Agiles Projektmanagement mit Scrum und Kanban – Ergänzung oder Konkurrenz?

Bachelorarbeit für die Prüfung zum
Bachelor of Art (B.A.)
Fakultät Wirtschaft
Duale Hochschule Baden-Württemberg Mannheim

Vorname Name:	Elisabeth Halm
Matrikel-Nummer:	[redacted]
Semesteranschrift:	[redacted]
Studiengang:	Digitale Medien - Medienmanagement und Kommunikation
Kurs:	[redacted]
Akademischer Gutachter:	[redacted]
Betrieblicher Betreuer:	[redacted]
Ausbildungsbetrieb:	[redacted]
Abgabetermin:	06. Juli 2020

Abstract

Die Bachelor Thesis befasst sich mit der Thematik des agilen Projektmanagements und soll klären, ob Scrum und Kanban konkurrierende Arbeitsweisen darstellen oder sich diese beiden Rahmenwerke sinnvoll ergänzen können. Im ersten Teil wird dem Leser[1] durch die Auflistung der theoretischen Merkmale und Eigenschaften von Scrum und Kanban die agile Arbeitsweise in ihren Grundzügen vorgestellt. Die Bachelorarbeit untersucht die Gemeinsamkeiten und Unterschiede beider Projektmanagementmethoden, um daraus resultierende Erfolgsfaktoren oder Ausschlusskriterien, die durch eine Verknüpfung entstehen könnten, aufzuzeigen.

Der praktische Teil der Arbeit befasst sich mit dem Projektmanagement bei [redacted], einem digitalen Anwaltssuchportal. Das Entwickler-Team des Portals nutzte Scrum als Projektmanagementmethode mit Erweiterungen durch die Kanban-Praktik, um damit die bereits bestehende Web-Plattform [redacted] zu etablieren und weiterzuentwickeln. Um die Ausführung des Projektmanagements beurteilen zu können, werden verschiedene Ansichten und Meinungen in Experteninterviews qualitativ untersucht, dargestellt und ausgewertet. Der Fokus wird dabei auf die Ergänzung von Scrum mit Kanban-Praktiken gelegt. Anschließend werden die im ersten Teil erarbeiteten Erkenntnisse einbezogen und darauf aufbauend mögliche Vor- und Nachteile betrachtet.

Die ermittelten Ergebnisse zeigen, dass Kanban und Scrum nicht nur theoretisch sinnvoll kombiniert werden können, sondern deren gemeinsamer Einsatz auch in der Praxis zu nachweisbaren Verbesserungen führt. Stärken können vereint und Schwächen reduziert werden. Während Scrum die dominierende Philosophie darstellt, kann Kanban als eine stark auf Visualisierung setzende Methode die Scrum-Prozesse ausgesprochen sinnvoll ergänzen.

[1] Soweit im Folgenden Berufs-, Gruppen- und / oder Personenbezeichnungen Verwendung finden, so ist auch stets die jeweils weibliche Form gemeint. Der Verfasser sieht aus Gründen der besseren Lesbarkeit bewusst von der Verwendung männlicher und weiblicher Sprachformen ab.

Inhaltsverzeichnis

Abstract II
Inhaltsverzeichnis III
Abkürzungsverzeichnis V
Abbildungs- und Tabellenverzeichnis VI
Glossar VII

I Einleitungsteil 1
1 Einleitung 1
1.1 Einführung in die Thematik und Begriffseinordnung 1
1.2 Problemstellung und Zielsetzung der Arbeit 5
1.3 Evaluationsmethodik und Vorgehensweise 6

II Theoretischer Teil 9
2 Kanban 9
2.1 Definition und Historie von Kanban 9
2.2 Kanban-Werte 10
2.3 Kanban-Grundprinzipien 12
2.3.1 Change-Management-Prinzipien 12
2.3.2 Service-Delivery-Prinzipien 13
2.4 Kanban-Kernpraktiken 14
2.4.1 Visualisiere 14
2.4.2 Limitiere die parallele Arbeit (WiP) 15
2.4.3 Manage den Arbeitsfluss 16
2.4.4 Mache Prozessregeln explizit 19
2.4.5 Implementiere Rückkopplungsschleifen 19
2.4.6 Verbessere gemeinsam, entwickle experimentell weiter 20
3 Scrum 21
3.1 Definition und Historie von Scrum 21
3.2 Scrum-Werte 23
3.3 Scrum-Rollen 25
3.3.1 Product Owner 25
3.3.2 Scrum Master 26

3.3.3 Entwicklungsteam 27
3.4 Scrum-Artefakte 28
3.4.1 Product Backlog 28
3.4.2 Sprint Backlog 31
3.4.3 Produktinkrement 32
3.5 Scrum-Zyklus 33
3.5.1 Sprint 34
3.5.2 Daily-Scrum 36
3.5.3 Sprint-Planning 36
3.5.4 Sprint-Review 37
3.5.5 Sprint-Retrospektive 37
4 Scrum und Kanban: Gemeinsamkeiten und Unterschiede 39

III Praktischer Teil 45
5 Projektmanagement bei [redacted]‘ 45
5.1 Projektvorstellung 45
5.2 Projektmanagementmethode und Arbeitsweise des [redacted]-Teams 46
5.3 Kombination von Lean und Agil in der Praxis 51
6 Potenzial der Erweiterung von Scrum mit Kanban 57
6.1 Empirische Erhebung (Vorgehensweise) 57
6.2 Ein hybrides Konzept (Auswertung) 58
6.3 Mögliche Erfolgsfaktoren (Schlussfolgerung) 62

IV Schlussteil 65
7 Fazit 65
7.1 Zusammenfassung 65
7.2 Ausblick 67

Danksagung X
Anhang XI
Literaturverzeichnis XXXII
Ehrenwörtliche Erklärung XXXV

Abkürzungsverzeichnis

Abb.	Abbildung
AG	Aktiengesellschaft
CFD	Cumulative-Flow-Diagram (Kumulatives Flussdiagramm)
DoD	Definition of Done
DoR	Definition of Ready
f.	folgend
ff.	fortfolgend
GmbH	Gesellschaft mit beschränkter Haftung
o.D.	ohne Datum
PBI	Product Backlog Item
PMI	Project Management Institute
S.	Seite
SWOT	Strengths (Stärken), Weaknesses (Schwächen), Opportunities (Chancen), Threats (Risiken)
WiP	Work in Progress

Abbildungs- und Tabellenverzeichnis

Abb. 1	Inkrementelle Auslieferung von Produkten	2
Abb. 2	Bausteine des Agilen Projektmanagements	3
Abb. 3	Begriffseinordnung	4
Abb. 4	Beispiel eines Kanban-Boards	14
Abb. 5	Push- vs. Pull-System	16
Abb. 6	Kumulatives Flussdiagramm (CFD)	18
Abb. 7	Das Gerüst von Scrum	24
Abb. 8	Scrum-Rollen	25
Abb. 9	Prodcut-Backlog mit Items	30
Abb. 10	Backlog-Artefakte	32
Abb. 11	Inkremente bei Scrum	33
Abb. 12	Scrum-Zyklus im Überblick	34
Abb. 13	Der Sprint	35
Abb. 14	Sprint-Board von [redacted] in Jira	48
Abb. 15	Cycle-Time-Streudiagramm von [redacted] mit PSK	53
Abb. 16	Work-Item-Age-Chart von [redacted]	54
Abb. 17	Throughput-Run-Chart von [redacted]	55

Tab. 1	SWOT-Analyse des Vorgehensmodells Kanban	39
Tab. 2	SWOT-Analyse des Vorgehensmodells Scrum	42

Glossar

Agil	Die Wortbedeutung ist ‚beweglich', ‚flink'. Im Geschäftsleben ist damit meist eine schnelle Reaktionsfähigkeit gemeint. (Kusay-Merkle 2018:312)
Agiles Manifest	Im Jahr 2001 niedergeschriebene Grundsätze und Werte zur agilen Softwareentwicklung. Das Agile Manifest umfasst vier grundlegende Werte und zwölf Prinzipien. (Kusay-Merkle 2018:311)
Artefakt	Wortbedeutung lt. Duden: von Menschenhand Geschaffenes; In Scrum werden damit spezifische Ergebnisse bezeichnet. (Backlog und Inkrement) (Kusay-Merkle 2018:312)
Backlog (Product oder Sprint Backlog)	Eine priorisierte Liste aus Projektanforderungen mit geschätzten Zeitangaben (Product Backlog) oder eine Liste mit Aufgaben, welche die Arbeit des Teams für einen Sprint definieren (Sprint Backlog) (Schwaber 2007:143f.)
Definition of Done (DoD)	Vereinbarung im Team, was zur Fertigstellung von Arbeitseinheiten zu tun ist. (Kusay-Merkle 2018:314)
Durchlaufzeit (Cycle-Time)	Die Durchlaufzeit misst die Dauer, die eine Anforderung benötigt, um alle Phasen einer Wertschöpfungskette vollständig zu durchlaufen. (Epping 2011:148)
Durchsatz (Throughput)	Die Anzahl an Arbeitseinheiten, die ein Prozess pro Zeiteinheit verlässt, unabhängig davon, ob fertiggestellt oder verworfen. (Anderson 2018:60)
Inkrement	Produktfunktionalität, die vom Team während jedes Sprints entwickelt wird. (Schwaber 2007:142)
Item (Product Backlog Item, PBI oder Element)	Funktionale Anforderung, nicht funktionale Anforderung und Probleme, die auf Basis der geschäftlichen Wichtigkeit und der vorhandenen

	Abhängigkeiten priorisiert werden. (Schwaber 2007:143)
Iteration	Ein Zyklus innerhalb eines Projekts. In Scrum besteht dieser Zyklus meist aus 30 hintereinander folgenden Kalendertagen, entsprechend einem Sprint. (Schwaber 2007:142)
Kumulatives Flussdiagramm (Cumulative Flow Diagram, CFD)	Ein Diagramm, das die kumulierte Anzahl der Erträge und Abgänge von einem Prozess oder Teilen eines Prozesses über einen bestimmten Zeitraum anzeigt. (Anderson 2018:62)
Lean	Lean kommt aus der Produktion und ist ein Set von Prinzipien, um eine bestimmte Qualität, Geschwindigkeit und Kundenausrichtung zu erreichen. Unnötige Arbeit (waste = Abfall/Ballast) wird eliminiert. (Böhm 2019:27)
Praktiken	Als agile Praktiken werden praktische kleine und große Werkzeuge, Methoden und Handlungsweisen bezeichnet, welche die praktische Art des Arbeitens erleichtern. (Böhm 2019:22)
Projektmanagement	Die Anwendung von Wissen, Können, Werkzeugen und Techniken auf Projektaktivitäten, um Projektanforderungen zu erfüllen. (Project Management Institute 2008:6)
Projekt	Unter einem Projekt wird ein zeitliches Unterfangen verstanden, um ein einzigartiges Produkt, eine individuelle Dienstleistung oder ein Ergebnis zu entwickeln. (Project Management Institute 2008:5)
Sprintziel (Sprint Goal)	Es ist die Vision des Sprints und definiert seinen Nutzen. Es hilft die Arbeit des Entwicklungsteams zu fokussieren. (Kusay-Merkle 2018:327)
Story Points	Einheitenloses Maß, um die Komplexität der Umsetzung von Product Backlog Items zu schätzen. Dabei werden

	neue Items mit einer bereits geschätzten Referenzstory des Teams verglichen. (Kusay-Merkle 2018:328)
Team	Eine funktionsübergreifende Gruppe von Menschen, deren Verwaltung in der eigenen Verantwortung liegt und die in jedem Sprint Software oder ein anderes auslieferbares Produkt entwickelt. (Schwaber 2007:144)
Ticket	Die visuelle Abbildung einer Arbeitseinheit auf dem Kanban-Board, alternativ als Karte oder Kanban bezeichnet. (Kusay-Merkle 2018:329)
User Story	Eine kurze Beschreibung einer Anforderung, die aus der Sicht des Nutzers definiert wird. (Kusay-Merkle 2018:329)
Velocity	Ausdruck für die Geschwindigkeit eines Scrum Teams. Sie errechnet sich aus der durchschnittlichen Anzahl von Story Points, die ein Team pro Sprint schafft. (Kusay-Merkle 2018:330)
Work-in-Progress (WiP, parallele Arbeit im System)	Die Arbeitseinheiten, die in das System oder einen betrachteten Status eingetreten sind, die aber noch nicht fertiggestellt oder verworfen wurden. (Anderson 2018:69)
Work-in-Progress-Limit (WiP-Limit)	Eine Prozessregel, die die Menge an Work in Progress (WiP) begrenzt, die sich in einem bestimmten Teil des Systems befinden darf. (Anderson 2018:69)

I Einleitungsteil

1 Einleitung

In der Einleitung werden der Gegenstand, sowie die Problemstellung und das Ziel der vorliegenden Bachelorarbeit vorgestellt. Dabei wird explizit auf den geplanten methodischen Ansatz und den weiteren chronologischen Verlauf eingegangen.

1.1 Einführung in die Thematik und Begriffseinordnung

Starke Begriffe bestimmen unsere Arbeitswelt. Nicht nur Agilität, sondern insbesondere agile Arbeitsmethoden wie Kanban, Scrum oder Design Thinking sind aktuell in aller Munde. Doch was bedeuten die hinter diesen Worten stehenden Methoden in der Praxis? Können sie Unternehmen dabei unterstützen, schneller Entscheidungen zu treffen und sich den Bedürfnissen des Marktes anzupassen?

Wendet man sich der Statistik zu, zeigt die Analyse von Kienbaum und StepStone vom Januar 2020, dass Kanban, Scrum und Co. tatsächlich schon einer Mehrheit der Beschäftigten bekannt sind und praktische Anwendung finden. Insgesamt setzt fast jedes dritte Unternehmen bereits agile Techniken ein, um den vielfältigen Aufgabenstellungen der digitalen Arbeitswelt effizient zu begegnen. Die Umfrage ergab, dass die bekanntesten agilen Arbeitsmethoden Kanban (34 Prozent) und Scrum (32 Prozent) sind.

Etwa 21 Prozent der Befragten nutzen zur Strukturierung ihrer parallel anfallenden Aufgaben ein Kanban-Board, die wohl bekannteste Tafel zur Abbildung eines Prozesses von der Planung bis zur Realisierung. Im Vergleich dazu arbeitet fast jeder Fünfte (18 Prozent) in einem nach Scrum organisiertem Team.[2]

Doch was ist Agilität überhaupt und wo hat sie ihren Ursprung?

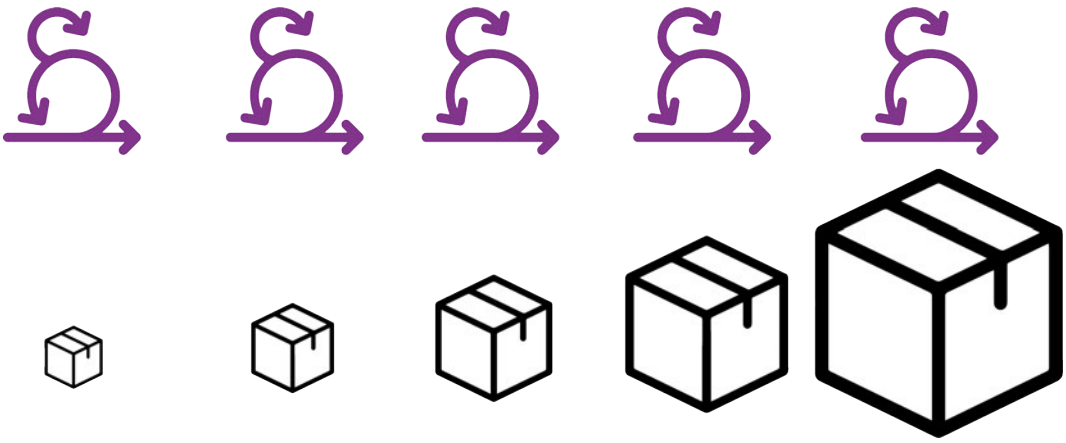
[2] Vgl. StepStone GmbH (Agile Unternehmen, 2020) [online]

Agilität ist ein weiter und kein neuer Begriff, der verschieden verstanden und interpretiert wird. Die vielen verschiedenen Facetten des Wortes in Publikationen und im beruflichen Sprachgebrauch lassen die Bedeutung oft verschwimmen.

Im Folgenden soll das Verständnis von Agilität geschärft werden, das dieser Arbeit zugrunde gelegt wird.

