Agil in großen Organisationen: Eine neue Rolle im Scrum-Framework

Patrick Daut

Cassini Consulting
Cassini Consulting Nord GmbH
Johannisbollwerk 16
20459 Hamburg
patrick.daut@cassini.de

Abstract: Scrum ist zunächst definiert für einzelne Teams und ist auf dieser Ebene gut erprobt und weit verbreitet. Die Realität zeigt jedoch den Bedarf, das agile Framework auch im Konzern-Kontext zur Anwendung zu bringen. Und große Organisationen erhoffen sich die Agilität einer Kleinen. Die Anwendung von Scrum in großen Organisationen mit einer Vielzahl von Teams ist jedoch weit weniger gut beschrieben.

Nach Scrum soll die Rolle des Product Owners explizit durch eine einzige Person und nicht durch ein Komitee wahrgenommen werden. Diese Singularität erschwert eine Skalierung von Scrum, zumal der Product Owner neben der Wertmaximierung des Produkts eine Fülle weiterer Aufgaben verantwortet. Der Product Owner wird so schon früh bei der Skalierung zum Engpass; entstehende Wartezeiten sind kontraproduktiv und widersprechen der „Lean"-Idee.

Zur Lösungsfindung wird ein Vorgehen entsprechend Goldratt's Theory of Constraints [Go90] aufgegriffen. Diesem Prinzip folgend werden mit wachsender Größe der Organisation verschiedene Lösungen diskutiert, die den Engpass „Product Owner" jeweils voll auslasten und dann beseitigen. Zur finalen Beseitigung des Engpasses wird die Verteilung der Verantwortlichkeiten auf zwei Rollen vorgeschlagen: Der Product Owner übernimmt nur noch die Verantwortung für die Anforderungspriorisierung während eine weitere Rolle, der „Story Owner", für die Detailausgestaltung und Klärung jeweils einer Story zuständig aber nicht einem festen Team zugeordnet ist. Der Story Owner lässt sich beliebig skalieren. Dieser Vorschlag folgt in Form von Delegation und Empowerment Prinzipien aus dem „*Lean*"-Konzept.

1 Der Product Owner als Engpass

Scrum beruht auf der Arbeit individueller Teams. Die Zusammenarbeit auf dieser Ebene ist gut erprobt und weit verbreitet. Bei einem Einsatz von Scrum in großen Unternehmen und bei der Entwicklung eines gemeinsamen Produkts durch eine große Anzahl von Teams werden wir in der Praxis aber zunächst allein gelassen: Literatur wie auch Erfahrungsberichte haben bislang keine Patentrezepte zur Skalierung von Scrum über die Teamebene hinaus geliefert. Die Realität zeigt jedoch den Bedarf, das agile Framework auch im Konzern-Kontext zur Anwendung zu bringen. Und große Organisationen erhoffen sich die Agilität einer Kleinen.

Die Rolle des Product Owners soll nach Scrum explizit durch eine einzige Person und nicht durch ein Komitee wahrgenommen werden, da sie über die Prioritäten entscheidet, in denen Anforderungen umgesetzt werden. So wird das Produkt definiert und dessen Wertmaximierung zentral verantwortet. Darüber hinaus hat der Product Owner weitere Aufgaben, wie die Detailabstimmung von Anforderungen und der Ansprechpartner für die Entwicklungsteams und Stakeholder zu sein. Wird Scrum in einer großen Organisation eingesetzt, zeigt sich ein Engpass in der Singularität des Product Owners und damit in der Beziehung zum Kunden: Durch den Einsatz mehrerer Teams kann die Kapazität der Organisation erhöht werden aber der Product Owner wird zum Engpass.

Im Fokus der Betrachtung steht hier nicht ein einzelnes Entwicklungsprojekt sondern eine Organisation oder ein Teil einer Organisation, die zu dem Zweck geschaffen ist, kontinuierlich neue Produkte zu entwickeln oder bestehende Produkte weiterzuentwickeln. Die Argumentation ist aber für Projekte ab einer gewissen Größenordnung ebenso gültig.

