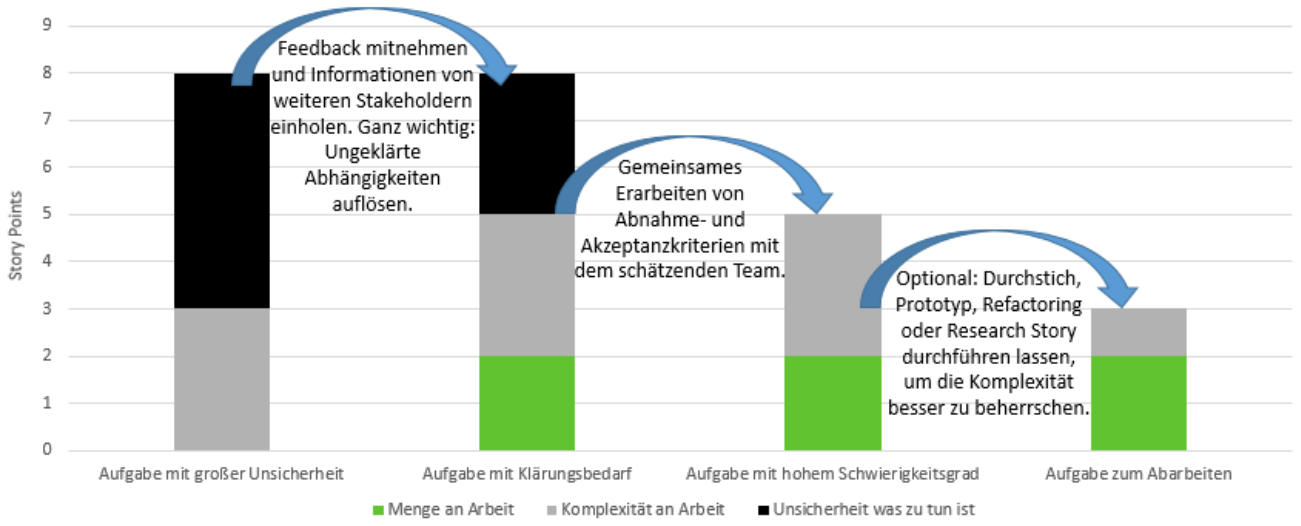

Abbildung 3: Handlungsempfehlungen

Es gilt hierbei, dass eine Aufgabe, welche ihren Schätzwert unter großer Unsicherheit erhalten hat, auf keinen Fall in einen Entwicklungssprint im Sinne eines Scrum-Vorgehens aufgenommen werden sollte. Das Team hat das Paket nur auf Basis einer ganzen Reihe vager Annahmen bewertet, dies jedoch nicht explizit gemacht. Zum Teil werden an dieser Stelle Quality Gates mit der Bezeichnung „Definition of Ready“ verwandt, um den nicht genau bewertbaren Charakter der Anforderung hervorzuheben[v]. Sollte dieses Verfahren nicht angewandt werden empfehle ich den Teams eigentlich immer, eine 100 oder das Unendlich-Symbol zu verwenden. Somit wird der kritische Zustand der Anforderungsbeschreibung hervorgehoben und ihr verantwortlicher Ersteller in die Pflicht zur Nachbesserung genommen. Für ihn ist es dann wichtig, die offenen Fragen mitzunehmen und zu klären sowie Abhängigkeiten zu dritten Parteien („das Design bekommen wir nächste Woche“) vorab aufzulösen.

Nachdem nachträglich möglichst viele Informationen aus dem Kontext des Anforderungspaketes gesammelt und dokumentiert wurden, kann man dazu übergehen, teaminterne Aspekte zu besprechen. Es gilt dem Team zu verdeutlichen, unter welchen Kriterien die Entwicklung als abgeschlossen gelten kann. Für konstante, wiederzuverwendende Aspekte kann man hierzu ein weiteres Quality Gate mit der Bezeichnung „Definition of Done“ nutzen[vi]. Für anforderungsindividuelle Punkte werden Akzeptanzkriterien formuliert, die für eine Abnahme notwendig sind. Sie begrenzen sozusagen das Handlungsfeld innerhalb dessen die Entwicklung die Aufgabe umsetzen darf. Es ist auch immer sehr hilfreich den Business Impact einer Anforderung direkt mit zu dokumentieren. Insofern nicht nur eine rein technische Beschreibung zu erstellen, sondern auch möglichst exakte Informationen über Mehrwerte aufführen. Alleine aus dem Verständnis dieser ergeben sich aber oft technische Aspekte, die sonst übersehen werden. Es kann auch für den Entwickler motivierend sein zu wissen, dass eine alternative Benutzerführung eine Umsatzsteigerung um ein Viertel ausmachen wird. Unterschieden wird demnach zwischen intern