Agilität ist eine iterative Philosophie, welche den Produktentstehungsprozess inkrementell beschreibt und salopp mit Beweglichkeit übersetzt wird. Es wird eine schnelle und wiederholte Auslieferung von fertigen Erzeugnissen in den Mittelpunkt gestellt, um mit Kundennutzen Vertrauen zu erzeugen. Änderungen können dabei noch spät in den Entstehungsprozess eingebracht werden. Planungsaufwände werden minimiert und erst spät und nur für diejenigen Elemente geleistet, die unmittelbar zur Umsetzung anstehen.[3]

Abb.1: Inkrementelle Auslieferung von Produkten



Quelle: Eigene Darstellung in Anlehnung an Scaled Agile (SAFe® 5.0, 2020), S.39

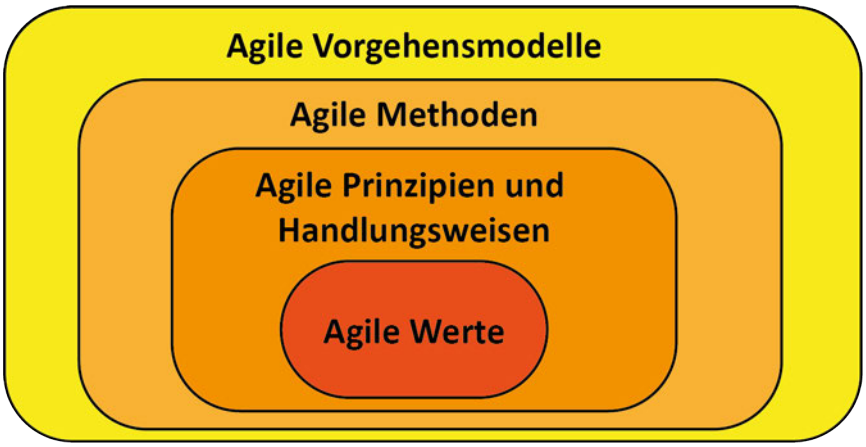
Das agile Projektmanagement stützt sich auf das im Jahr 2001 entwickelte *Agile Manifest*, das ein wichtiger Anhaltspunkt bei der Entwicklung von Software-Projekten ist und Scrum als eine essentielle agile Projektmanagementmethode verankert. Es beruht auf vier Werten sowie auf zwölf Prinzipien, die

[3] Vgl. Böhm, Janko (Erfolgsfaktor Agilität, 2019), S.30

zusammengenommen als Leitsatz für agile Arbeit gelten.[4] Eine Vertiefung der agilen Werte und Prinzipien, die hier aufgrund des klar vorgegebenen Rahmens der Arbeit leider nicht vorgenommen werden kann, findet man in der weiterführenden Literatur.

Das Ziel eines agilen Vorgehensmodells ist es, Projekte in einer bestimmten Art und Weise zu managen und sich dabei auf die agilen Werte, Prinzipien und Methoden zu stützen. Agile Werte bilden den Kern agiler Vorgehensmodelle und werden durch agile Prinzipien und Handlungsweisen konkretisiert. Auf welche Werte sich ein Einzelner, ein Team oder eine ganze Organisation fokussiert, hängt vom jeweiligen Umfeld und der Einschätzung der Beteiligten ab und kann sich auch im Laufe der Zeit ändern. Der wichtige Punkt dabei ist, dass überhaupt eine gemeinsame und einheitliche Klarheit über das grundlegende Wertesystem geschaffen wird.[5]

Abb.2: Bausteine des Agilen Projektmanagements

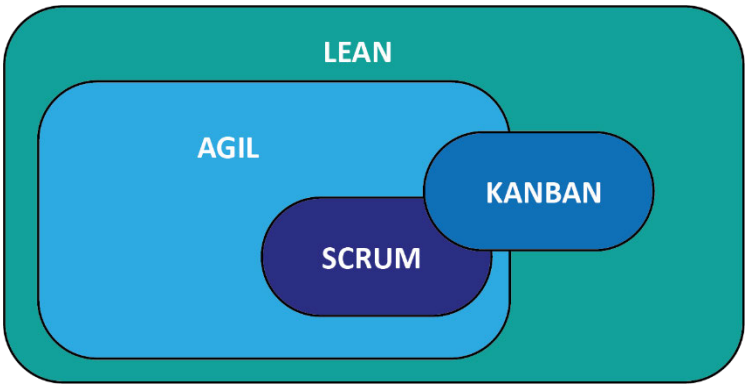


Quelle: Eigene Darstellung in Anlehnung an Preußig, Jörg (2018), S.143

Die Begriffe Lean, Agil, Scrum und Kanban werden teilweise nicht trennscharf gebraucht und sogar synonym eingesetzt. Eine erste Hilfestellung soll daher die Abbildung 3 geben, die die wesentlichen Begriffe, mit denen sich diese Arbeit beschäftigt, grafisch einordnet.

[4] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.26
[5] Vgl. Preußig, Jörg (Agiles Projektmanagement, 2018), S.143 f.

Abb.3: Begriffseinordnung



Quelle: Eigene Darstellung in Anlehnung an Böhm, Janko (Erfolgsfaktor Agilität, 2019), S.26)

Unter dem Begriff Lean, dessen wörtliche Übersetzung ‚schlank' bedeutet, wird ein Set von Prinzipien verstanden, um eine bestimmte Qualität, Geschwindigkeit und Kundenausrichtung zu erreichen. Unnötige Arbeit wird dabei eliminiert, der Prozess wird ‚verschlankt'.

Seine Ursprünge hat Lean in Japan mit dem sogenannten Toyota-Produktionssystem. Die Ideen von Vordenkern wie Taiichi Ohno haben die industrielle Fertigung weltweit revolutioniert und die Produktivität enorm gesteigert. Die Unternehmen haben diese Ansätze der Lean Production dann auf den Bereich der Softwareentwicklung übertragen, und mittlerweile ist die IT-Industrie zu einem wichtigen Vorreiter neuer Lean-Konzepte geworden.[6]

Die Lean Production lieferte auch wichtige Impulse für die agile Bewegung, die sich jedoch durch einige grundlegende Unterschiede von Lean abgrenzen lässt. Die Unterschiede und Gemeinsamkeiten von Lean und Agil sollen in dieser Arbeit anhand ihrer Vertreter Scrum und Kanban untersucht werden.

In der Praxis gelten häufig nicht nur Scrum, sondern auch Kanban als agile Vorgehensmodelle[7], wobei Kanban in der Theorie zur schlanken Softwareentwicklung (Lean Software Development) sortiert wird. Durch eine

[6] Vgl. Boes, Andreas et al. (Lean und agil im Büro, 2018), S.12 f.
[7] Vgl. Dechange, André (Projektmanagement, 2020), S.39

Betrachtung der charakterisierenden Elemente von Kanban wird jedoch deutlich, dass sowohl Werte des agilen als auch des Lean-Projektmanagements in Kanban verankert sind.[8] In den folgenden Kapiteln wird Kanban jedoch überwiegend als Vertreter des Lean Developments angesehen.

1.2 Problemstellung und Zielsetzung der Arbeit

Die Welt wird immer schneller, vielfältiger, ungeduldiger und digitaler. Die Menschheit will nicht mehr lange warten, bis ein neues Angebot vollständig erdacht, finanziert, designt und entwickelt ist, um eine neue Handy-App oder den neuen Küchenmixer in die Hände zu bekommen.

„Die zunehmende Digitalisierung zwingt Unternehmen, schneller auf Marktveränderungen zu reagieren und somit schneller Entscheidungen zu treffen. Agile Arbeitsmethoden können dabei sehr hilfreich sein“, sagt Stepstone CEO Dr. Sebastian Dettmers.[9]

Ein gemeinsames Verständnis für die Arbeit, die wir tun, einschließlich der Regeln, nach denen wir Arbeit verrichten, bildet die Grundlage, mit Verbesserungen beginnen zu können. Das Ziel eines Unternehmens ist es, rasch den Kurs korrigieren zu können, wenn sich Marktbedingungen ändern oder Probleme mit Abhängigkeiten auftreten.

Wie genau können agile Vorgehensweisen dabei unterstützen? Wie funktionieren Scrum und Kanban?

Um diese Fragen zu beantworten, besteht ein Ziel der Arbeit darin, die Rollen, den Prozess und die Praktiken der Scrum-Methodik sowie die Funktionsweise und Anwendung von Kanban zu analysieren und darzustellen.

---

[8] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.25
[9] Dettmers, Sebastian (Stepstone-Umfrage, 2020), S.13

„Doch am Ende werden agile Methoden nur dann zum Erfolg führen, wenn Unternehmen es zulassen und fördern, dass jeder einzelne Mitarbeiter frische Ideen entwickelt, Verantwortung übernimmt und eigenständig Entscheidungen trifft.“, beendet Dettmers seine Aussage.[10]

Eine dieser ‚frischen Ideen‘, die aufgekommen ist, ist die Kombination von Scrum mit Kanban. Daher soll in dieser Arbeit konkret auf die Forschungsfrage ‚Inwiefern können die beiden Projektmanagementmethoden Scrum und Kanban kombiniert werden, um einem Projekt strategisch zum Erfolg zu verhelfen?‘ eingegangen werden.

Bei der Beantwortung der Forschungsfrage ist die Analyse von Gemeinsamkeiten und Unterschieden beider Methoden wichtig. Existiert eine kausale Wechselbeziehung? Was bedingt sich? Was schließt sich aus? Anhand eines konkreten Projektes soll die Theorie in die Praxis transferiert werden, um so Rückschlüsse auf mögliche Ausschluss- oder Erfolgskriterien der Scrum-Kanban-Kombination ziehen zu können.

Ziel der vorliegenden Bachelor Thesis ist es, das Potenzial der Scrum-mit-Kanban-Methode am Beispiel des Projektmanagements von [redacted] aufzuzeigen. Im Ergebnis werden mögliche Erfolgsfaktoren des hybriden Vorgehensmodells präsentiert, die für zukünftige Projekte interessant sein können.

1.3 Evaluationsmethodik und Vorgehensweise

Im ersten Kapitel wird zunächst eine Einführung in das Thema gegeben. Es folgt die Darstellung der Zielsetzung und der Problemstellung sowie die Beschreibung der Vorgehensweise und Methodik der vorliegenden Arbeit.

[10] Dettmers, Sebastian (Stepstone-Umfrage, 2020), S.13

Im zweiten Kapitel beginnt der theoretische Teil der Bachelorarbeit. Zuerst wird das Vorgehensmodell Kanban mit den dazugehörigen Werten, Prinzipien und Kernpraktiken vorgestellt.

Das dritte Kapitel gibt eine Charakteristik des Vorgehensmodells Scrum mit den dazugehörigen Rollen, Werkzeugen und Events. Um die Gemeinsamkeiten und Unterschiede von Scrum und Kanban zu beleuchten, wurde ein Stärken-Schwächen-Profil beider Vorgehensmodelle erstellt. Mit diesem Vergleich von Scrum und Kanban im Kapitel 3.6 schließt der theoretische Teil dieser Bachelorarbeit.

Scrum und Kanban werden gemäß der Literatur erläutert und interpretiert. Hierbei sollen die unterschiedlichen Ausführungen verschiedener Autoren miteinander verglichen und Unterschiede aufgezeigt werden.

Auf Basis der theoretischen Vorarbeit wird der praktische Teil der Bachelorarbeit mit der Durchführung von Experteninterviews vervollständigt. In dieser Erhebungsmethode steht die Expertise eines Menschen im Vordergrund und das Ziel ist die Generierung von neuen Informationen und von praktischem Wissen im Rahmen des vorliegenden Sachverhalts.[11]

Im vierten Kapitel startet der Praxisteil der Arbeit. Hier wird zunächst das Projekt des Anwaltsportals [redacted] vorgestellt und danach die Projektmanagementmethode und Arbeitsweise des [redacted]-Teams beschrieben.

Im folgenden fünften Kapitel findet die empirische Untersuchung in Form von Experteninterviews statt. Dazu werden vier vom Autor ausgewählte Experten hinsichtlich verschiedener Scrum- und Kanban-Praktiken befragt, wobei konkret Bezug auf das Produkt [redacted] genommen wird. Anschließend werden die

[11] Vgl. Bogner, Alexander (Interviews mit Experten, 2014), S.17

Erkenntnisse aus den Experteninterviews ausgewertet und mögliche Schlussfolgerungen abgeleitet.

Abschließend folgt in Kapitel sechs ein Fazit, inklusive Zusammenfassung und Ausblick.

II Theoretischer Teil

2 Kanban

Dieses Kapitel behandelt das populäre, aus dem Lean Product Development stammende und zu Scrum einige Ähnlichkeiten aufweisende Vorgehensmodell Kanban. Darin sollen dem Leser die Werte, Prinzipien und Praktiken erläutert werden.

2.1 Definition und Historie von Kanban

Kanban ist ein Framework zur Steuerung von aufeinander aufbauenden Abläufen in einer Wertschöpfungskette und findet hauptsächlich Anwendung in der kontinuierlichen Prozessentwicklung. Im Mittelpunkt steht dabei die Visualisierung des Arbeitsablaufes.[12] Arbeiten, sowie Warteschlangen werden sichtbar gemacht, um sicherzustellen, dass ein Service die richtige Menge an Aufgaben bearbeitet. Hierzu wird ein Kanban-System verwendet – ein Fluss-System für Lieferungen, welches die Menge paralleler Arbeit (im Englischen auch mit Work in Progress oder kurz WiP bezeichnet) durch den Einsatz visueller Signale begrenzt.[13]

Das Wort Kanban stammt aus dem Japanischen, denn dort wurde dieses Vorgehensmodell erfunden und erstmalig angewendet. Es setzt sich aus den beiden Wörtern kan (übersetzt etwa Signal) und ban (übersetzt etwa Karte) zusammen. Diese Namensgebung enthält bereits eine Grundidee von Kanban: Verwende einfache und direkt erfahrbare Mittel, um bei Bedarf Aufmerksamkeit zu wecken.[14]

Seinen Ursprung fand Kanban in der Automobilindustrie. In der Mitte des 20. Jahrhunderts entwickelte der Japaner Taiichi Ohno für den Automobilhersteller Toyota eine neue Produktionsmethode, in der es vor allem darum ging, den Anteil an Ballast in der Produktion zu minimieren.

---

[12] Vgl. Böhm, Janko (Erfolgsfaktor Agilität, 2019), S.31
[13] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.1
[14] Vgl. Epping, Thomas (Kanban für die Softwareentwicklung, 2011), S.23

Diese Minimierung von Ballast beinhaltete insbesondere den Verzicht von unnötigem Materialvorrat. Stattdessen kamen Signalkarten zum Einsatz, die den Bedarf von Material genau zu dem Zeitpunkt anzeigten, an dem er tatsächlich benötigt wurde.

Nach diversen Weiterentwicklungen wurde diese Idee der schlanken Produktion Anfang des 21. Jahrhunderts auf den Bereich der Softwareentwicklung übertragen. Seither hat sie einen festen Platz im Sortiment der agilen Produktentwicklungsansätze.[15]

Öffentlich vorgestellt wurde Kanban als Vorgehensmodell der Lean Softwareentwicklung das erste Mal im Jahre 2007 von David Anderson, der seitdem als der Vater von Kanban in der Softwareentwicklung gilt.[16]

Anderson sah sich als Leiter eines Software-Teams bei der Firma Microsoft in den 2000er Jahren mit der Problematik konfrontiert, dass sehr viele Änderungsanfragen und Fehlerberichte in sein Team eingesteuert wurden, sodass das Team regelrecht verstopft war.[17] Erst durch den Einsatz von Kanban konnte die Durchlaufzeit für kleinere Anwendungserweiterungen und Fehlerbehebungen um 90 Prozent reduziert werden, und die vereinbarten Fälligkeitstermine wurden von dem Team mit einer Verlässlichkeit von 98 Prozent eingehalten. Ein entscheidender Faktor für den Erfolg war die Limitierung der Anzahl von Anwendungserweiterungen und Fehlerbehebungen, die das Team gleichzeitig bearbeitet hatte. Es stellten sich schnell kürzere Bearbeitungszeiten, eine gleichmäßigere Arbeitsbelastung, eine höhere Qualität und am Ende die erwartete Kundenzufriedenheit ein.[18]

2.2 Kanban-Werte

Die Kanban-Methode wird von Werten bestimmt. Dabei können laut Anderson die Werte in einem einzigen Wort ausgedrückt werden: Respekt. Es ist jedoch sinnvoll,

[15] Vgl. Epping, Thomas (Kanban für die Softwareentwicklung, 2011), S. 33 f.
[16] Vgl. Epping, Thomas (Kanban für die Softwareentwicklung, 2011), S.35
[17] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.45 f.
[18] Vgl. Epping, Thomas (Kanban für die Softwareentwicklung, 2011), S.35

dieses eine Wort zu einer Menge von neun Worten zu erweitern, die nachfolgend *Werte* genannt werden und das Wesentliche zusammenfassen. Warum gibt es die Prinzipien und Praktiken von Kanban und wie lauten die neun Werte?

Als Erstes wird der Wert *Transparenz* genannt, der der Überzeugung dienen soll, dass ein offener Informationsaustausch den Fluss von geschäftlichen Werten verbessert. Der zweite Wert ist die *Balance*. Sie steht für das Verständnis der verschiedenen Aspekte, Sichtweisen und Fähigkeiten, die untereinander ausgeglichen sein müssen, um eine hohe Leistungsfähigkeit zu erreichen. *Kollaboration* – gemeinsames Arbeiten – ist der dritte Wert und gleichzeitig der Kernpunkt der Kanban-Methode. Viertens, der *Kundenfokus*, ein Wert, der verhindert, dass das endgültige Ziel aus den Augen verloren wird. Der fünfte Wert, der *Arbeitsfluss*, stärkt die Erkenntnis, dass Arbeit einen dauerhaften oder gelegentlichen Fluss von Geschäftswerten darstellt. Nicht zu vergessen ist Nummer sechs, die *Führung*, das Festlegen strategischer Ziele und deren Durchsetzung, die auf allen Ebenen erforderlich ist, um die Lieferung von Werten und Verbesserungen zu erreichen. Der siebte Wert, das *Verständnis,* steht für die Selbsterkenntnis, vorwärtszukommen. Das Bekenntnis, gemeinsam Ziele zu verfolgen, dabei unterschiedliche Meinungen oder Herangehensweisen zu respektieren und sich, wenn möglich, entgegenzukommen, wird bei Kanban mit dem achten Wert, der *Vereinbarung,* festgehalten. Der letzte Wert *Respekt* – also die Wertschätzung, das Verständnis für und die Rücksichtnahme auf andere Menschen – bildet das Fundament, auf dem die anderen acht Werte beruhen.