2 Ein Blick über den Tellerrand

Wie können wir uns methodisch diesem Problem nähern? Die Theory of Constraints, die auf den Arbeiten von Goldratt beruht, floss in Methoden ein, die in erster Linie in Bereichen wie Produktion und Logistik Anwendung finden. Im Kern besagt sie [Go90]:

> (A) Ein Engpass ist alles, was das System von der Erreichung einer höheren Leistung – im Sinne des Durchsatzes – abhält.
>
> (B) Jedes System muss über mindestens einen solchen Engpass verfügen.

Weiter wird im Rahmen der Theory of Constraints ein Vorgehen vorgeschlagen, wie in Abbildung 1 dargestellt.

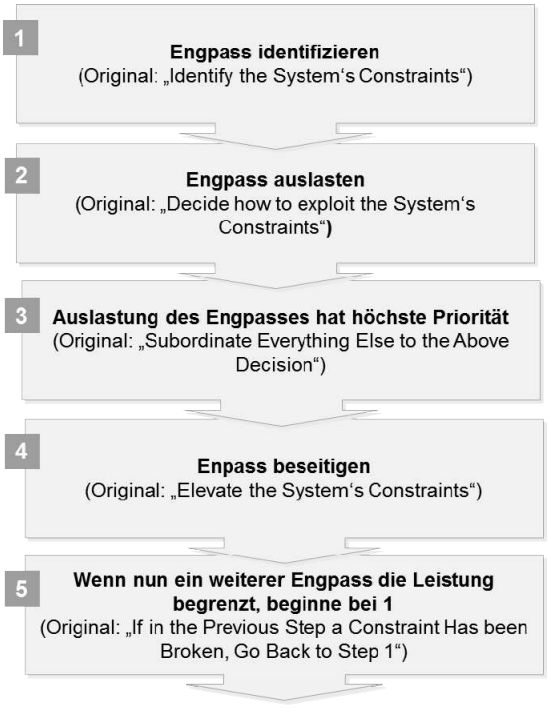

Abbildung 1: Vorgehen der Theory of Constraints [Go90]

Dahinter steht der folgende Gedanke: Ist ein System soweit ausgelastet, dass der Engpass den Durchsatz begrenzt, so bringt das Hinzufügen von Ressourcen an anderer Stelle keinen weiteren Nutzen bis der Engpass aufgehoben wurde. Im Fokus liegt also, zunächst alles daran zu setzen, den Engpass auszulasten und ihn im Anschluss zu beseitigen.

3 Die Theory of Constraints und die Skalierung von Scrum

Übertragen wir nun diese Methode auf die Skalierung von Scrum in großen Organisationen.

Scrum ist zunächst nur für ein Team definiert. Ein ideales Scrum Team sollte nicht mehr als 7-9 Mitglieder umfassen. Mit dem Begriff „Skalierung“ im Rahmen dieses Beitrags einer der beiden folgenden Fälle gemeint:

> (A) Die Einführung von Scrum als Arbeitsmodell in einer Entwicklungsorganisation, die mehr Mitarbeiter umfasst als in einem Scrum Team organisiert werden können.

(B) Das Wachstum einer nach Scrum arbeitenden Organisation über die ideale Größe eines Teams hinaus.

Die Skalierung kann durch die Aufteilung von Arbeitsaufträgen bzw. Kundenanfragen auf mehrere Scrum Teams erreicht werden: Wächst eine Organisation (Fall B) oder in einer bestehenden Organisation wird Scrum schrittweise eingeführt (Fall A), wird im Zuge dessen die Anzahl an Teams erhöht, um den benötigten Durchsatz erreichen zu können. Entsprechend der Theory of Constraints können immer weitere Scrum Teams hinzugefügt werden, bis der erste Engpass auftritt. Erkennbar ist diese Situation daran, dass weitere Ressourcen in Form eines zusätzlichen Teams keine oder nur noch eine geringe Steigerung des Durchsatzes in der Entwicklungsorganisation bewirken können. Der Engpass ist ausgelastet.