und extern zu klärenden Informationen, wobei fragliche Punkte nach außen hin mehr Einfluss auf die Schätzexaktheit haben. Beide müssen jedoch geklärt werden.

Bei Aufgaben, deren fragliche Faktoren geklärt sind, die aber eine hohe Komplexität aufweisen, gilt es abzuwägen, ob das Risiko eingegangen werden sollte und sie ohne Vorarbeiten in die Entwicklung übernimmt. Das Ergebnis kann unter Umständen überraschend sein, indem ein Risiko unterschätzt wurde wodurch man länger braucht als erwartet. Als Alternative bieten sich nachforschende Vorarbeiten an, um schon mal ein Gerüst zu entwickeln, sich mit einer schwierigen Technologie bekannt zu machen oder aber einfach die komplexen Codebestandteile so umzubauen, dass sie einfach zu handhaben sind. Diese Entscheidung liegt im Ermessen des Anfordernden, sollte aber in Rücksprache mit den Entwicklern getroffen werden.

Wie man sieht, ist es möglich durch klärende Vorarbeit und Komplexitätsreduktion die Schätzung für ein Aufgabenpaket zu verkleinern (letztendlich wurde relevante Arbeit vorab durchgeführt) und auch zu präzisieren (neue Informationen geben dazu den Ausschlag). In der Scrum Methodologie wird der Scrum Master als Servant Leader bezeichnet, der durch seine Arbeit den Arbeitsprozess beschleunigt. Aber auch sein ergebnisverantwortliches Gegenstück, der Product Owner, kann sich hier als Servant Leader verstehen. Seine Aufgabe ist es, das Entwicklungsteam mit so wenig Unsicherheit wie nur eben möglich zu belästigen und möglichst viele notwendige Informationen vorab einzuholen[vii].

4 Kurzer Ausblick auf die Release- und Projektplanung

Abschließend darf die Frage gestellt werden, wie der Zeitaspekt, der bei der Aufwandschätzung keine Rolle spielt dann einbezogen wird. Lösung dieses Problems ist es, dass hier ausschließlich eine ex-post Betrachtung über die vergangene Durchschnittsgeschwindigkeit (Velocity) passieren muss. Die planende Person kann schauen, wie viele Story Points im Durchschnitt in einer Zeiteinheit abgearbeitet wurden und somit eine relativ genaue Prognose für die Zukunft aufstellen. Realistisch betrachtet wird der Blick in die Zukunft immer weniger exakt, je weiter er entfernt ist[viii]. Zudem wird es immer wieder einen Ausreißer in der Einschätzung geben. Durch das Gesetz der großen Zahlen ist sichergestellt, dass mit fortlaufender Entwicklungsdauer die Voraussagen immer exakter werden.

Wenn Sie mehr Interesse an dem Thema oder Fragen haben, besuchen Sie uns auf agile.cassini.de.

---

[i] Cohn, Mike: „Agile Estimating and Planning“, 2008
[ii] http://www.infoq.com/news/2010/10/estimation-techniques
[iii] http://www.mountaingoatsoftware.com/blog/story-points-are-still-about-effort
[iv] http://www.mountaingoatsoftware.com/blog/its-effort-not-complexity
[v] http://www.armerkater.de/2011/07/agile-breakfast-konstanz-definition-of-ready/
[vi] https://www.scrum.org/Resources/Scrum-Glossary/Definition-of-Done
[vii] http://www.scrumguides.org/scrum-guide.html
[viii] Boehm, Barry: “Software Engineering Economics”, 1981