Laut Anderson verkörpern diese Werte die Beweggründe von Kanban, das Bestreben, Dienstleistungen zu verbessern, die durch zusammenarbeitende Teams perfektioniert geliefert werden.[19]

[19] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.3 ff.

2.3 Kanban-Grundprinzipien

Kanban wird durch sechs Grundprinzipien definiert, die wiederum in zwei Gruppen unterteilen werden: die Change-Management-Prinzipien und die Service-Delivery-Prinzipien. Neben der Definition über die Werte ist das Verständnis der zugrundeliegenden Prinzipien von entscheidender Bedeutung für eine Wichtung des Kanban-Prozesses. Im Folgenden werden die beiden Prinzipiengruppen näher erläutert.

2.3.1 Change-Management-Prinzipien

Organisationen bestehen aus Individuen, die psychologisch und soziologisch vernetzt sind, von Natur aus jedoch Veränderungen eher skeptisch gegenüberstehen. Veränderungen machen die Zukunft ungewisser und werden häufig mit Gefahren und Risiken gleichgesetzt. Kanban erkennt diese menschlichen Aspekte und begegnet ihnen mit den drei Change-Management-Prinzipien.

Das wahrscheinlich bekannteste Prinzip von Kanban lautet: ‚*Beginne mit dem, was du gerade tust'.* In diesem Grundprinzip steckt einerseits, dass die aktuelle Arbeit erst beendet werden muss, bevor etwas Neues begonnen werden kann, andererseits enthält es die Aussage, dass Kanban sehr einfach und schnell eingeführt werden kann.

Das zweite Grundprinzip: ‚*Vereinbare, dass evolutionäre Veränderung verfolgt wird',* steht für eine essentielle Weiterentwicklung.

Verbesserung kann nur funktionieren, wenn sich alle Ebenen in der Organisation daran beteiligen. Damit wird das Grundprinzip: ‚*Fördere Führung auf allen Ebenen der Organisation'* durchgesetzt.[20]

[20] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.11 ff.

2.3.2 Service-Delivery-Prinzipien

Jede größere Organisation stellt, laut Kanban-Vater Anderson, im Grunde ein Netzwerk aus voneinander abhängigen Dienstleistungen dar. Kanban verdeutlicht dies mit drei Service-Delivery-Prinzipien. Diese Prinzipien richten sich stark nach dem hohen Wert der Service- sowie Kundenorientierung.

‚Verstehe und fokussiere die Bedürfnisse und Erwartungen der Kunden.‘ Der Kunde steht im Mittelpunkt des Geschehens und dessen Anforderungen und eventuelle Unzufriedenheit sollen als Motivatoren dienen, den Service mit den damit verbundenen Arbeitsabläufen und Prozessen zu verbessern. Wichtig ist herauszufinden, nach welchen Kriterien die Kunden beurteilen, ob sie mit den Dienstleistungen der Unternehmen zufrieden sind oder nicht.

Das zweite Prinzip lautet *‚Manage die Arbeit, lass die Menschen sich selbst organisieren‘*. Es steht für die Fokussierung auf die Arbeit an sich als gesamten Prozess, anstatt auf die einzelnen Mitarbeiter und deren Arbeitszeiten. Die Bearbeitungsdauer kreativer Wissensarbeit ist ein nicht-bestimmbares Problem.

Das Denken in vorgegebenen Bahnen, die in Richtlinien ausgedrückt werden, hilft den Mitarbeitern, sich an den gerade beschriebenen Change-Management-Prinzipien auszurichten. Dafür steht das dritte Grundprinzip: *‚Entwickle Regeln, um die Ergebnisse für die Organisation und deinen Kunden zu verbessern.‘* Wenn ein Prozess nicht mehr als Methode aus definierten Arbeitsabläufen, Rollen und Verantwortlichkeiten betrachtet wird, sondern als eine Abfolge von Regeln für die Entscheidung, wie Arbeit behandelt werden soll, wird die Konzentration auf die Arbeit gelegt.[21]

[21] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.13

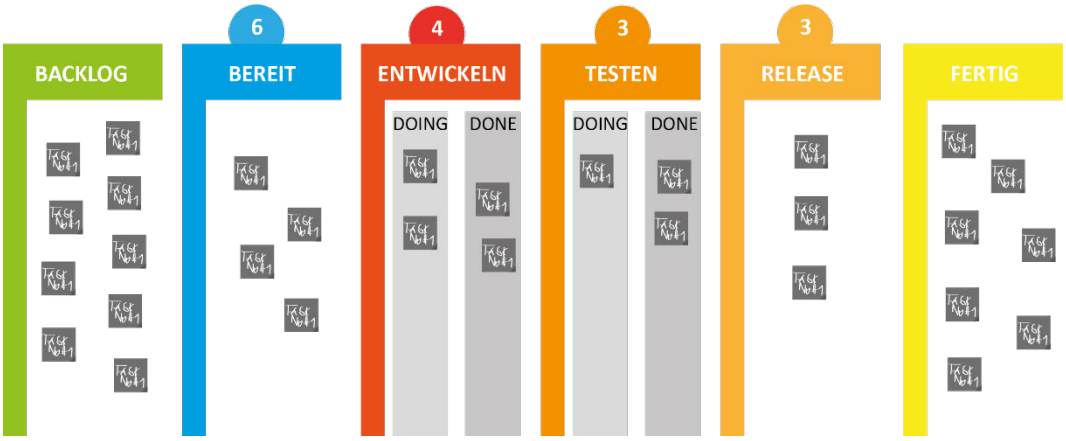
2.4 Kanban-Kernpraktiken

Die Kernpraktiken von Kanban definieren wichtige Aktivitäten für alle, die Kanban-Systeme managen. Es gibt sechs davon, die in den folgenden Unterkapiteln vorgestellt werden.

2.4.1 Visualisiere

Visualisiere, was sich visualisieren lässt! Ein Kanban-Board, wie jenes in Abbildung 4, ist wohl die bekannteste Möglichkeit, Arbeit und deren Ablauf darzustellen.
Neben dem Ablauf werden auch Vereinbarungen über den Arbeitsprozess festgehalten und die WiP-Limits auf dem Board angezeigt, um die parallel anfallenden Arbeiten in jeder Phase zu begrenzen.[22]

Abb.4: Beispiel eines Kanban-Boards



Quelle: Eigene Darstellung in Anlehnung an Dechange, André (Projektmanagement, 2020), S.307

Den Vorgang ‚Arbeit', sowie den Prozess sichtbar zu machen – sei es auf einem physischen Board im Büro oder einem elektronischen Board wie zum Beispiel mit dem Online-Tool Jira – ist das Ergebnis einer längeren, wichtigen Zusammenarbeit, um das derzeitige System zu verstehen und mögliche Bereiche für Verbesserungen zu finden. [23]

[22] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.21
[23] Vgl. Anderson, David (Die Essenz von Kanban, 2018), S.21

Die unterschiedlichen Stationen (wie zum Beispiel Programmierung, Dokumentation, Test, Inbetriebnahme) werden als Spalten dargestellt. Die einzelnen Anforderungen (Tasks, Features, User Stories etc.) werden auf Karteikarten oder Haftnotizen festgehalten und durchwandern mit der Zeit als sogenannte Tickets das Kanban-Board von links nach rechts. Die Gestaltung des Boards kann stark zwischen den unterschiedlichen Kanban-Systemen variieren. Das Vorgehensmodell selbst schränkt die Gestaltung dabei nicht ein.[24]

Ein weiterer wesentlicher Aspekt der Visualisierung ist die Gestaltung der Karte oder des Feldes, das eine Arbeitseinheit beschreibt. Wenn Elemente durch Abhängigkeiten von anderen Services oder aus anderen Gründen blockiert sind, sollte dies optisch hervorgehoben werden.[25]

2.4.2 Limitiere die parallele Arbeit (WiP)

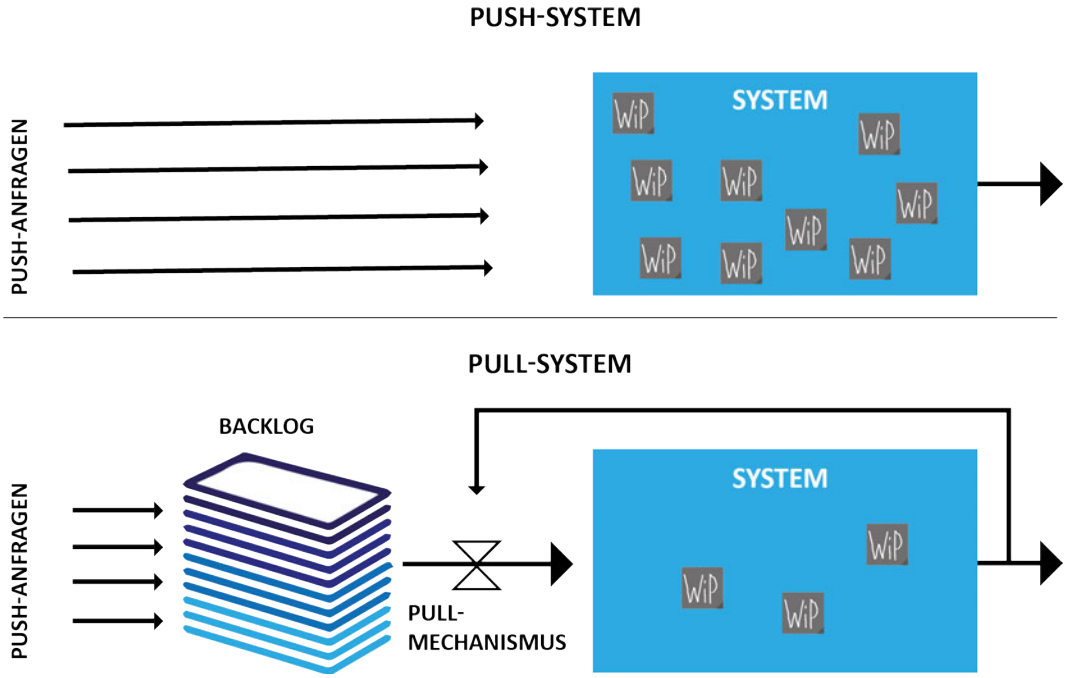
Die Einführung und Beachtung von Grenzen in Bezug auf die parallele Arbeit (WiP) im System sorgen dafür, dass keine neuen Einheiten begonnen werden, bevor die gerade anliegende Arbeit nicht fertiggestellt ist. Eine Anhäufung von nur teilweise erledigter Arbeit bedeutet nicht nur Verschwendung von Arbeitszeit, sondern es verlängert auch die Durchlaufzeiten deutlich und hält Organisationen davon ab, gegenüber sich ändernden Umständen und Möglichkeiten reaktionsfähig zu bleiben.[26]

Durch die Mengenlimitierung der angefangenen Arbeit entsteht ein Pull-System. Jede Station holt / zieht sich ihre Arbeit bei ihrem Vorgänger, anstatt fertige Arbeit an die nächste Station zu übergeben (‚to push'). Das Pull-Prinzip fördert Selbstorganisation: die Mitarbeiter ziehen die Aufgaben selbständig an sich, statt sie vom Projektmanager zugewiesen zu bekommen.[27]

---

[24] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 306
[25] Vgl. Anderson, David (Kanban, 2018), S.21 f.
[26] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 307
[27] Vgl. Anderson, David (Kanban, 2018), S.22

Abb.5: Push- vs. Pull-System



Quelle: Eigene Darstellung in Anlehnung an Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.117

Eine Beschränkung der Arbeitseinheiten pro Phase verbessert zusätzlich den Workflow im Sinne einer kürzeren Durchlaufzeit (im Englischen unter Cycle-Time bekannt) und eines höheren Durchsatzes (im Englischen der sogenannte Throughput). Die Durchlaufzeit ist die Dauer, die ein Arbeitselement pro Phase und in der Summe bis zur Fertigstellung benötigt – das heißt, es können die Durchlaufzeit pro Phase für eine Arbeitseinheit und die Gesamtdurchlaufzeit gemessen werden. Der Durchsatz ist die Anzahl von Einheiten, die pro Zeitraum (z.B. pro Woche) abgearbeitet werden.[28]

2.4.3 Manage den Arbeitsfluss

Der Arbeitsfluss in einem Kanban-System sollte die Lieferung von Geschäftswert (Business Value) maximieren, Durchlaufzeiten minimieren und so gleichmäßig wie möglich sein. Das sind manchmal widersprüchliche Ziele.

[28] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 307

Ein gleichmäßiger Arbeitsfortschritt ist nach der Theorie von Kanban essentiell, weil durch die Gewährleistung eines kontinuierlichen Arbeitsflusses die höchstmögliche Arbeitsproduktivität erreicht werden kann. Es wird Synchronizität auf allen Arbeitsgebieten erreicht und Staus werden vermieden. Die Beseitigung von Ballast ist wichtig, muss sich aber dem gleichmäßigen Arbeitsfortschritt unterordnen.

Weil der Geschäftswert von Arbeit erst dann erzielt wird, wenn die Arbeit das Ende der Wertschöpfungskette erreicht, wird in Kanban auch der Zeitraum, in dem eine Person mit einer Arbeitseinheit beschäftigt ist, als Ballast angesehen. Vor diesem Hintergrund haben sich in Kanban die folgenden typischen Messgrößen - sogenannte Kanban-Flowmetriken - für die Arbeit an Arbeitselementen etabliert.[29]

Die Durchlaufzeit (Cycle-Time) bezeichnet, wie bereits erwähnt, den Zeitraum vom tatsächlichen Beginn bis zum Ende der Arbeit an einem Arbeitspaket, gemessen über die gesamte Wertschöpfungskette hinweg. Im Gegensatz dazu bezeichnet die Vorlaufzeit (Lead Time), die Dauer ab dem Zeitpunkt, zu dem eine neue Aufgabe angefordert wird, und endet, wenn sie abgeschlossen ist. Mit anderen Worten, mithilfe der Vorlaufzeit kann analysiert werden, ob Arbeitselemente zu lange warten, bevor sie ausgeführt werden. Die Durchlaufzeit hilft zu verstehen, wie viel Zeit für die tatsächliche Ausführung einer bestimmten Aufgabe benötigt wird.[30]

Durch die Messung des Durchsatzes kann die Durchlaufrate (Average Completion Rate) berechnet werden. Diese bezeichnet die durchschnittliche Anzahl der Items, an denen innerhalb eines festen Zeitraumes die Arbeit beendet wird (die also das Ende der Wertschöpfungskette erreichen).

Zwischen diesen Messgrößen besteht der folgende Zusammenhang, der von John D. C. Little im Jahr 1961 formuliert wurde und seitdem als Little’s Law bekannt ist:

---

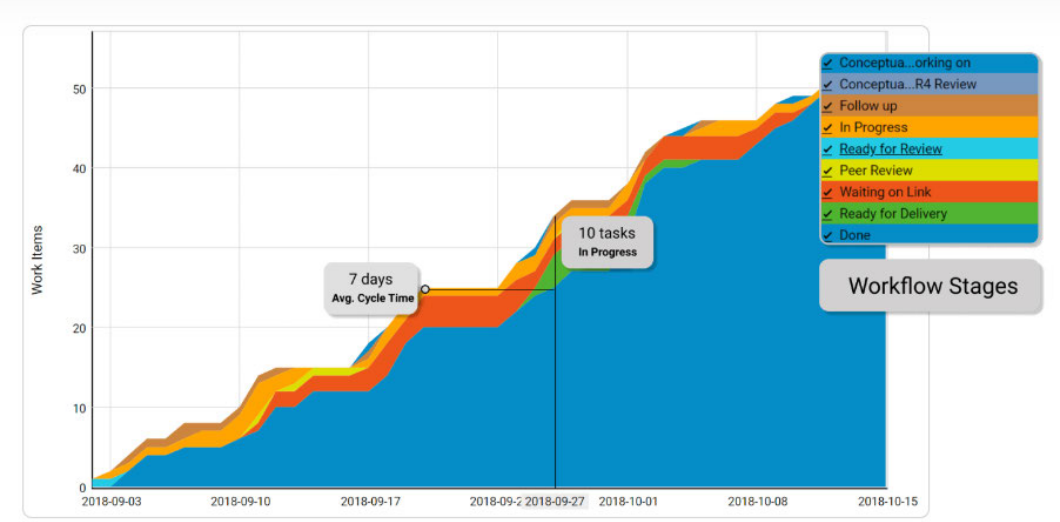
[29] Vgl. Epping, Thomas (Kanban für die Softwareentwicklung, 2011), S.28
[30] Vgl. Kanbanize (Kanban-Analysen, 2020) [online]

$$\text{Cycle-Time} = \frac{\text{Work in Progress}}{\text{Average Completion Rate}}$$

Littles Gesetz zeigt, dass für einen bestimmten Prozess mit einem bestimmten Durchsatz im Allgemeinen gilt: Je mehr Aufgaben zu einem bestimmten Zeitpunkt (im Durchschnitt) gleichzeitig bearbeitet werden, desto länger dauert es, diese Arbeiten (im Durchschnitt) zu beenden.[31]

Littles Gesetz kann grafisch mittels eines kumulativen Flussdiagramms (Cumulative Flow Diagram) dargestellt werden, wie in Abbildung 6 gezeigt, in dem die kumulative Anzahl der Elemente aufgetragen wird, die in einem System ankommen und dieses wieder verlassen.[32]

Abb.6: Kumulatives Flussdiagramm (CFD)



Quelle: Kanbanize (Kanban-Analysen, o.D.) [online]