4 Die Organisation wächst

4.1 Der Anfang: Warum wird der Product Owner zum Engpass?

Gehen wir somit im Falle einer Skalierung davon aus, dass mehrere Teams existieren. Weiter wird angenommen, dass diese an einem einzigen gemeinsamen Produkt arbeiten und somit Koordinationsbedarf zwischen den Teams besteht. Zunächst belassen wir es auch bei steigender Anzahl von Teams bei einem einzigen Product Owner, den sich alle Teams teilen. Damit ist eine grundlegende Anforderung von Scrum erfüllt: Es ist explizit vorgesehen, dass die Rolle des Product Owner durch eine Person ausgefüllt wird, nicht durch ein Komitee oder eine Gruppe. Dies liegt darin begründet, dass der Product Owner inhaltlich für das Produkt verantwortlich sein und verantwortlich gemacht werden soll. Dieser darf als einziger Anforderungen priorisieren und übernimmt damit die Verantwortung für die Ausgestaltung des Produkts zu einem bestimmten Zeitpunkt. Stakeholder nehmen lediglich indirekt darauf Einfluss.

Entsprechend der Theory of Constraints kann es sinnvoll sein, allen Teams einen einzigen Product Owner zuzuweisen bis dieser zum einschränkenden Faktor wird. Je nach Komplexität der Arbeitsaufgaben und des zu entwickelnden Produkts kann dieser Fall bereits bei zwei oder auch erst mehr Teams eintreten. Solange es nicht nötig ist, muss die oben beschriebene Singularität des Product Owners nicht aufgelöst werden. Wichtig ist, dass der Product Owner von allen Aufgaben befreit wird, die nicht zu seiner Rolle gehören und alle nötige Bedingungen geschaffen werden, die ihm als Engpass möglichst effizientes Arbeiten ermöglichen. Diesem muss höchste Priorität zugeordnet werden entsprechend Schritt 3 des Vorgehens der Theory of Constraints [Go90].[1] Es ist allerdings unwahrscheinlich, dass eine Person mehr als zwei bis drei Teams als Product Owner dienen kann, ohne die Arbeit dieser Teams zu bremsen.

Die Erfahrung zeigt, dass – wie in Abschnitt 1 postuliert – dieser erste Engpass oft an der Rolle des Product Owners auftritt. Wie kommt es dazu? Der Product Owner verantwortet einerseits die Priorisierung von Anforderungen, andererseits ist es von großer

[1] S. auch Abbildung 1

Bedeutung, dass er während der Detaillierung von Anforderungen bzw. während deren Umsetzung dem Team als Ansprechpartner und ggf. als zentraler Kanal zum Kunden zur Verfügung steht. Die ständige Verfügbarkeit des Product Owners in seiner Funktion als „on-site customer" ist für den Erfolg agilen Vorgehens von großer Bedeutung, da Kommunikation nach Bedarf einer detaillierten Spezifikation im Vorfeld vorgezogen wird.[2] Mit steigender Anzahl von Teams wird der Product Owner überlastet und bremst die Arbeit der Teams aus. Darüber hinaus kommen mit steigender Organisations- und Produktgröße weitere Stakeholder, darunter verschiedene Managementpositionen, unternehmensweite Querschnittsfunktionen wie Security und Compliance oder Investoren, weitere Kunden, Dienstleister, Partner und Regulierungsinstanzen hinzu. Für all diese – verschiedenen – Aufgaben sieht Scrum die Rolle des Product Owners vor.

Eine Überlastung des Product Owners führt dazu, dass Entwicklungsteams bei Design und Implementierung auf den Product Owner warten müssen: Der Product Owner ist ein ausgelasteter Engpass und auch durch weitere Teams kann der Durchsatz der Organisation kaum noch erhöht werden. Weiterhin versucht ein Team, die entstehende Wartezeit anderweitig zu nutzen um seine Auslastung zu erhöhen. Es erhöht damit aber auch seine „Work in Progress" und durch häufige Fokuswechsel der Entwickler entsteht eine Verschwendung im Sinne der *Lean*-Idee, also Aufwand und Zeit, die nicht zur Schaffung von Wert beitragen [Oh88].[3]

4.2 Eine Lösung für kleinere bis mittelgroße Organisationen

Um den Engpass aufzulösen wird in diesem Beitrag vorgeschlagen, in einem ersten Schritt mehrere Product Owner – jeweils einen pro Team – einzuführen. Durch die Abhängigkeiten der Arbeit der einzelnen Teams entsteht Koordinationsbedarf unter den Teams bzw. ihren dedizierten Product Ownern. Ein übergeordneter Chief Product Owner verantwortet das Gesamtprodukt, zerlegt vom Kunden beauftragte Features[4] in kleinere Teile und verteilt sie an die Product Owner der Teams. Diese Organisationsstruktur deckt den entstandenen Koordinationsbedarf. Der Chief Product Owner bildet – im Rahmen seiner Gesamtverantwortung – auch nach der Umsetzung in den Teams und Abnahme durch die Product Owner wieder die schließende Klammer. Eine solche Struktur ist in der Praxis häufig anzutreffen. Erfahrungsgemäß kann sie gut funktionieren für eine Teamanzahl bis etwa sechs, sieben oder acht Teams je nach Komplexität der Aufgaben. Doch tut sie das auch für unbegrenzt viele?

Die geschaffene Organisationsstruktur stellt die Konsistenz des Gesamtprodukts sicher, aber mit steigender Anzahl an Teams muss sich irgendwann wiederum ein Engpass an der Stelle des Chief Product Owners bilden: Er kann nicht unbegrenzt viele Product Owner koordinieren. Es zeigt sich erneut ein Engpass gestoßen und dessen Auslastung

[2] Die Relevanz eines „on-site customers" wird schon mit dem Konzept des eXtreme Programming beschrieben, s. dazu [Ex01, BT04].
[3] [PP07], Kapitel 2.2, und [Re09], z.B. S. 32 ff, übertragen dies auf die Software- bzw. Produktentwicklung und konkretisieren.
[4] Ein „Feature" wird hier definiert als eine Gruppe von User Stories, deren Nutzen gemeinsam höher ist als Summe des Nutzens der einzelnen Stories. Es handelt sich um eine übergeordnete Ebene, in der betriebswirtschaftliche Betrachtungen, beispielsweise des ROIs, im Fokus stehen.

muss zur höchsten Priorität erhoben werden. Eine Möglichkeit dazu ist die Erweiterung der entstandenen Hierarchie um eine – oder bei ansteigender Anzahl an Teams um mehrere Ebenen. Es entsteht eine hierarchische Struktur wie in Abbildung 2 dargestellt. Die Leitungsspanne des Chief Product Owners wird so verkleinert.