Um das Kanban-System zu verbessern, müssen die ausgewählten Kennzahlen berechnet und das System entsprechend gesteuert werden. Im Kanban liegt der Fokus der Messungen und Optimierungen auf dem System, nicht auf einzelnen Mitarbeitern.[33]

[31] Vgl. Anderson, David (Kanban, 2018), S.17
[32] Vgl. Anderson, David (Kanban, 2018), S.18
[33] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 309

2.4.4 Mache Prozessregeln explizit

Ein Prozess, der über Workflow und Prozessregeln ausgedrückt wird, schafft Randbedingungen für Handlungen, ermächtigt innerhalb dieser Einschränkungen und führt zu neu auftretenden Eigenschaften, die mithilfe von Versuchen optimiert werden können. Die Prozessregeln müssen knapp, einfach, klar definiert und sichtbar sein, immer direkt und offen angewandt werden und leicht veränderbar durch jene sein, die einen Service erbringen.[34]

WiP-Limitierungen zu setzen und diese dann nie infrage zu stellen, zu ändern, um herauszubekommen, ob andere Limits unter anderen Umständen zu besseren Ergebnissen führen, wäre eine unzureichende Anwendung dieser Praktik.[35]

WiP-Limits sind eine Art von Prozessregeln. Andere Formen sind die Zuteilung und Balance von Kapazitäten, die *Definition of Done* (Regelung, die angibt, wann eine Arbeit als fertig betrachtet wird), Kommunikationsregeln oder andere Richtlinien für beispielsweise die Auswahl neuer Arbeit.[36]

Die gemeinsame Formulierung von Regeln schafft eine Basis für die erfolgreiche Zusammenarbeit der Beteiligten und Transparenz im Arbeitsprozess.[37]

2.4.5 Implementiere Rückkopplungsschleifen

Rückkopplungsschleifen sind ein wesentlicher Bestandteil jedes gesteuerten Prozesses und besonders wichtig für eine evolutionäre Veränderung. Die Verbesserung von Feedback ist in allen Bereichen des Prozesses wichtig.[38]

Alle Rückmeldungen aus dem System, insbesondere die Kennzahlen der Messungen, aber auch Hindernisse oder Ideen zur Verbesserung des Workflows

[34] Vgl. Anderson, David (Kanban, 2018), S.25
[35] Vgl. Anderson, David (Kanban, 2018), S.25
[36] Vgl. Anderson, David (Kanban, 2018), S.26
[37] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 309
[38] Vgl. Anderson, David (Kanban, 2018), S.27

sollten gesammelt und gemeinsam im Rahmen eines Verbesserungsprozesses ausgewertet und umgesetzt werden. Es gibt verschiedene Kanban-Elemente, die den Prozess kontinuierlich verbessern sollen. Die Auswertung der gemessenen Kennzahlen kann schnell Transparenz in den Entwicklungsprozess bringen, sodass Verbesserungen identifiziert und bewertet werden können. Aber auch Feedback-Meetings können konkrete Hinweise oder Ideen zur Prozessverbesserung liefern.[39]

2.4.6 Verbessere gemeinsam, entwickle experimentell weiter

Kanban ist ursprünglich eine Methode zur kontinuierlichen Prozessentwicklung. Kanban beginnt mit der Organisation, so wie sie jetzt ist, und nutzt das Fluss-Paradigma von Lean, um kontinuierliche und inkrementelle Verbesserung zu verfolgen. Es gibt für derartige Veränderungsprozesse keinen Endpunkt, da Perfektion in einer sich ständig ändernden Landschaft unerreichbar ist.[40]

Dabei ist essenziell, dass Feedback-Zyklen mehrere Personen mit einzubeziehen. Denn wenn ein Team selbstorganisiert arbeitet, entwickeln sich Verbesserungen und Innovationen aus Kooperationen.

Das Spannende an Kanban ist seine Vielseitigkeit und sein evolutionärer Ansatz. Der Fokus liegt dabei auf einer kontinuierlichen Verbesserung im Arbeitsprozess und steigert so den Geschäftswert, indem Kosten durch Effizienzsteigerungen reduziert werden und Ballast abgebaut wird.[41]

[39] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S. 310
[40] Vgl. Anderson, David (Kanban, 2018), S.30
[41] Vgl. Kusay-Merkle, Ursula (Agiles Projektmanagement im Berufsalltag, 2018), S.42

3 Scrum

Dieses Kapitel gibt eine Charakteristik des agilen Projektmanagementframeworks Scrum. Dabei wird der Begriff Scrum definiert, die verschiedenen Rollen und Voraussetzungen werden näher erläutert und anschließend werden die Scrum-Artefakte vorgestellt. Wie Zahnräder in einem Getriebe greifen in einem auf Scrum basierenden Entwicklungsprozess drei Rollen, drei Artefakte und fünf Events ineinander. Das Kapitel schließt mit dem Scrum-Zyklus, der alle bis dahin erläuterten Punkte zusammenfasst und deren Zusammenhänge darstellt. Eine anschließende Gegenüberstellung der Vorgehensweisen Scrum und Kanban soll einen Überblick über Gemeinsamkeiten und Unterschiede geben.

3.1 Definition und Historie von Scrum

Scrum ist ein agiles Projektmanagementframework, also ein Rahmen oder Gerüst für das Bearbeiten von komplexen Projekten, das erstmalig 1999 von Ken Schwaber eingeführt wurde. Laut Scrum Guide wird es wie folgt definiert: „A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value“.[42]

Scrum ist weder ein Prozess, noch eine Technik oder vollständige Methode. Das Rahmenwerk bringt verschiedene, aufeinander abgestimmte Prozesse und Techniken zum Einsatz. Es ist gedacht für die Entwicklung komplexer Produkte in einem komplexen Umfeld, wo *Inspect and Adapt*, also häufiges Überprüfen und darauf aufbauendes Lernen sowie Anpassen besonders wichtig sind.[43]

Im Scrum Guide, der von Ken Schwaber und Jeff Sutherland gepflegt und veröffentlicht wird, werden die Kernelemente von Scrum erläutert. Scrum sollte in der Praxis durch Praktiken und Tools individuell erweitert werden, da nicht alle Eventualitäten im Scrum Guide beschrieben sind.[44]

---

[42] Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.3
[43] Vgl. Kusay-Merkle, Ursula (Agiles Projektmanagement im Berufsalltag, 2018), S.34 f.
[44] Vgl. Böhm, Janko (Erfolgsfaktor Agilität, 2019), S.30

Scrum wurde ursprünglich nur in der Softwarenentwicklung eingesetzt. Es wird von der Annahme ausgegangen, dass es in dieser Branche kein wirklich fertiges Produkt gibt. Das ist die Grundlage dieses Verfahrens, es lebt und lernt. Es gibt also immer Iterationen, die ausgeliefert werden, als Ergebnis eines Entwicklungsprozesses und gleichzeitig als Beginn eines neuen. Scrum ist darauf angelegt, dieses Denken in Zyklen in feste Abläufe zu bringen. Dabei ist es als Vorgehensmodell so vielseitig, dass die hier genannten Abläufe nur das grobe Grundmuster widerspiegeln.[45]

Ursprünglich ist Scrum ein Begriff aus dem Rugby und heißt übersetzt Gedränge. Damit betont der Begriff ein wesentliches Merkmal des agilen Projektmanagements, nämlich die Teamarbeit mit einer zeitnahen und effizienten Kommunikation.[46] Die geistigen Väter von Scrum, Ken Schwaber und Jeff Sutherland, beschäftigten sich zu Beginn der 1990er-Jahre zunächst unabhängig voneinander, später gemeinsam mit der Frage, wie Produkte schneller und flexibler entwickeln werden können. 1999 veröffentlichten sie zusammen mit anderen Autoren den Artikel ‚Scrum: An extension pattern language for hyperproductive software development'. Seitdem sind unzählige Veröffentlichungen über Scrum und agile Softwareentwicklung erschienen. [47] Mit der zunehmenden Verbreitung von Scrum hat das Framework mittlerweile auch in vielen anderen Bereichen Einzug gehalten – von der Produktentwicklung über den Dienstleistungsbereich bis hin zur Verwaltung und dem Management von Organisationen.[48]

Sein immer breiter werdendes Einsatzspektrum erweckt bei einigen Menschen den Anschein, dass sich mit Scrum selbst komplexe Probleme sehr leicht lösen lassen. Das ist leider nicht der Fall – Scrum ist kein Allheilmittel, dennoch ein sehr hilfreiches Instrument: Scrum kann keine Risiken verhindern, macht sie aber frühzeitig transparent und somit kontrollierbar.[49]

---

[45] Vgl. Linz, Tilo (Testen in Scrum-Projekten, 2013), S.11 f.
[46] Vgl. Dechange, André (Projektmanagement – Schnell erfasst, 2020), S.290
[47] Vgl. Dräther, R.; Koschek, H.; Sahling, C. (Scrum – kurz & gut, 2019), S.4 f.
[48] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.4
[49] Vgl. Dräther, R.; Koschek, H.; Sahling, C. (Scrum – kurz & gut, 2019), S.1

Eine kurze Einleitung über die fünf Scrum-Werte (Verpflichtung, Fokus, Offenheit, Respekt und Mut), gefolgt von den drei Scrum-Prinzipien (Transparenz, Überprüfung und Anpassung) soll im Folgenden das Fundament für die spätere Analyse des Scrum-Frameworks schaffen.

3.2 Scrum-Werte

Anders zu denken, beginnt bei den Werten. Scrum sorgt für einen nachhaltigen Prozess. Aus diesem Grund ist der optimale Wert nicht einfach der in Praxis oder Theorie höchstmöglich erreichbare Wert, sondern vielmehr der teamspezifisch höchstmögliche Wert, dessen Niveau das Team unbegrenzt halten kann, ohne zu ermüden oder dessen Kräfte über die Maße zu verschleißen. Deshalb beschreibt der Scrum Guide ein Fundament aus fünf Werten, auf dem Scrum aufbaut: Selbstverpflichtung (Commitment), Mut (Courage), Fokus (Focus), Offenheit (Openness) und Respekt (Respect).[50]

Darüber hinaus kann ein Scrum-Team zusätzlich eigene Werte entwickeln oder sie aus dem Wertekanon der Organisation beziehen, in die das Team eingebettet ist.[51]

Die Basis von Scrum ist die Theorie empirischer Prozesssteuerung. Unter Empirie wird verstanden, „dass Wissen aus Erfahrung gewonnen wird und Entscheidungen auf Basis des Bekannten getroffen werden.“[52]

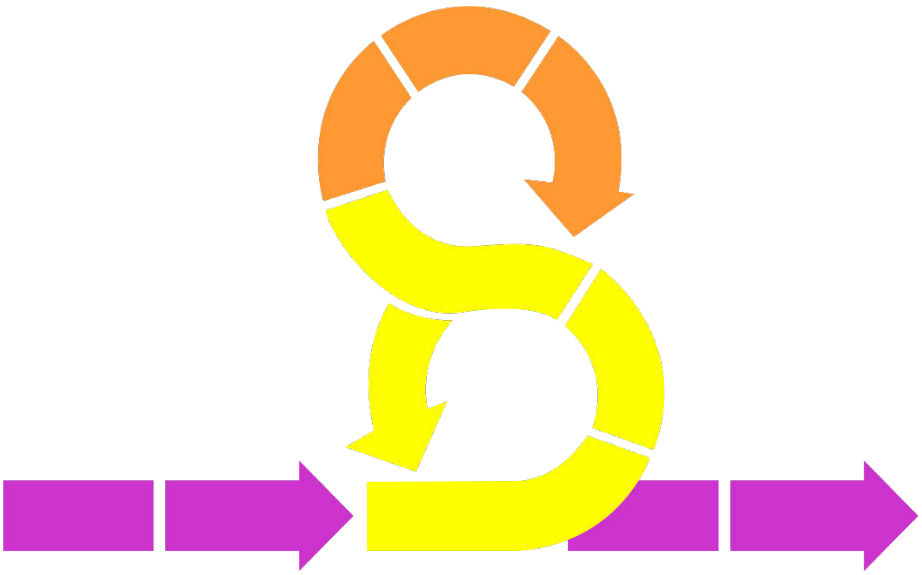
Die drei Säulen Transparenz (Transparency), Überprüfung (Inspection) und Anpassung (Adaption) tragen jede für sich eine empirische Prozesssteuerung. Aufgrund der engen Zusammenarbeit sowie des täglichen Austausches im Scrum-Team gibt es eine hohe Transparenz bezüglich der Arbeit, des Fortschritts und der Hindernisse. Des Weiteren werden in festgelegten Zyklen funktionierende Teilergebnisse übergeben und getestet. Auf Basis dieser Überprüfung erfolgt eine

[50] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.5
[51] Vgl. Dräther, R.; Koschek, H.; Sahling, C. (Scrum – kurz & gut, 2019), S.11
[52] Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.4

Anpassung der Anforderungen. Damit ergibt sich nach jedem Zyklus eine dem Bedarf und den Wünschen des Kunden angepasste aktuelle Anforderungsliste.[53]

In Scrum sind alle Vorgehensweisen an einem iterativen, inkrementellen Prozessrahmen festgemacht, der in Abbildung 7 schematisch dargestellt wird. Der untere Kreis stellt eine Iteration von Entwicklungsaktivitäten dar, die nacheinander stattfinden. Das Ergebnis jeder Iteration ist ein Inkrement des Produkts. Der obere Kreis repräsentiert die tägliche Überprüfung (Inspection), die während der Iteration durchgeführt wird und in der sich die einzelnen Mitglieder des Teams treffen, um die Aktivitäten gegenseitig zu untersuchen und die entsprechenden Anpassungen (Adaption) vorzunehmen. Dieser Zyklus wird Sprint genannt und wiederholt sich während der gesamten Projektphase.[54]

Abb. 7: Das Gerüst von Scrum



Quelle: Eigene Darstellung in Anlehnung an Schwaber, Ken (Agiles Projektmanagement mit Scrum, 2007), S. 6

Scrum ist im Wesen ein sehr strukturiertes Vorgehensmodell, das, wie bereits erwähnt, durch die drei Bestandteile Rollen, Artefakte (Werkzeuge) und Ereignisse

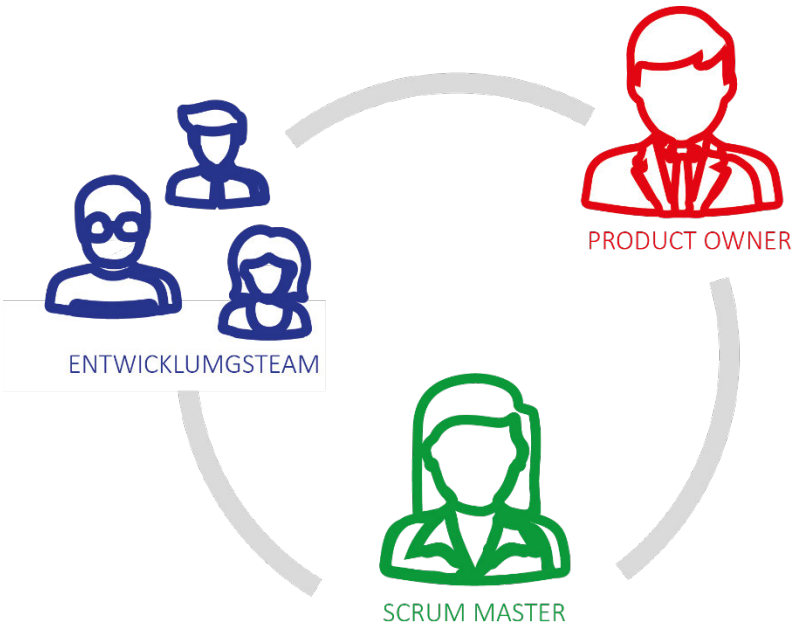
[53] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.5
[54] Vgl. Schwaber, Ken (Agiles Projektmanagement mit Scrum, 2007), S.6

charakterisiert wird, die im Scrum-Guide von Schwaber und Sutherland beschrieben sind und in den folgenden Unterkapiteln wiedergegeben werden.[55]

3.3 Scrum-Rollen

Scrum im klassischen Sinn umfasst drei festgelegte Rollen: Scrum Master, Product Owner und das Entwicklungsteam. Alle diese drei Rollen zusammen werden als Scrum-Team bezeichnet. Im folgenden Abschnitt werden diese Rollen beschrieben, die einzelnen Verantwortlichkeiten der jeweiligen Rolleninhaber, sowie die idealen Eigenschaften für jede Rolle.

Abb. 8: Scrum-Rollen



Quelle: Eigene Darstellung in Anlehnung an Scaled Agile (SAFe® 5.0, 2020), S.77

3.3.1 Product Owner

Der Product Owner (wörtlich Produkt-Besitzer) repräsentiert laut Schwaber und Sutherland die Endkundenbedürfnisse. Er steuert die Entwicklung über die Priorisierung des Product Backlogs, welches die Anforderungen an das finale Produkt beinhaltet (siehe Abschnitt 3.4.1). Seine Aufgabe besteht darin, den Wert der Arbeit der Entwicklungsteams zu maximieren.