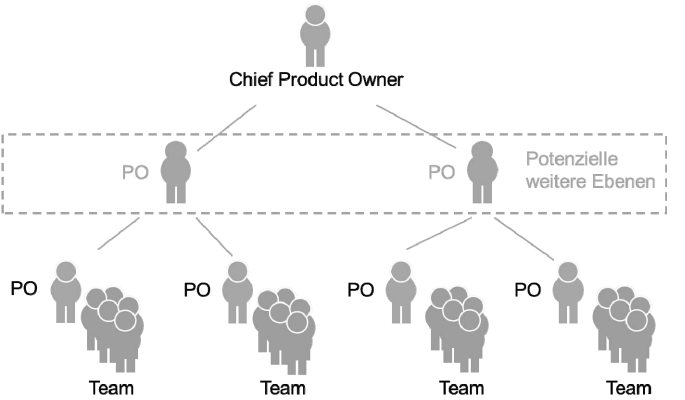

Abbildung 2: Wachsende hierarchische Product Owner-Struktur

Spätestens hier wird deutlich, dass dieses Modell seinerseits auf Grenzen stößt. Abbildung 3 zeigt, dass Entscheidungswege letztlich den Chief Product Owner aufgrund seiner Gesamtverantwortung mit einbeziehen. Der Koordinationsaufwand ist hoch und steigt mit weiteren Ebenen noch stark an. Die Product Owner warten auf Entscheidungen, was in der Begrifflichkeit der Theory of Constraints den Durchsatz irgendwann begrenzen muss. Auch hier bedeuten Wartezeiten Verschwendung in einem *Lean*-Sinne.

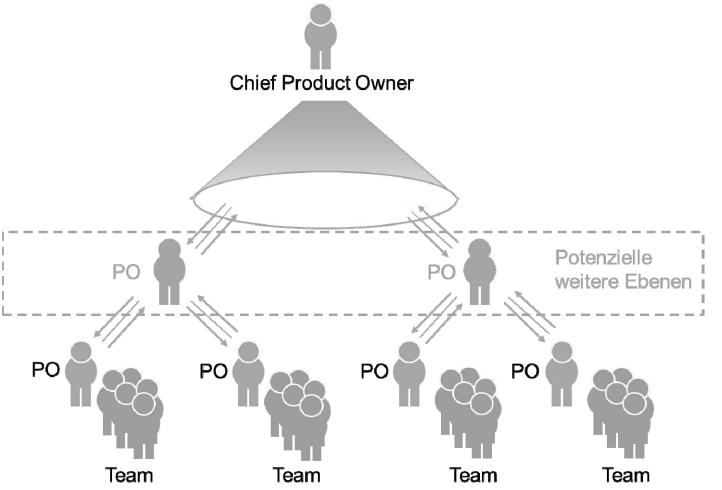

Abbildung 3: Steigender Koordinationsaufwand in einer wachsenden hierarchischen Struktur

Des Weiteren nimmt eine solche Struktur die Vorteile agilen Vorgehens wenn sie zu komplex wird: Je mehr Personen involviert sind, desto weniger kann ein Product Owner die Verantwortung und Aufgaben wahrnehmen, die ihm im Scrum-Framework zugedacht werden. Er hat nur noch sehr eingeschränkte Verantwortung für die Wertmaximierung in seinem Bereich, vieles ist bereits in der Hierarchie über ihm entschieden. Es besteht die Gefahr, dass zur Verteilung von Arbeitspaketen auf Product Owner und Teams vermehrt vorgelagerter Analyseaufwand ohne Beteiligung der Teams entsteht. Entgegen der agilen Idee wird „Big upfront design" betrieben, dessen Wert immer stärker verfällt während es auf die Umsetzung wartet.[5]

Diese Faktoren zusammen bewirken, dass ab einer gewissen Anzahl von Teams weitere Ressourcen kaum zusätzlichen Durchsatz bewirken können. Wiederum ist der Product Owner – genauer: die Hierarchie von Chief Product Owner und Product Owner – ein ausgelasteter Engpass.

4.3 Mittlere und große Organisationen: Skalierung durch eine neue Rolle

Das bisherige Vorgehen bei der Skalierung scheint an diesem Punkt gänzlich ausgereizt. Welche Möglichkeiten haben wir? Schauen wir uns dazu die Product Owner-Rolle noch einmal genauer an. Die Rollendefinition umfasst grob die folgenden Verantwortlichkeiten:

- Priorisierung der Anforderungen
- Unterstützung bei der Detaillierung von Anforderungen, Repräsentation des Kunden, Klärung von Detailfragen
- Unterstützung bei Konzeption und Design
- Durchführung der Akzeptanztests
- Stakeholdermanagement

Die in Abschnitt 4.1 beschriebene Singulariät der Product Owner-Rolle in Verbin-dung mit seinen breiten Verantwortlichkeiten ist der Grund für die Schwierigkeit, diese Rolle zu skalieren. Die Hypothese ist nun, mit der Verteilung der Aufgaben auf mehrere Rollen eine besser skalierende Struktur zu erreichen. Eine klassische funktionale Trennung auf mehrere Rollen, entsprechend der Funktionen Business Analyse, Architektur, Konzeption und Abnahme, erlaubt die Skalierung durch Spezialisierung. Jedoch ist ein solches Vorgehen nicht mehr agil: Es würde beispielsweise klassisches Requirements Engineering betrieben und Konzepte weit im Voraus und stark getrennt von der Umsetzung erstellt. Wir sind erneut beim oben erwähnten „Big upfront design". Unter der Voraussetzung einer veränderlichen Umwelt und veränderlicher, komplexer Anforderungen

[5] Reinertsen beispielsweise spricht von den durch Wertverfall entstehenden Kosten als „Holding Costs" und nennt sie als einen Parameter für die Bestimmung der optimalen „Batch Size", s. [Re09], 35f.

verfällt der Wert der Konzepte während sie auf ihre Umsetzung warten und stellen Verschwendung im *Lean*-Sinne dar.

Bleiben wir bei der Lösungssuche bei den *Lean*-Prinzipien: Ein zentraler Aspekt ist ein Empowerment der Basis [LeMa, LePr] und mehr Verantwortung soweit entlang einer Hierarchie herunter zu delegieren, bis genug Wissen für eine Entscheidung vorhanden ist. Eine solche Dezentralisierung und Delegation von Entscheidungen soweit abwärts wie möglich in einer Hierarchie ermöglicht die benötigte Skalierbarkeit. Eine grundsätzliche Empfehlung ist, Entscheidungen, die in unregelmäßigen Abständen zu treffen sind, große Auswirkungen nach sich ziehen oder signifikante Economies of Scale haben, zu zentralisieren und dagegen schnell zu treffende Entscheidungen, deren Effekte einem Wertverfall im Laufe der Zeit unterliegen, zu dezentralisieren [Re09]. Dementsprechend wird in diesem Beitrag vorgeschlagen, neben dem Product Owner eine weitere Rolle einzuführen, die die Verantwortlichkeiten übernimmt, die dezentralisiert werden können. Ein einziger Product Owner ist weiter verantwortlich für die Anforderungspriorisierung und bestimmt damit über das Produkt. Davon getrennt und in einer zweiten Rolle, dem „Story Owner“, werden die folgenden Aufgaben zusammengefasst:

- Die Detailabstimmung mit dem Kunden zu einer Story,
- deren Ausarbeitung zusammen mit dem Entwicklungsteam und die
- Funktion der Schnittstelle zum Kunden bei Rückfragen auf Story-Ebene.

Abbildung 4 zeigt den entstehenden Schnitt durch die Funktionen.

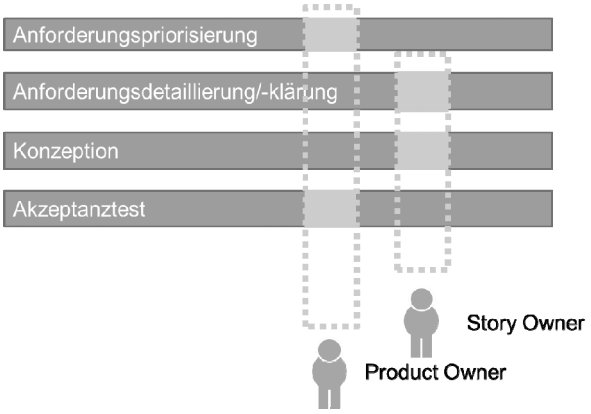

Abbildung 4: Vertikaler Schnitt in Rollen

Es ergibt sich eine Struktur wie in Abbildung 5 dargestellt: Mehrere Story Owner – jeweils einer pro Story untersteht einem zentralen Product Owner.