[55] Vgl. Dechange, André (Projektmanagement, 2020), S.291

Dabei ist das Product Backlog sein wichtigstes Werkzeug. Über das Product Backlog legt er die Ausprägungen des Produktes fest und bestimmt über dessen Priorisierung Zeitpläne und Budgets. Er trifft alle Entscheidungen im Rahmen der Entwicklung eines Produktes, um es zu einem wirtschaftlichen Erfolg zu machen.[56]

In die Verantwortung des Product Owners fällt beispielsweise die Optimierung des Arbeitswertes, die das Entwicklungsteam erledigt, sowie die Sicherstellung, dass das Product Backlog sichtbar und transparent ist. Der Product Owner kann auch einige seiner Aufgaben an das Entwicklungsteam delegieren, jedoch bleibt er immer rechenschaftspflichtig.[57]

Die Rolle des Product Owners ist immer nur mit einer Person besetzt, damit das Entwicklungsteam einen definierten Ansprechpartner für alle Fragen hat.[58] Damit der Product Owner erfolgreich sein kann, muss die gesamte Organisation seine Entscheidungen respektieren.

Er ist für die Eigenschaften und den wirtschaftlichen Erfolg des Produktes verantwortlich und sollte daher über ein sehr gutes Kunden- und Produktverständnis verfügen. [59]

3.3.2 Scrum Master

Der Scrum Master ist für die erfolgreiche Abwicklung der Scrum-Vorgehensweise verantwortlich und unterstützt das Team hinsichtlich einer effizienten Arbeitsweise. Er stellt sicher, dass jeder den Scrum-Regeln und -Verfahren folgt. Darüber hinaus besitzt er große kommunikative und soziale Fähigkeiten und ist in der Lage, Konflikte in einem Team zu lösen.

---

[56] Vgl. Schwaber, Ken (Agiles Projektmanagement mit Scrum, 2007), S.7
[57] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.6
[58] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.32
[59] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.6

Der Scrum Master ist für die Implementierung von Scrum zuständig, damit sich dieses innerhalb der Unternehmenskultur einfügt und damit den erwarteten Nutzen bringt.[60]

Das Aufgabengebiet eines Scrum Masters besteht unter anderem darin, zwischen Team, Product Owner und externen Interessengruppen sowie dem internen Team zu moderieren und Hindernisse aus dem Weg zu räumen.

Außerdem ist er dafür zuständig, Verfahren zur Messung quantifizierbarer Einheiten, sogenannten Metriken, anzuwenden, um die Leistung des Teams zu analysieren und diesem zwecks Motivation und Weiterentwicklung die Ergebnisse mitzuteilen und umzusetzen.

Der Scrum Master hat für einen erfolgreichen Prozess eminente Bedeutung, da er sich administrativ um die Einhaltung der Scrum-Regeln kümmert, somit das Team entlastet und eine höhere Konzentration des Teams auf das Erreichen des Sprintziels legt.[61]

3.3.3 Entwicklungsteam

Das Entwicklungsteam ist, wie der Name bereits vermuten lässt, für die Entwicklung und Lieferung der Produktinkremente (siehe Abschnitt 3.4.3) und damit das Produkt verantwortlich. Dabei wird gänzlich auf die Unterscheidung von Rollen verzichtet. So gibt es keinen Backend-Entwickler, keinen Analysten, User Interface Experten oder Tester.[62] Der Scrum Guide definiert keinerlei Hierarchie. Alle Teammitglieder eines Scrum Entwicklungsteams sind somit ausnahmslos Entwickler. Das Team ist so strukturiert und befähigt, dass es seine eigene Arbeit selbst organisiert und managt. Die daraus resultierende Synergie optimiert die Gesamteffizienz und –Effektivität des Entwicklungsteams.[63]

---

[60] Vgl. Schwaber, Ken (Agiles Projektmanagement mit Scrum, 2007), S. 7
[61] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.33
[62] Vgl. Böhm, Janko (Erfolgsfaktor Agilität, 2019), S.36
[63] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.7

Die ideale Größe des Entwicklungsteams sind fünf Mitglieder, der Scrum Guide empfiehlt eine Größe zwischen drei und neun Mitgliedern. Das Entwicklungsteam verfügt über alle notwendigen Kompetenzen, um die Einträge des Product Backlogs ohne weitere Rücksprachen mit anderen Experten umsetzen zu können. Es schätzt, wieviel Arbeit in einen Sprint passt und erzeugt aus dem Product Backlog potenziell auslieferbare Funktionalität.[64]

3.4 Scrum-Artefakte

Neben den drei beschriebenen Rollen im Scrum-Prozess gibt es ebenfalls drei Artefakte, ähnlich einer Art Werkzeug, mit dessen Hilfe die Arbeit dokumentiert und gesteuert wird. Zu diesen Artefakten zählen das Product Backlog, das Sprint Backlog sowie das Produktinkrement. In den folgenden Unterkapiteln werden diese drei Anwendungen näher erläutert und vorgestellt.

3.4.1 Product Backlog

Ein Backlog stellt eine dynamische Liste von durchzuführenden Arbeitsaufträgen dar. Dabei versteht man unter dem Product Backlog eine Art Box zum Erfassen und Managen von Anforderungen, die an das Produkt im Scrum-Prozess gestellt werden und welche für das Erreichen des Projektzieles notwendig sind. Ausschließlich der Product Owner ist für das Product Backlog, seine Inhalte, den Zugriff darauf und die Reihenfolge der Einträge verantwortlich.[65]

Das Product Backlog wird zwar zu Beginn eines Projektes angelegt, jedoch kann und muss es während des gesamten Projektverlaufes fortlaufend verändert und angepasst werden. Neue Anforderungen kommen hinzu, bestehende verändern sich oder fallen ganz weg. Im Product Backlog werden alle Features, Funktionalitäten, Verbesserungen und Fehlerbehebungen als sogenannter Product-Backlog-Eintrag (*Product Backlog Item* oder kurz PBI) aufgelistet. Diese sollen laut Scrum Guide

[64] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.32 f.
[65] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.15

folgende Attribute enthalten: eine Beschreibung, die Reihenfolge, die Schätzung und den Geschäftswert.[66]

Der Scrum Guide in seiner Eigenschaft als Rahmenwerk äußert sich jedoch nicht dazu, wie diese *Product Backlog Items* geschätzt werden sollen. Dafür lässt Scrum den Anwendern des Rahmenwerks größtmöglichen Freiraum. Jedoch haben sich in der Praxis einige Konzepte und Methoden als eine Art Standard bewährt, um mit Scrum zu arbeiten.

Die häufigste Methode ist wohl die Schätzung der Arbeitseinheiten mithilfe einer abstrakten Zahlengröße, den sogenannten Story-Punkten, die sich an der Fibonacci-Reihe orientiert (1,2,3,5,8,13,20). Je größer die Zahl, desto höher schätzt das Team die Komplexität des Items ein. Dadurch können die einzelnen Arbeitseinheiten in Relation zueinander gestellt werden. Es erfolgt keine Erfassung des direkten zeitlichen Aufwandes. Eine andere Methode ist die Schätzung des Aufwands in Zeiteinheiten (Stunden, Tage, Wochen). Welche Methode zum Schätzen genutzt wird, wird nicht vorgeschrieben und obliegt einer individuellen Entscheidung des Scrum-Teams. Die Komplexität soll eine Voraussage der Kosten geben, die investiert werden müssen.

Neben der Schätzung der Komplexität kalkulieren viele Scrum-Teams zusätzlich den Geschäftswert. Dieser sagt aus, was im Gegenzug für die Kosten als Return erzielt werden kann. Das Ziel der Schätzung des Geschäftswertes und der Komplexität ist, dass der Product Owner besser bestimmen kann, welcher Eintrag die höchste Wertigkeit besitzt und damit eine höhere Priorität in der Abfolge der Bearbeitung erhält.[67]

Über mehrere Sprints hinweg kann aus dieser Schätzung der fertigen Items eines jeden einzelnen Sprints die sogenannte Velocity, die Teamgeschwindigkeit,

[66] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.15
[67] Vgl. Bartonitz, Martin (Agile Verwaltung, 2018), S.106-108

ermittelt werden. Die Velocity wird dazu benutzt, Release- und Sprintplanungen zu erstellen.[68]

Ein sich fortlaufend ändernder Product Backlog mit geschätzten und priorisierten Anforderungen, sowie mit wichtigen Anforderungen, die detailliert vorbereitet sind, stellt somit einen wesentlichen Teil des Grundgerüstes für einen erfolgreichen durch Scrum gemanagten Prozess dar.[69]

Abb.9: Prodcut-Backlog mit Items

Quelle: Eigene Darstellung in Anlehnung an Itemis AG (Eigenschaften des Product Backlog, 2016) [online]

Ganz oben im Product Backlog rangieren die *Product Backlog Items*, die vom Development Team in den nächsten Sprint gezogen werden. Dazu müssen diese über eine bestimmte Qualität und Größe verfügen. Damit diese Items bereit (ready) sind, um in einen Sprint gezogen zu werden, müssen sie die Kriterien der sogenannten *Definition of Ready* erfüllen. Sie müssen eine bestimmte Qualität aufweisen.[70]

Scrum schreibt nicht vor, in welcher Form die einzelnen Elemente des Product Backlogs (PBI) formuliert werden müssen.[71] Oft wird eine Produktidee hinsichtlich der Anforderungen anhand sogenannter User Stories beschrieben. Eine User Story

[68] Vgl. Maximini, Dominik (Scrum – Einführung in der Unternehmenspraxis, 2018), S.203
[69] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.15
[70] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.49
[71] Vgl. Maximini, Dominik (Scrum – Einführung in der Unternehmenspraxis, 2018), S.193

ist eine Möglichkeit, um *Product Backlog Items* zu formulieren. Dabei wird mit einem bestimmten Satzmuster die Interaktion eines menschlichen Individuums mit einem technischen System beschrieben: Als [Rolle] möchte ich [Funktion] um zu [Nutzen].[72]

Auch wenn sie in der Praxis oft Anwendung finden, sind User Stories eigentlich nicht Bestandteil von Scrum.[73] Um die Nähe zur Praxis in dieser Arbeit beizubehalten, werden User Stories jedoch auch in den weiteren Kapiteln mit in Betracht gezogen.

Um zu erkennen, wie User Stories in den Gesamtzusammenhang der Menge an Anforderungen passen, werden sie mithilfe eines sogenannten Storyboards in einen gemeinsamen Kontext gestellt. Ein Storyboard (oder auch Storymap) ist die visuelle Darstellung der geordneten Anforderungen (PBI) entlang zweier Achsen. Horizontal läuft eine zeitlich geordnete Sicht auf die Anforderungen und vertikal die Wichtigkeit einer Aufgabe oder einer Detailaufgabe für das Produkt.[74]

Wenn die User Story im Backlog weiter nach oben ‚rutscht', werden detailliertere Anforderungen dahinter festgelegt, um ein besseres inhaltliches Verständnis für die Entwickler zu sichern. Diese werden dann als sogenannte Akzeptanzkriterien in der User Story vermerkt.

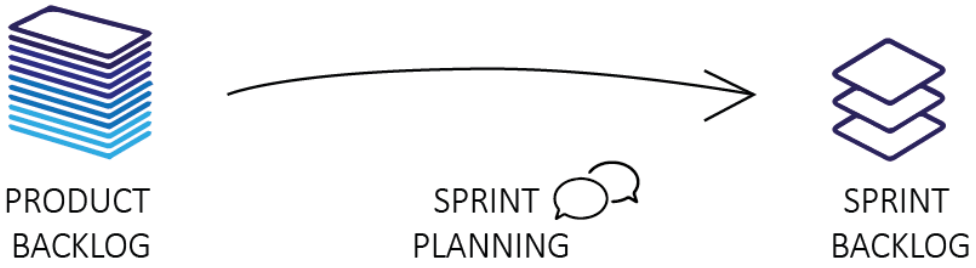
3.4.2 Sprint Backlog

Das Sprint Backlog ist eine Liste ausgewählter Einträge aus dem Product Backlog, heruntergebrochen auf den konkreten Zeitabschnitt eines Sprints. Mithilfe dieser Liste soll in einem Sprint ein potenziell lieferbares Inkrement des Produktes entstehen und das Sprintziel erreicht werden. Das Sprint Backlog ist eine Prognose des Entwicklungsteams darüber, welche Funktionalität im nächsten Inkrement

[72] Vgl. Dechange, André (Projektmanagement, 2020), S.292
[73] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.49
[74] Vgl. Vigenschow, Uwe (APM – Agiles Projektmanagement, 2015), S.50

enthalten sein wird, sowie über die erforderliche Arbeit, die benötigt wird, um diese Funktionalität zu liefern.[75]

Abb. 10: Backlog-Artefakte



Quelle: Eigene Darstellung in Anlehnung an Preußig, Jörg (2018), S.157

Auf Basis der Priorisierung und der Schätzung werden vom Team so viele *Product Backlog Items* in einen Sprint übernommen, wie es meint, in dieser Zeit erledigen zu können (siehe Sprint-Planning in Kapitel 3.5.3).[76]

Das Sprint Backlog macht die gesamte Arbeit sichtbar, die das Entwicklungsteam für notwendig erachtet, um das Sprintziel zu erreichen.[77] Als Instrument zur besseren Visualisierung des Sprint Backlogs findet das Taskboard häufig Anwendung. Wie es der Name bereits sagt, dient es zur Darstellung von Aufgaben (Tasks), welche durch die Dreiteilung des Boards in *Tasks open*, *Tasks in progress* und *Tasks done* relativ einfach strukturiert und gesteuert werden.[78]

3.4.3 Produktinkrement

Am Ende eines Sprints steht das fertige Produktinkrement, das die Product-Backlog-Einträge bzw. die Anforderungen in Form von Funktionen und weiteren Merkmalen umsetzt, zur Abnahme durch den Product Owner bereit. Es handelt sich um ein potentiell auslieferbares Zwischenprodukt. Daraus ergibt sich pro Sprint immer ein Inkrement, das wiederum Bestandteil der folgenden Inkremente ist, wie in Abbildung 11 grafisch dargestellt.[79]

---

[75] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.16
[76] Vgl. Dechange, André (Projektmanagement, 2020), S.296
[77] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.16
[78] Vgl. Dechange, André (Projektmanagement, 2020), S.296
[79] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.17

Abb.11: Inkremente bei Scrum

Quelle: Eigene Darstellung in Anlehnung an Preußig, Jörg (2018), S.157

Das Inkrement ist das Ergebnis eines Sprints und muss daher die *Definiton of Done* des Teams erfüllen. Die *Definition of Done* (DoD) ist eine bindende Vereinbarung zwischen Entwicklungsteam und Product Owner. In ihr wird festgehalten, welche Maßnahmen für jede umgesetzte Anforderung erfüllt sein müssen, damit diese von allen als fertig (done) angesehen wird. Die DoD ist das wichtigste Werkzeug des Scrum-Teams zur Sicherstellung der Produktqualität.

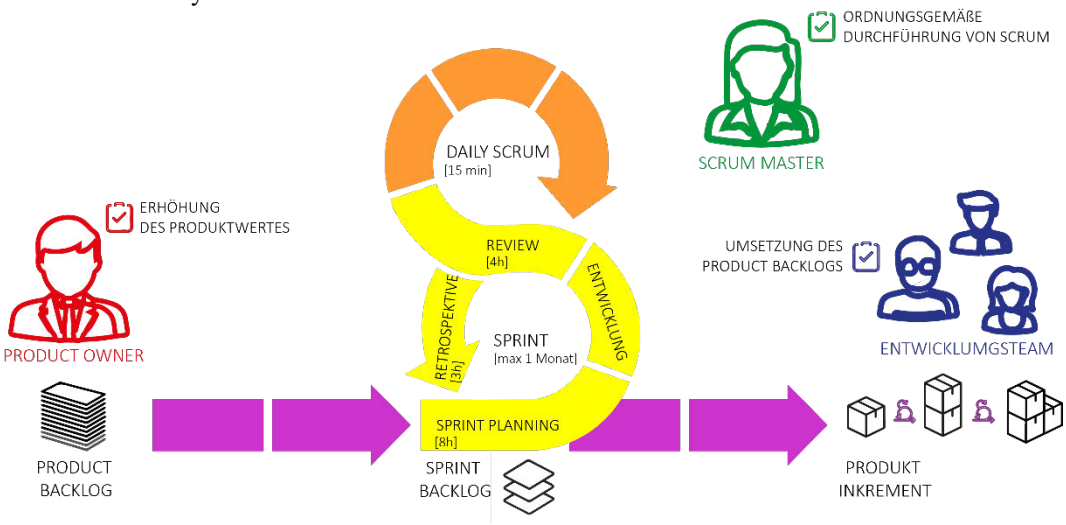
Alle vereinbarten Maßnahmen sind stets einzuhalten. Es gelten folglich nur solche umgesetzten Product-Backlog-Einträge als fertig, die alle Anforderungen der DoD erfüllen. Ist auch nur ein Element der *Definition of Done* nicht erfüllt, so kann das *Product Backlog Item* nicht abgenommen werden, wird in den nächsten Sprint übernommen und muss dort fertiggestellt werden.[80]

3.5 Scrum-Zyklus

Scrum beinhaltet mehrere Events. Neben den handelnden Personen und den Artefakten helfen diese dabei, Regelmäßigkeit herzustellen und einen strukturierten Ablauf sicherzustellen. Dieses Kapitel beschäftigt sich mit Erklärungen und Darstellungen zu den jeweiligen Ereignissen.

[80] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.17 f.