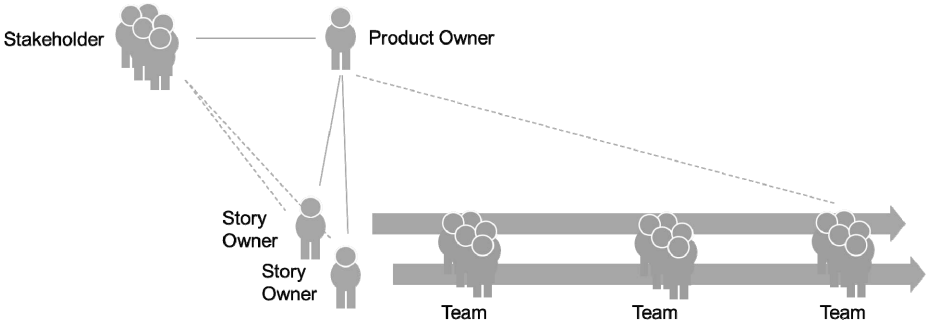

Abbildung 5: Story Owner-Struktur

Ein Story Owner ist nicht einem Team sondern einer Story dediziert zugeordnet. Er betreut diese möglicherweise über mehrere an der Umsetzung beteiligte Teams hinweg.[6] Erhöht sich die Anzahl an Teams, zieht dies also nicht notwendigerweise nach sich, dass mehr Story Owner benötigt werden. Andererseits können mit steigenden Kundenanforderungen flexibel weitere Story Owner hinzugefügt werden. Für jedes Feature und jede Story existiert mit dem Story Owner eine Person mit fachlichem Verständnis des Themas sowie der ungeteilten Verantwortung dafür, das Thema voranzubringen und mit den Teams zusammen im Detail auszugestalten. Er „repräsentiert" eine Story, ist die Schnittstelle zwischen Teams und Kunden in Bezug auf sein Thema und koordiniert sich eigenverantwortlich mit anderen Story Ownern im Fall von Schnittmengen.

Mit der Anzahl der konkreten Stellen, die die Story Owner-Rolle ausfüllen, begrenzt man den Work in Progress in der Struktur. Durch die Möglichkeit, nach Bedarf Story Owner hinzuzufügen, ist der Engpass durch die Kapazität des Product Owners bei der Skalierung aufgehoben.

Warum bleibt diese Struktur trotzdem schlank und agil? Die Herleitung der Aufgabentrennung in Product Owner und Story Owner basiert auf der Lean-Idee und dem Gedanken der Empowerment der Basis – in diesem Fall der Story Owner. Es werden genau die Entscheidungen an die Story Owner deligiert, die sie mit ihrer Nähe zur Umsetzung und zu der Umsetzung in den Teams am besten treffen können. Abbildung 6 verdeutlicht dies.

[6] Mit „Story" ist hier nicht notwendigerweise eine User Story gemeint, es kann sich auch um ein größeres zusammen-hängendes Feature handeln.

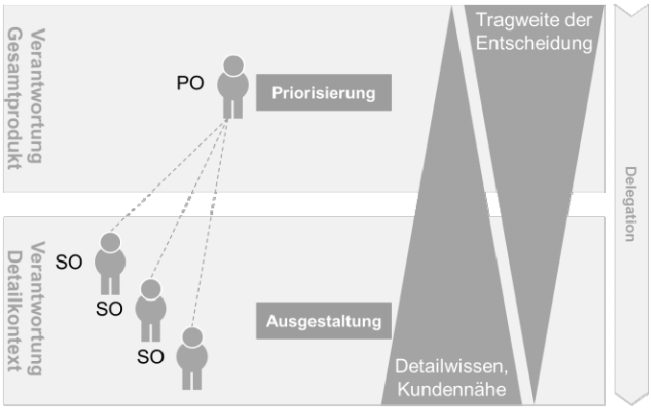

Abbildung 6: Verantwortungsbereiche von Product Owner und Story Owner

Eine Entscheidung oder Freigabe durch den Product Owner ist nicht notwendig. Die Story Owner stimmen sich auf ihrer Ebene direkt miteinander ab und arbeiten eng mit den Teams zusammen.