Abb.12: Scrum-Zyklus im Überblick



Quelle: Eigene Darstellung in Anlehnung an Preußig, Jörg (2018), S.157

Bei den Scrum-Events wird auf das Prinzip des Timeboxing zurückgegriffen. Dabei wird für ein bestimmtes Meeting ein Zeitraum (Timebox) definiert, der nicht verlängert werden darf. Eine Timebox stellt eine Obergrenze dar, kann also auch kürzer sein. Lediglich der Sprint wird nicht verkürzt, sondern immer erst zum vereinbarten Zeitpunkt beendet.[81]

3.5.1 Sprint

Der gesamte Arbeitszyklus ist in Teilprozesse, die sogenannten Sprints gegliedert. Ein Sprint stellt einen Arbeitsablauf dar, der iterativ, also wiederholend abläuft. Ein Sprint wird deshalb auch als Iteration bezeichnet. Der Sprint ist ein Container-Event, in dem die anderen vier Events beinhaltet sind und besitzt daher auch keinen Meeting-Charakter.[82] Ein Sprint dauert gemäß Scrum Guide maximal einen Monat, innerhalb dessen ein fertiges (done), nutzbares und potenziell auslieferbares Produktinkrement hergestellt wird. Alle Sprints innerhalb eines Entwicklungsprozesses sollten die gleiche Dauer haben. Dabei startet der neue Sprint sofort nach Abschluss des vorherigen.[83]

[81] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.35
[82] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.34
[83] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.9

Jeder Sprint hat ein Ziel, der die Umsetzung, die Arbeit und das resultierende Produktinkrement in die richtige Richtung lenkt. Während eines Sprints werden keine Änderungen vorgenommen, die das jeweils festgesetzte Sprintziel gefährden würden.[84]

Ein Sprint beinhaltet folgende Ereignisse: das Sprint Planning, die Daily Scrums, die Entwicklungsarbeit, das Sprint Review und die Sprint Retrospektive. Es ist eine Art Miniprojekt: Zu Beginn werden Anforderungen geklärt und die Arbeit im Sprint geplant, dann findet die Produktentwicklung statt. Am Ende des Sprints werden die Arbeitsergebnisse gemeinsam begutachtet, und es wird eine Sprint Retrospektive durchgeführt.[85]

Abb.13: Der Sprint

Quelle: Eigene Darstellung in Anlehnung an Preußig, Jörg (2018), S.157

Während eines Sprints kann der Anforderungsumfang neu ausgehandelt werden, wenn sich neue Erkenntnisse ergeben haben, jedoch darf dabei der Qualitätsanspruch nicht geschmälert werden.[86]

[84] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.9
[85] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.34
[86] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.9

3.5.2 Daily-Scrum

In einer täglichen Besprechung, dem sogenannten Daily-Scrum-Meeting, kommen das Team und der Scrum Master für eine kurze und festgelegte Zeit (der Scrum Guide empfiehlt 15 Minuten) zusammen und berichten kurz über den erreichten Status, über Hindernisse sowie die geplanten Aufgaben bis zum nächsten Daily Scrum. Das Daily Scrum sollte bewusst kurzgehalten werden, aber täglich stattfinden, um eine optimale Kommunikation im Team zu gewährleisten. [87]

3.5.3 Sprint-Planning

Beim Sprint Planning trifft sich das Scrum Team, um den soeben begonnenen Sprint zu planen. Der Product Owner beschreibt das Ziel, das mit dem Sprint erreicht werden soll, und präsentiert die Product Backlog Einträge, welche das Ziel erfüllen sollen. Das ganze Team erarbeitet gemeinsam ein Verständnis für die Arbeitsinhalte der nächsten Wochen und das Entwicklungsteam legt anschließend fest, wie viele Product-Backlog-Einträge in den aktuellen Sprint passen. Das Team stellt eine Prognose über die Funktionalität, die im Sprint entwickelt werden soll, auf.[88]

In diesem Meeting wird die technische Umsetzung der Items geplant, und die dafür notwendigen Einzelaufgaben werden fixiert. Das Entwicklungsteam organisiert selbst, wie es die Arbeiten im Sprint Backlog angeht.

Laut Scrum Guide hat das Sprint Planning eine maximale Dauer von acht Stunden für einen einmonatigen Sprint. Anschließend kann das Entwicklungsteam mit der Arbeit beginnen. [89]

Das Product Backlog Refinement ist kein offizielles Event gemäß des Scrum Guides. Dieser definiert das Backlog Refinement lediglich als Tätigkeit der Verfeinerung und überlässt die Umsetzung den Anwendern. Das Refinement dient

[87] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.12
[88] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.34
[89] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.10

dazu, das Product Backlog für den nächsten Sprint vorzubereiten. Dazu werden neue Product-Backlog-Einträge (PBI) vorgestellt und erklärt sowie bestehende Items bei Bedarf zugeschnitten und neu geschätzt.[90]

3.5.4 Sprint-Review

Die Abnahme des im Sprint entwickelten Inkrements erfolgt im sogenannten Sprint Review. Anhand des Inkrements begutachten Product Owner und bei Bedarf auch die Stakeholder den aktuellen Stand der Produktentwicklung und definieren auf dieser Basis die weiteren Schritte bzw. die nächsten umzusetzenden Anforderungen im Product Backlog.[91]

Beim Sprint Review handelt es sich um ein informelles Meeting, es ist kein Statusreport. Die Vorführung des Inkrements ist als Anregung für Feedback und als Basis für die Zusammenarbeit gedacht.

Das Sprint Review sollte bei einem einmonatigen Sprint nicht länger als vier Stunden dauern. [92]

3.5.5 Sprint-Retrospektive

In der Sprintretrospektive trifft sich das ganze Scrum Team, um auf die Zusammenarbeit im aktuellen Sprint zurückzublicken. Es gibt dem Team die Gelegenheit zu überprüfen, was gut lief und wo Verbesserungspotenziale bestehen. Der Scrum Guide gibt hierfür eine Obergrenze von drei Stunden vor. [93]

Zum Ende der Sprint Retrospektive sollte das Scrum-Team konkrete Maßnahmen zur Verbesserung des kommenden Sprints definiert haben und diese im nächsten Sprint eigenständig umsetzen. [94]

---

[90] Vgl. Pfeffer, Joachim (Produkt-Entwicklung, 2020), S.35
[91] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.13
[92] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.13
[93] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.14
[94] Vgl. Schwaber, Ken; Sutherland, Jeff (The Scrum Guide™, 2017), S.14

Nach der Darstellung der wesentlichen Merkmale der beiden agilen Vorgehensweisen Scrum und Kanban ergeben sich nun die Fragen: Wo liegen die Stärken und Schwächen der Modelle? Haben Kanban und Scrum Gemeinsamkeiten und durch welche Merkmale können sie klar voneinander differenziert werden?

4 Scrum und Kanban: Gemeinsamkeiten und Unterschiede

Um einen Überblick über die Stärken und Schwächen von Kanban und Scrum zu erhalten, wurden die beiden Vorgehensmodelle mithilfe der SWOT-Analyse untersucht. Die Ergebnisse wurden zur besseren Anschaulichkeit in den folgenden zwei Tabellen gesammelt. Die erste Tabelle beinhaltet die Stärken und Schwächen mit den daraus resultierenden Chancen und Risiken des Vorgehensmodells Kanban. Die zweite Tabelle veranschaulicht parallel dazu die Merkmale des Frameworks Scrum. Anhand der SWOT-Analyse können die Gemeinsamkeiten und Unterschiede von Scrum und Kanban abgelesen werden.

Der Hauptunterschied zwischen den beiden Vorgehensmodellen besteht darin, dass sich Scrum auf die *iterative Produktentwicklung* und Kanban auf die *kontinuierliche Prozessverbesserung* konzentriert.[95] Während Scrum pro Sprint auf ein in sich geschlossenes Produktinkrement hinarbeitet und das Produkt Iteration für Iteration erweitert und verbessert wird, ist Kanban bestrebt, den Prozessfluss zu optimieren sowie Schwachstellen und Engpässe des Prozesses aufzudecken.[96]

Wo hat Kanban seine Stärken und Schwächen?

Tabelle 1: SWOT-Analyse des Vorgehensmodells Kanban

STÄRKEN	**SCHWÄCHEN**
– **Visualisierung** des Workflows (1) – gute **Dokumentation** durch Kanban-Board (2) – **direkte Limitierung** der parallelen Arbeit (WiP) und Reduktion des Multitaskings (3) – **Selbstorganisation** des Teams und Pull-System der Aufgaben (4) – **Flow-Metriken,** z.B. Messung der Durchlaufzeit und damit Schaffung einer Kennzahl für die Produktivität (5)	– Teammitglieder müssen **überlappende Kompetenzen** haben (A) – **wenige feste Regeln**, an die sich jeder halten muss – stellt Anforderungen an Selbstorganisation von Teams (B) – **Komplexitätsmaximum** (sehr umfangreiche, langwierige und vielschichtige Projekte übersteigen die Möglichkeit des Kanban-Boards) (C)

[95] Vgl. Frömling, Sabine (Was Scrum von Kanban unterscheidet, 2020) [online]
[96] Vallon, Raoul (Kombination von Agil und Lean, 2014), S.29

– Bearbeiten von **Änderungen** und neuen Anforderungen jederzeit möglich (6)	
CHANCEN – mehr Zusammenarbeit (3,4) – höhere Transparenz (1,2,5) – bessere Auslastung (Effizienz) (3) – kurzfristige Problem-Identifikation (1,6) – erhöhte Arbeitsmotivation (4) – größere Flexibilität (1,4,6) – erhöhte Produktivität (3,4,5,6) – Verbesserung einzelner Prozessschritte (2) – Optimierung der Prozessgeschwindigkeit (5) – kontinuierliche Prozessverbesserung (1, 2,5,6) – faktenbasierte, bessere Kommunikation (1,2,5)	**RISIKEN** – Engpässe können entstehen (A) – weniger Zielorientierung (B) – Überforderung mit Selbstorganisation (B) – keine ausschließliche Bearbeitung komplizierter und komplexer Projekte (C)

Quelle: Eigene Darstellung

Kanban setzt als zentrales Prinzip auf die Visualisierung, um die Effektivität zu steigern. Dadurch wird die Transparenz im Team erhöht – jeder weiß zu jeder Zeit, was zu tun ist und wo wer gerade steht. Mindestens genauso wichtig sind die direkte Limitierung der parallelen Arbeit und damit die Konzentration auf die wesentlichen Aufgaben sowie die Reduktion des Multitaskings. Durch den gesteigerten Fokus können Aufgaben schneller und flexibler bearbeitet werden.[97] Neue Arbeitspakete können bei vorhandener Kapazität bei Kanban jederzeit hinzugefügt werden. Im Gegensatz dazu soll bei Scrum verhindert werden, innerhalb einer Iteration neue Arbeitspakete in den Sprint-Backlog hinzuzufügen, um den Prozess nicht zu unterbrechen und das Ergebnis unter Umständen permanent zu verschieben.

Bei Kanban steht das Kanban-Board im Zentrum. Es dient als Dokumentation der Arbeit, auf die jedes Teammitglied Zugriff hat und wichtige Informationen im Projektverlauf nachvollziehen kann. Die Protokollierung erhöht nicht nur die

[97] Vgl. Weller, Kay (Kanban und Scrum im Vergleich, o.D.) [online]

Transparenz, sondern bildet auch die Basis für eine kontinuierliche Prozessverbesserung.[98]

Die Arbeit wird bei beiden begrenzt, jedoch auf unterschiedliche Art und Weise. Während Kanban explizit pro Prozessschritt WiP-Limits definiert, findet bei Scrum eine implizite Limitierung durch die Definition des Sprint-Umfangs statt.[99]

Auch bei den Metriken lässt sich ein entsprechender Unterschied zwischen Kanban und Scrum feststellen. Scrums *Velocity* misst die durchschnittliche Anzahl an Story-Punkten, die für die Arbeitspakete pro Iteration geschätzt wurden, während Kanbans *Cycle-Time* die Durchschnittszeit von Arbeitspaketen untersucht, die es benötigt, um den Workflow komplett zu durchlaufen.[100] Es gibt weitere Flow-Metriken (Work-Item-Age, Throughput, Lead-Time), die es in Kanban zu messen und verfolgen gilt, damit der Fokus auf die Optimierung des Arbeitsflusses gerichtet wird.[101]

Eine Gemeinsamkeit findet sich bei beiden Messmethoden in der Verfolgung des Ziels, den Release-Plan zu optimieren und eine zuverlässige Vorhersage für den nächsten Releasetermin treffen zu können. Sowohl Kanban, als auch Scrum streben danach, auslieferbare Softwareinkremente möglichst schnell und oft zu releasen. [102] In Scrum heißt das, die Velocity zu erhöhen, in Kanban, die Cycle-Time zu reduzieren.

Es ist ersichtlich, dass Kanban out of the box sehr wenig vorschreibt und Scrum daher vergleichsweise restriktiv ist. Kanban muss daher in der Praxis erweitert werden. Auch Scrum bedarf situationsbedingter Anpassungen.[103]

---

[98] Vgl. Rehkopf, Max (Was ist ein Kanban Board?, o.D.) [online]
[99] Vgl. Fritz, Patrick (Kanban vs. Scrum im Vergleich, 2017) [online]
[100] Vgl. Vallon, Raoul (Kombination von Agil und Lean, 2014), S.29
[101] Vgl. Vacanti, Daniel; Yeret, Yuval (Der Kanban Guide für Scrum-Teams, 2019), S.3 f.
[102] Vgl. Frömling, Sabine (Was Scrum von Kanban unterscheidet, 2020) [online]
[103] Vgl. Vallon, Raoul (Kombination von Agil und Lean, 2014), S.29

Ein Risiko von Kanban ist, dass die Mitglieder des Teams überlappende Kompetenzen haben müssen, damit keine Engpässe entstehen. Wenn einzelne Entwicklungsstadien ausgelastet sind, verschiedene Mitarbeiter aufgrund ihrer fehlenden Kompetenz jedoch nicht einspringen können, um das Backlog auf null zu bringen, bleibt die Arbeit nicht im Fluss. Ab einer gewissen Projektgröße besteht die Gefahr, dass Kanban unübersichtlich wird, sodass eine Aufteilung in kleinere Einheiten notwendig wird. Dadurch, dass Kanban weniger Regeln aufweist, ist es einerseits ein guter Einstieg in die agile Arbeitsweise, anderseits stellt Kanban aber hohe Anforderungen an die Selbstorganisation von Teams. [104]

Was sind die Stärken und Schwächen von Scrum?

Tabelle 2: SWOT-Analyse des Vorgehensmodells Scrum

STÄRKEN	**SCHWÄCHEN**
– **Iterationen** (1) – **stringentes Regelwerk** mit fester **Rollen**zuteilung (2) – regelmäßige **Events** (Meetings) (3) – **Selbstorganisation** des Teams und Pull-System der Aufgaben (4) – **indirekte Limitierung** der Arbeit durch Sprint-Umfang (5) – **Commitment** in Form eines Sprintziels (6) – **Schätzung** der Arbeitspakete und Velocity als Metrik (7) – Geordnetes **Product Backlog** als ‚single source of truth' – es enthält alle Arbeiten, die für die Entwicklung des Produktes notwendig sind (8)	– **rigide Strukturen** (A) – **komplexes Regelwerk** (B) – **zeitintensive** Events (Meetings) (C) – **Skalierung** bei Projekten / Produkten mit mehreren Teams unklar (D) – Philosophie (Mindset) von **Inspect and Adapt** ist wichtig, wird aber oft nicht verstanden (E)
CHANCEN	**RISIKEN**
– Klare Prioritäten (8) – höhere Transparenz (3,6,8) – bessere Auslastung (Effizienz) (5,6,7)	– hohe Einstiegshürde (A, B, C) – Frustration des Teams (A, B, C)

[104] Vgl. Henkel, Paul (Schnelleinstieg: Kanban im Projektmanagement, 2018) [online]

– kurzfristige Problem-Identifikation (1,7,3) – erhöhte Arbeitsmotivation (4,6,8) – größere Flexibilität (1,3,4) – erhöhte Produktivität (2,4,6,7) – konstante Optimierung (1,7,3) – kürzere Kommunikationswege (3) – Arbeitsroutine (2) – Robustheit im Team (2,6)	– zu viele offene Fragen für das Management von Projekten mit mehreren Teams (D) – erwartete Erfolge durch Scrum bleiben aus (E)

Quelle: Eigene Darstellung

Im Vergleich zu Kanban legt Scrum einen stärkeren Fokus auf die organisatorische Struktur eines Teams. Das stringente Regelwerk und die feste Rollenzuteilung bei Scrum verleihen dem Team eine gewisse Robustheit. Durch die immer wiederkehrenden Abläufe (Sprints, Daily Scrum, Retrospektive etc.) wird dem Team erleichtert, Routine zu entwickeln. Ein daraus resultierendes Risiko ist, dass die Meetings unnötig viel Zeit in Anspruch nehmen und das komplexe Regelwerk eine hohe Einstiegshürde darstellt, vor der einige Teams zurückschrecken.[105]

Sowohl bei Kanban als auch bei Scrum stellt Commitment einen Grundwert dar. Die Mitglieder eines Kanban-Teams verpflichten sich dabei, Aufgaben zu beenden, bevor sie neue Arbeiten beginnen. In Scrum hingegen erfolgt die Verpflichtung für einen Sprint in Form eines Sprintziels und einer Prognose.[106]

Scrum weist jedoch auch viele Gemeinsamkeiten mit Kanban auf. Beide Verfahrensmodelle sind agil und verwenden das Pull-Prinzip. Pull bedeutet, dass das Team bestimmt, wann das nächste Arbeitspaket begonnen wird; eine Vorgabe von außen (push) gibt es nicht.[107] Die Projekte werden bei beiden Methoden in kleinen Teams durchgeführt, die selbstorganisierend arbeiten. Dadurch kann die Auslastung der einzelnen Kompetenzen verbessert und die Arbeitsmotivation jedes

[105] Vgl. Kiel, Christopher (Scrum vs. Kanban, 2015) [online]
[106] Vgl. Novkov, Alex (Kanban vs. Scrum, 2019) [online]
[107] Vgl. Vallon, Raoul (Kombination von Agil und Lean, 2014), S.28

Einzelnen erhöht werden. Beide Methoden setzten auf Transparenz, um den Prozess zu verbessern und verwenden einfache Hilfsmittel.[108]

Die Wahl zwischen Scrum und Kanban ist sehr kontextsensitiv. Sie hängt von vielen Faktoren ab, von denen im obigen Vergleich nur die häufigsten berücksichtigt wurden. In der täglichen Praxis lohnt es sich deshalb, beide Vorgehensmodelle im Hinblick auf die eigenen Projektstrukturen zu untersuchen und auch hybride Formen ins Auge zu fassen.