Diese beschriebene Struktur wird hier für eine Entwicklungsorganisation mit bis zu – in Abhängigkeit von der Produktkomplexität – mehreren Dutzend Teams vorgeschlagen. Bei hoher Komplexität oder noch größeren Organisationen kann ein Framework wie Leffingwell's *Scaled Agile Framework* [SAFe (für einen Überblick), Le07, Le11] Lösungsideen liefern.

5 Verschiedene Lösungen – je nach Organisationsgröße

In diesem Beitrag wird gedanklich das von der Theory of Constraints vorgeschlagene Vorgehen auf die Skalierung einer nach Scrum arbeitenden Entwicklungsorganisation angewandt. Entlang dieses Vorgehens wird argumentiert, dass der aufgezeigte Engpass an der Rolle des Product Owners auf verschiedene Weise gelöst werden kann. Je nach Größe der Entwicklungsorganisation genügt eine andere Struktur damit diese Rolle nicht den Durchsatz in der Organisation limitiert.

Ab einer gewissen Größe der Organisation aber verlangt die weitere Skalierbarkeit nach der Aufteilung der Verantwortlichkeiten des Product Owners auf zwei Rollen und damit der Einführung der Story Owner-Rolle, wodurch eine weiterhin schlanke und agile Struktur entsteht. Es stellt sich die Frage, warum das Story Owner-Konzept nicht in jedem Fall – schon bei kleinen Organisationen – eingesetzt werden sollte. Auch wenn diese Struktur die flexibelste der hier beschriebenen ist, zeigt der im Beitrag verfolgte Gedanke der Theory of Constraints dass ihre Einführung schlicht nicht nötig ist solange der Engpass auf eine der anderen beschriebenen Weisen noch nicht voll ausgelastet ist. Denn neben seiner Vorteile in der Skalierbarkeit birgt der Ansatz auch eine gewisse Komplexität.

Schlussendlich ist der durch die Product Owner-Rolle bei der Arbeit nach Scrum in großen Organisationen induzierte Engpass aufgelöst. Folgen wir der Theory of Constraints, so wird eine weitere Skalierung irgendwann wohl erneut durch einen Engpass limitiert; wahrscheinlich an ganz anderer Stelle. Die Theory of Constraints gibt uns die generischen Hinweise, wie wir auch damit umgehen können.

Literaturverzeichnis

[ExPr] http://www.extremeprogramming.org/rules/customer.html. Letzter Zugriff: 20.08.2014.

[BT04] Boehm, Barry W.; Turner, Richard: Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley, 2004.

[Go90] Goldratt, Eliyahu M.: Theory of Constraints. North River Press, 1. Aufl., 1990.

[Le07] Leffingwell, Dean: Scaling Software Agility – Best Practices for Large Enterprises. Addison Wesley, Pearson Education, 2007.

[Le11] Leffingwell, Dean: Agile Software Requirements – Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison Wesley, Pearson Education, 2011.

[LeMa] http://de.wikipedia.org/wiki/Lean_Management. Letzter Zugriff: 20.08.2014.

[LePr] http://de.wikipedia.org/wiki/Lean_Production. Zugriff: 20.08.2014.

[Oh88] Ohno, Taiichi: Toyota Production System: Beyond Large-scale Production. Productivity Press, 1988.

[PP07] Poppendieck, Mary B.; Poppendieck, Tom: Implementing Lean Software Development: From Concept to Cash. Addison Wesley, 2007.

[Re09] Reinertsen, Donald G.: The Principles of Product Development Flow – Second Generation Lean Product Development. Celeritas Publishing, 2009.

[SAFe] http://scaledagileframework.com. Letzter Zugriff: 20.08.2014.