Die SWOT-Analyse zeigt Bedarf und Potenzial, durch sinnvolle Kombination der Arbeitsflussfokussierung von Kanban und der Struktur von Scrum einen beständigen hybriden Prozess zu schaffen. Ziel ist es, jeweilige Stärken zu vereinen und Schwächen zu verringern. Wenn Scrum als Basis-Modell Anwendung findet, kann es den klar geregelten Rahmen für die Struktur und die Organisation des Teams bilden. Durch die Erweiterung mit einzelnen Kanban-Praktiken, zum Beispiel die Visualisierung, kann nicht nur mehr Transparenz in den Arbeitsprozess gebracht, sondern der kontinuierliche Arbeitsfluss insgesamt verbessert werden.

Das in den bisherigen Kapiteln vermittelte theoretische Wissen bildet eine stabile Grundlage, um agiles Projektmanagement mit Scrum und Kanban adäquat im Rahmen eines Projektes untersuchen zu können. Um ein stabiles Fundament zu bauen, werden nun Informationen darüber benötigt, wie eine Synergie von Scrum mit Kanban in realen Situationen angewendet wird und welche Lehren, Hinweise oder Konsequenzen sich daraus ergeben. Auf die praktische Verwirklichung von Scrum mit Kanban wird im folgenden Teil der Bachelorarbeit eingegangen.

[108] Vgl. Neng, Alexander (Scrum und Kanban – wann nutze ich was?, 2018) [online]

III Praktischer Teil

5 Projektmanagement bei [redacted]

Nach der ausführlichen theoretischen Behandlung der Thematik wird in diesem Kapitel der Blick auf die praktische Anwendung des agilen Projektmanagements am Beispiel der Weiterentwicklung und Optimierung des Portals [redacted], einer hauseigenen Softwarekonzeption der [redacted] gerichtet. Begonnen wird mit einer kurzen allgemeinen Projektvorstellung, gefolgt von einer ausführlichen Beschreibung der Arbeitsabläufe und -weise mit Scrum. Dies dient als Grundlage für das bessere Verständnis des Inhaltes im Unterkapitel 4.3, in dem die Erweiterung von Scrum mit Kanban beschrieben wird. Der Fokus wird dabei auf die einzelnen Praktiken, mit denen [redacted] im Laufe des Projekts gearbeitet hat, gerichtet.

5.1 Projektvorstellung

[redacted] ist eines der größten spezialisierten und digitalisierten Anwaltsportale in Deutschland. Mit über 70.000 dort registrierten Rechtsanwälten bildet das Online-Portal ein breites Online-Anwaltsverzeichnis für Rechtsuchende. Für Rechtsanwälte bietet [redacted] entscheidende Funktionen zur umfangreichen Präsentation ihrer speziellen Kompetenzen und zur Vernetzung mit den Kollegen untereinander. [redacted] ist, wie bereits erwähnt, eine selbstgeschaffene Web-Lösung der [redacted]. [redacted]. [109]

In der IT-Entwicklung für [redacted] waren acht Mitarbeiter beschäftigt, die zunächst eine normale Konzernabteilung bildeten und später zu einem Scrum-Team avancierten. Die Weiterentwicklung einer Software, die ständig im Fluss und nie abgeschlossen ist, machte die Einführung von agilen Arbeitsweisen notwendig. [redacted]

[109] Vgl. [redacted] (Portfolio für Anwälte, 2020) [online]

[redacted] Es wurde neben Scrum auch *Scrum with Kanban* als eine Adaption aus Scrum und Kanban zum Einsatz ausgewählt.

Wie viele andere Softwareunternehmen kämpfte auch [redacted] mit seinem digitalen Produkt [redacted] vor der erfolgreichen Einführung von Scrum mit folgenden bekannten Problemen: lange Entwicklungszeiten für zu umfangreich dimensionierte Arbeitspakete; starre, ständig veraltete Dokumentationen und viel zu lange Releasezyklen mit den damit verbundenen langen Reaktionszeiten auf Kundenwünsche und verzögerten Fehlerbehebungen im Portal.

Die heutige Internetwirtschaft erfordert die Entwicklung und Bereitstellung neuer skalierbarer IT-Leistungen in einem enorm hohen Tempo bei gleichbleibend hochwertiger Qualität. Der Konsum medialer Inhalte verlagert sich ungebrochen ins Internet und ermöglicht so neue Vertriebswege und Geschäftsmodelle. Um mit dieser Entwicklung Schritt halten zu können, war eine Umstellung auf eine schnellere, agilere Portalentwicklung notwendig. Konnte das [redacted]'-Team mithilfe agiler Methoden eine Verbesserung der Produktqualität und kürzere Markteinführungszeiten erreichen?

5.2 Projektmanagementmethode und Arbeitsweise des [redacted]-Teams

Der wichtigste Bestandteil des Projektes sind die Mitarbeiter. Die Personalauswahl im Team sollte in erster Linie aufgrund der fachlichen Eignung erfolgen, jedoch spielen auch Kostensenkung und Flexibilität eine Rolle. Zu dem Scrum-Team von [redacted] gehörten zwei Entwickler in Münster, drei Softwareentwickler in Armenien, der Product Owner, der Scrum Master in Köln und ein Operations Manager, der für die Contentpflege des Portals zuständig war. Die in Scrum nicht zwingend vorgeschriebenen, aber üblicherweise vertretenen Positionen des Testingenieurs, sowie des Designers waren bei [redacted] nicht besetzt.

Wie leicht zu erkennen ist, waren die geographischen Distanzen zwischen den Teammitgliedern sehr groß. Durch die Einführung von Scrum sollten die veralteten, teilweise verkrusteten Organisationsstrukturen aufgebrochen werden, die

Entwickler aus ihren individuellen ‚Silos' geholt und in eine kollaborative Beziehung zueinander und mit den anderen Teammitgliedern gebracht werden. Die räumliche Trennung zwischen Produktmanagement, Entwicklung und Qualitätssicherung blieb zunächst bestehen, stellte später aber kein großes Problem mehr dar, weil sich die Kommunikation durch die regelmäßigen Daily-Scrum- sowie die Sprint-Planning-Meetings tatsächlich spürbar verbesserte.

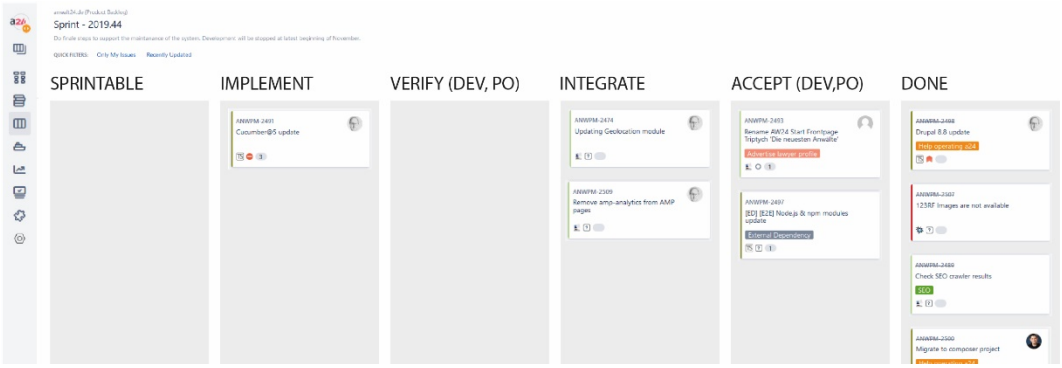
In dem [redacted]-Team wurde Scrum in einem 3-Wochen-Zyklus gelebt. Die Koordination erfolgte über verschiedene Wege, die im Folgenden aufgezeigt werden.

Für das Projektmanagement wurde die Software zur Vorgangs- und Projektverfolgung ‚Jira' verwendet. Dieses Tool fand im Haus sowohl bei den Entwicklern als auch bei dem Product Owner und dem Scrum Master die größte Akzeptanz und versprach durch eine aktive Community eine ständige Weiterentwicklung.

Die Webanwendung Jira diente als System für die Verteilung von Aufgaben und strukturierte den Workflow des Entwicklerteams, bildete den Backlog ab und dokumentierte den Status der einzelnen Aufgaben und damit den Aufgabenfortschritt. Die Entwickler entnahmen (pull) aus diesem System ihre Aufgaben, bearbeiteten sie und trugen den aktuellen Status ein. Damit war jederzeit ersichtlich, was als nächstes ansteht, wer für die jeweiligen Aufgaben verantwortlich ist und wie der Bearbeitungsstatus ist. Mit Hilfe der Software konnten klare und übersichtliche Strukturen abgebildet werden.

In Jira wurde auch das sogenannte Task-Board, oder auch Sprint-Board, gepflegt, wie in Abbildung 14 dargestellt. Dieses wurde gemeinsam von Entwicklern, Scrum Master und Product Owner genutzt und visualisierte sechs verschiedene Bearbeitungsstadien der Sprint Backlog Items: Sprintable (Sprintfähig), Implement (Implementieren), Verify (Überprüfen), Integrate (Integrieren), Accept (Akzeptieren) und Done (Erledigt).

Abb. 14: Sprint-Board von [redacted] in Jira



Quelle: Eigene Darstellung in Anlehnung an [redacted] (Screenshot Sprint-Board in Jira, 2020) [vertraulich]

In der Spalte *Sprintable* lagen die offenen Sprint Backlog Items, die bereits in den Sprint gezogen waren und am Anfang der Bearbeitungskette standen. Sobald mit der Arbeit an einem Item begonnen wurde, wurde es in die nächste Spalte *Implement* geschoben. In dem dritten Stadium *Verify* bekam der Entwickler sowohl Feedback von einem zweiten Entwickler, als auch vom Product Owner. Erst nach der Kontrolle dieser beiden Personen wurde das Item in die nächste Spalte *Integrate* geschoben. Nachdem der Code von lokal auf die Testumgebung verlagert wurde, konnte das Ticket auf den nächsten Status *Accept* gesetzt werden. Hier fand eine abschließende Überprüfung durch den Product Owner statt, und das Product Backlog Item wurde dem jeweiligen Deployment zugeordnet. Erst dann landete das Item in der Spalte *Done* des Boards, da es fertig gestellt und der Arbeitsschritt beendet worden war.

Am Anfang eines jeden Sprints verpflichtete sich das [redacted]-Team auf die Erreichung eines Sprintziels, eines übergreifenden Ziels, welches in den nächsten drei Wochen erreicht werden sollte. Aus dem Product Backlog wurden dann nur die Items in den Sprint Backlog gezogen, die dazu dienen sollten, dieses Sprintziel zu erreichen.

Beispiel: Das Scrum-Team setzte sich im Planning als Ziel für den nächsten Sprint, dass die bestehenden Anwälte von [redacted] eine Veröffentlichung zusätzlich im Monat schreiben sollen. Anschließend wurde identifiziert, welche der Product

Backlog Items für dieses Sprintziel relevant sein könnten. Schließlich hatte sich ein Item herauskristallisiert: das Ranking der jeweils besten fünf Autoren des Monats sollte auf der Frontseite der Webseite abgebildet werden. Mit Hilfe des Rankings soll der Ansporn bei den Anwälten geschaffen werden, mehr Artikel zu verfassen. Diese Art von Commitment als übergreifendes Ziel, als gemeinsame Orientierung wurde von allen Rollen als sehr wichtig empfunden und in den täglichen Arbeitsablauf integriert.

In einem täglichen fünfzehnminütigen *Daily-Meeting* wurden aktuelle Entwicklungen und Ereignisse kurz besprochen, um schnell auf Engpässe und Veränderungen reagieren zu können.

[redacted] führte noch ein zusätzliches Event ein: das sogenannte *Refinement*. Es fand wöchentlich während des laufenden Sprints statt und diente, wie bereits der Name vermuten lässt der Feinjustierung: der Product Owner stellte die vorbereiteten Items den Entwicklern vor, um alle anfallenden Fragen zu klären. Des Weiteren nutzte das Team dieses Meeting, um die User Stories, d.h. die zu erwartenden Anforderungen aus Nutzersicht, sowie den Geschäftswert zu schätzen. Das Ziel dieser Schätzung war es, eine Ordnung im Product Backlog herzustellen und so diejenigen Items zuerst zu bearbeiten, die den größten Mehrwert (Business Value) erbringen. Eine exakte Dokumentation der User Stories durch den Product Owner ist essentiell, weil bereits eine kleine Ungenauigkeit dazu führen kann, dass ein Kriterium der *Definition of Done* (DoD) nicht erfüllt wird und damit Zeit und Geld verschwendet werden. Das Element User Story wird im Scrum Guide nicht genannt, ist aber hier in der Praxis ein wertvoller Zugewinn für die erfolgreiche Durchführung des Prozesses gewesen.

Die Einheit mit der die Komplexität der User Stories geschätzt wurde, waren anfangs Story-Punkte nach der Fibonacci-Zahlenreihe. Dafür wurden verschiedene Methoden, wie beispielsweise das Planning Poker angewandt. Die relative Schätzung der Product Backlog Items sollte einerseits dazu dienen, dass sich das Entwicklungsteam über die Komplexität der Aufgaben austauscht und ein

gemeinsames Verständnis der anstehenden Arbeit geschaffen wird. Anderseits half die relative Schätzung der Planung. Wenn die durchschnittliche Anzahl an Story-Punkten pro Iteration gemessen wird, erhält man die sogenannte Velocity (Team-Geschwindigkeit). Aufgrund dieser Messung konnte nun besser geplant werden, wie viele Items im nächsten Sprint bearbeitet werden können.

Zu Beginn eines jeden Sprints fand, original wie im Scrum Guide vorgesehen, als Kick-Off-Meeting das *Sprint Planning* statt. Der Product Owner bereitete die Product Backlog Items vor und die Entwickler zogen sich diese nach dem Pull-System (siehe Abschnitt 2.4.2) zum Abarbeiten. Der Product Owner legte die Anforderungen fest, aber das Entwicklungsteam entschied, welche und wie viele Aufgaben in den jeweiligen Sprint übernommen werden.

Am Ende eines jeden Sprints fand das sogenannte *Sprint-Review-Meeting* statt. Im Review wurden die Sprint-Ergebnisse den Stakeholdern präsentiert, neue Erkenntnisse mit dem ganzen Team geteilt und Feedback ausgetauscht.

Durch das Meeting *Scrum Retrospektive* am Ende eines jeden Sprints schaffte sich das [redacted]-Team einen fixen Zeitpunkt, um den letzten Sprint zu reflektieren und gegenseitiges Feedback einzuholen.

Die Scrum-Vorgehensweise wurde von den Entwicklern bei [redacted] anfangs nicht durchgehend positiv angenommen. Die ersten Monate nach der Einführung waren von dezidierter Ablehnung der Projektmanagementmethode und einer starken Unzufriedenheit der Beschäftigten geprägt. Diese Zeit wird von den Beschäftigten in den Interviews als „super schwierig“ bewertet und mit Prädikaten „hitzige Diskussionen“ sowie „viele Konflikte“ charakterisiert.[110] Die Entwicklung bei [redacted] war durch eine lange Tradition geprägt. Es sollte ein Produkt gepflegt und inkrementell weiterentwickelt werden, welches schon über zehn Jahre auf dem Markt war. Die Entwickler waren durchgehend ausgewiesene Spezialisten

[110] Aussagen des [redacted]-Teams in den Experteninterviews siehe Anhang

mit fundierten Kenntnissen der Produkte, großer Erfahrung auf ihrem Fachgebiet und langjähriger Betriebszugehörigkeit.

Die erfolgreiche Implementierung von Scrum in das Projektmanagement von [redacted] war ein langer und aufwendiger Prozess, an dessen Ende jedoch eindeutig eine positive Entwicklung zu konstatieren war: eine Verbesserung der Kommunikation im Team, eine Konkretisierung und größere Klarheit der Produktanforderungen und eine bessere Strukturierung und Transparenz der Arbeitsschritte. Nach der anderthalbjährigen Praktizierung von Scrum konnte das [redacted]-Team als weiteres Verbesserungspotenzial immer noch zu lange Release-Zyklen und eine noch weiter optimierbare Produktqualität identifizieren.

Durch die Einführung von Scrum mit Kanban sollten diese Probleme in den Fokus rücken und intensiv verbessert werden. Der agile Coach der [redacted] war durch das professionelle Scrum-With-Kanban-Training mit den Kanban-Patriken und besonders mit den Flow-Metriken bestens vertraut und begleitete deshalb als Spezialist die Einführung von Scrum mit Kanban bei dem Scrum-Team von [redacted].

5.3 Kombination von Lean und Agil in der Praxis

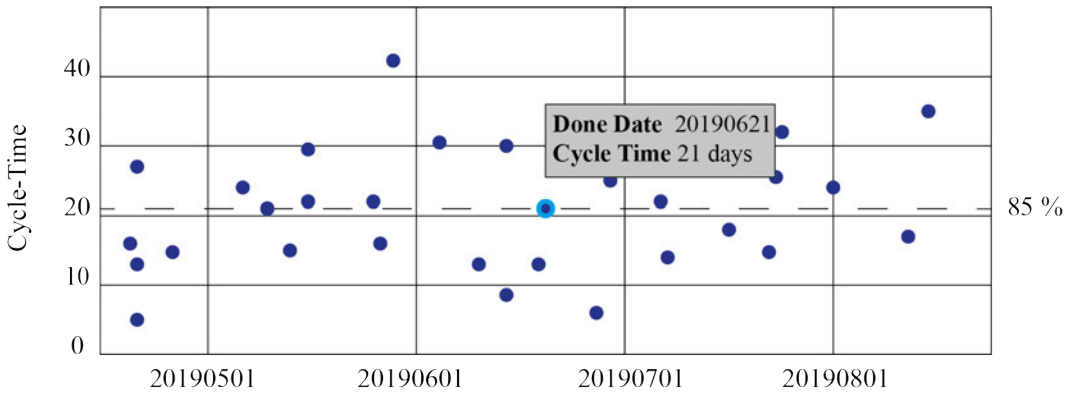
Das Scrum-Team von [redacted] war fest entschlossen, neue Wege zu gehen und wandte dafür Scrum in Kombination mit Kanban an. Eingeführt wurde diese Erweiterung während eines Allmember-Meetings, welches das Ziel hatte, die Optimierung des Workflows zu planen. Diese Aufgabe konnte erfüllt werden. Nach der Visualisierung ihres Workflows erkannte das Team, dass viele Items angefangen und nie beendet worden sind. Daher entschieden sie sich dafür, eine Taktik aus der Kanban-Praxis, die WiP-Limitierung einzuführen, also eine Eingrenzung der parallelen Arbeit, den *Work in Progress*. Diese Limitierung der laufenden Arbeiten gehört, wie in den vorangegangenen Kapiteln beschrieben, zu den grundlegenden Kanban-Praktiken.

Durch die WiP-Limitierung auf zwei Items pro Entwickler konnte das Scrum-Team im Laufe des Projektes den Fokus auf den Workflow erhöhen. Damit profitierte das gesamte Team von der Erhöhung der Effektivität und Produktivität. Normalerweise findet eine Limitierung nicht pro Person, sondern pro Spalte des Kanban-Boards statt. Da das Team sich jedoch mit der agilen Arbeitsweise, dass ein Entwickler kein Item besitzen darf, sehr schwergetan hat, war die Limitierung pro Person schon ein sehr guter Schritt in Richtung Erhöhung der Agilität.

Durch die Einführung eines durch Kanban-Elemente ergänzten Scrum sollte das Ziel erreicht werden, mit weniger Items pro Zyklus zu starten und diese schneller fertig zu stellen, um den Arbeitsfluss (Workflow) noch weiter voranzutreiben. Dazu nutzte das Team die sogenannten Flow-Metriken. Vorher trat oft das Problem auf, dass im Sprint-Planning zu viele Items in den Sprint gezogen wurden, sodass am Ende weder das Sprintziel noch die Prognose (Forecast) erreicht werden konnten. Mit Hilfe der Flow-Metriken, Durchlaufzeit und Durchsatz, wurde eine Eingrenzung der zu ziehenden Items erreicht und der totale WiP verbessert. Die Messungen machten dem Team bewusst, dass Items mehrere Tage in Folge im selben Bearbeitungsstatus geblieben und daher noch konkrete Veränderungen zur Erreichung des Ziels notwendig waren.

Das aus dem Kanban-Projektmanagement entliehene und in der Praxis bereits bewährte und aussagekräftige Cumulative Flow Diagramm fand auch bei [redacted] Anwendung. Mithilfe dieser Darstellung konnte das Team die Beziehung zwischen Work in Progress und Cycle-Time untersuchen. Durch einen horizontalen Querschnitt kann abgelesen werden, wie lange die Arbeitspakete durchschnittlich in einem Teil des Workflows verweilen. Bei [redacted] wurden anfangs durchschnittlich 85 Prozent der Items in 124 Tagen fertig. Diese Cycle-Time konnte das Team nach der Praktizierung von Scrum mit Kanban auf 21 Tage reduzieren. Das Streudiagramm in Abbildung 15 bildet die Messung des [redacted]-Teams nach ungefähr einjähriger Praktizierung von Scrum mit Kanban ab. Dabei kann für jedes Item (blauer Punkt) das Datum der Fertigstellung und die Dauer der Bearbeitung in Tagen abgelesen werden.

Abb. 15: Cycle-Time-Streudiagramm von [redacted] mit PSK

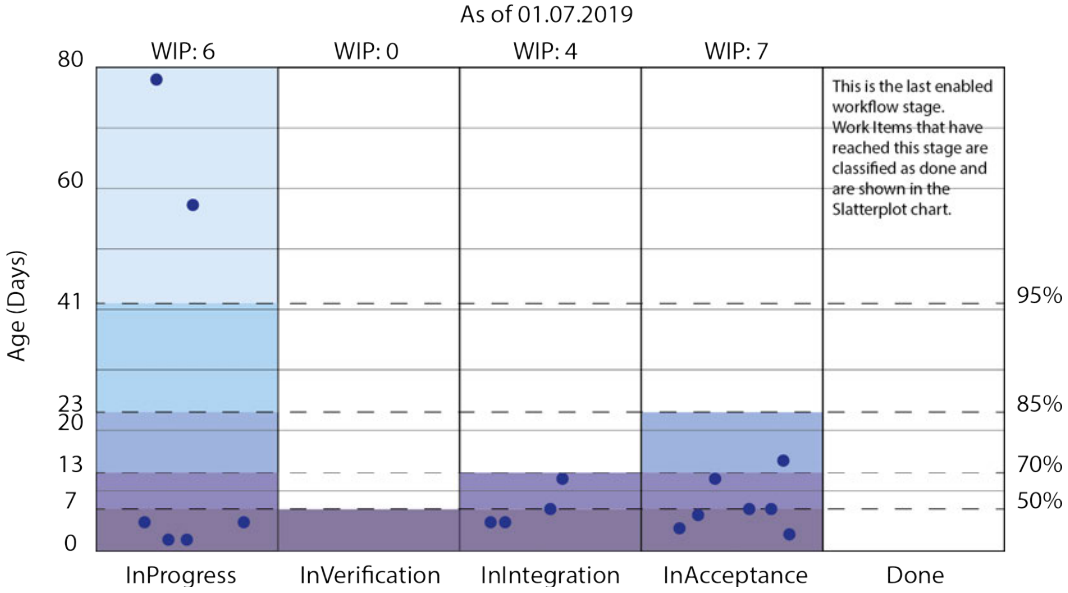


Quelle: Eigene Darstellung in Anlehnung an [redacted] (Screenshot Streudiagramm in ActionableAgile, 2020) [vertraulich]

Die Scrum-Events änderten sich durch die Einführung der Kanban-Flow-Metriken insofern, dass der Inhalt eine neue und stärkere Fokussierung auf den Workflow des Produktes bekam. Im Folgenden wird die Verschmelzung von Kanban mit Scrum anhand der beiden Scrum-Events Daily und Retrospektive näher erläutert.

Die drei, von den Teammitgliedern im Verlauf für ineffizient empfundenen Fragen im Daily Scrum (Was hast du gestern gemacht? Was machst du heute? Was siehst du für Hindernisse?) wurden bei [redacted] durch die Diskussionen über die Flow-Metriken abgelöst. Hauptsächlich beschäftigte sich das Team mit den aktuellen Veränderungen des Work-Item-Age-Chart. Auf diesem Diagramm werden alle Items, die gerade in Bearbeitung sind, in ihrem jeweiligen Status abgebildet. Es kann dabei die Work-Item-Age, d.h. die Zeitdauer vom Beginn der Bearbeitung eines Items bis zum aktuellen Tag, abgelesen werden. (Im Unterschied dazu kann die Cycle-Time nur nach Fertigstellung der Items gemessen werden.) Durch die Fokussierung auf die Items, die schon eine hohe Bearbeitungszeit (Item-Age) aufwiesen, konnte das Ziel erreicht werden, und viele Arbeitspakete sind schneller fertig geworden.

Abb. 16: Work-Item-Age-Chart von [redacted]

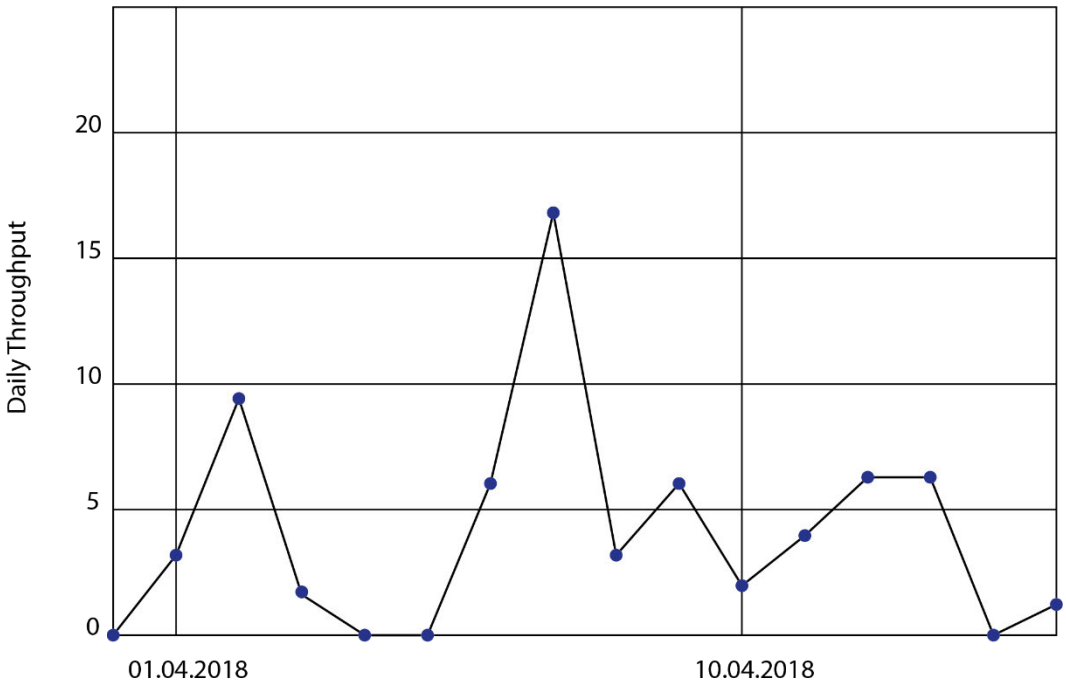


Quelle: Eigene Darstellung in Anlehnung an [redacted] (Screenshot Aging-Work-Chart in ActionableAgile, 2020) [vertraulich]

Durch die neu eingeführten Flow-Metriken wurde dem Team schnell klar, dass die Scrum-typische Metrik Velocity und das dazugehörige Schätzen in Story-Punkten überflüssig geworden waren. Wieso etwas schätzen, wenn eindeutige Messwerte vorliegen? Damit der Austausch über das gegenseitige Verständnis der Product Backlog Items nicht verloren geht, entschied sich das Team jedoch dafür, die Schätzung nicht komplett wegzulassen. So schätzte das Entwicklungsteam die Items nur noch in der Größenordnung: ein, zwei oder drei Wochen.

Die Retrospektive diente dazu, über den Prozess an sich sowie dessen Adaptierung zu reflektieren. Auch bei diesem Event waren die Flow-Metriken von großem Vorteil. Die Diskussionen waren nicht so abstrakt wie früher, sondern konnten mit Hilfe der Messwerte auf das wirkliche Problem fokussiert werden. Die Verwendung des bereits erwähnten kumulativen Flussdiagramms (CFD) und die Visualisierung des durchschnittlichen Durchsatzes wie in Abbildung 16 waren dem [redacted]-Team dabei sehr nützlich.

Abb. 17: Throughput-Run-Chart von [redacted]



Quelle: Eigene Darstellung in Anlehnung an [redacted] (Screenshot Troughtput-Run-Chart in ActionableAgile, 2020) [vertraulich]

Der Durchsatz (Throughput) bildet zusammen mit dem Work in Progress (WiP), der Durchlaufzeit (Cycle-Time) und dem Alter des Arbeitselements (Work-Item-Age) die vier grundlegenden Flussmetriken, die nach Kanban gemessen und verfolgt werden müssen, um den Arbeitsfluss zu verbessern. Der Durchsatz bildet die Anzahl der abgeschlossenen Arbeitseinheiten pro Zeiteinheit ab. Im Diagramm in Abbildung 17 entspricht jeder Punkt der Anzahl beendeter Items zu einem bestimmten Datum.

Anfangs konnte das [redacted]-Team ungefähr aller drei Sprints ein Release, d.h. ein fertiges Zwischenergebnis veröffentlichen. Nach einem Jahr verzeichneten die Messungen des Teams einen Release-Zyklus von drei Veröffentlichungen innerhalb eines dreiwöchigen Sprints.

Nachdem das Team sowohl die Dauer von der Produktentwicklung bis zur Platzierung des Produkts am Markt, als auch die Zufriedenheit des Teams

optimieren konnte, konzentrierten sie sich mehr auf das Ergebnis und die Wirkung ihrer Arbeit. Dafür rückte das Team die Erreichung der Sprintziele in den Fokus. Vor der Arbeit mit der Scrum-Kanban-Kombination erreichte das [redacted]-Team durchschnittlich jedes vierte Sprintziel, also 25 Prozent. Nach der Einführung von Scrum mit Kanban stieg die Erfüllungsrate des Sprintziels auf 69 Prozent an.

Diese positiven Auswirkungen bei [redacted] weisen darauf hin, dass eine Erweiterung von Scrum durch die Kanban-Praktiken für die Produktentwicklung sehr profitabel war. Doch sind diese Verbesserungen tatsächlich ein Ergebnis des agilen Projektmanagements gewesen? Und wenn ja, entstanden auch Risiken, die das Management nicht außer Acht lassen sollte? Wie nahmen die Projektmitglieder von [redacted] die Arbeitsweise mit Scrum und Kanban wahr? Welche Vor- und Nachteile sahen sie bei den einzelnen agilen Praktiken?

6 Potenzial der Erweiterung von Scrum mit Kanban

Nachdem die praktische Anwendung des Projektmanagement mit Scrum und Kanban ausführlich beschrieben wurde, soll das fünfte Kapitel der Bachelorarbeit Aufschluss über das Potenzial dieser Kombination geben. Für diese Beurteilung wurden Experteninterviews durchgeführt. Zuerst wird die Vorgehensweise der Durchführung der Interviews erläutert. Danach finden die Auswertung der Interviews und eine anschließende Schlussfolgerung statt.

6.1 Empirische Erhebung (Vorgehensweise)

Für die Beurteilung des Hybrid-Konzeptes *Scrum mit Kanban* wurden persönlich Experteninterviews in einem Fachverlag mit hierarchischen und internationalen Konzernstrukturen durchgeführt. Dabei handelt es sich um vier nicht-standardisierte Interviews, bei denen dem Interviewpartner die Fragen, die Reihenfolge und die Antwortmöglichkeiten freigelassen werden. Nichtstandardisierte Interviews lassen sich weiter differenzieren in Leitfadeninterviews, offene Interviews und narrative Interviews. Innerhalb dieser drei Möglichkeiten wurde das offene Interview für diese Untersuchung präferiert, da die vorher definierten Themengebiete zu einer konkreten Zielführung beitragen, ohne dass ein vorgefertigter Leitfaden die freie Entwicklung und die natürliche Gesprächsstruktur einschränkt.

Nachdem die Vorgehensweise festgelegt wurde, folgte die Planung und Durchführung der Experteninterviews. Da sich der Forschende schon intensiv mit der Thematik auseinandergesetzt hatte, konnte die Planung der Interviews im Voraus erfolgen. Zunächst wurden die Themengebiete definiert, die die Richtung des Interviews vorgeben soll. Darauf aufbauend wurde im nächsten Schritt die Auswahl der Experten getroffen.

Für die Durchführung der Interviews wurden vier Experten aus der Praxis ausgewählt. Um den Interviewpartnern ihre Anonymisierung zu gewährleisten, wurden ihre Namen durch die Rollenbezeichnung nach Scrum ersetzt.
Ziel der Interviews war es, vorhandene Stärken und Schwächen der Erweiterung von Scrum mit Kanban in der Praxis, genauer bei [redacted] zu ermitteln. Daher waren alle vier Experten Teil des [redacted]-Teams. Um die unterschiedlichen Sichtweisen und ein realistisches Abbild des Projektmanagements bei [redacted] zu erlangen, richteten sich die Fragen an den Product Owner, den Scrum Master, den Agilen Coach und einen Entwickler.

Nach der Durchführung folgte als nächster Schritt die Auswertung der Experteninterviews. Die Interviews wurden dafür aufgezeichnet, um sie anschließend schriftlich fixieren zu können. Bei der Transkription ist allerdings noch nicht erkennbar, welche relevanten Informationen sich aus der Befragung ergeben. Es existieren verschiedene Verfahren, wie die sequenzanalytische Methode, das Kodieren, die freie Interpretation oder die qualitative Inhaltsanalyse, um aus dem Datenmaterial zielführende Erkenntnisse zu filtern und diese im Rahmen des Sachverhaltes zu interpretieren. In dieser Arbeit wurden die Inhalte der Interviews qualitativ ausgewertet.

Bei der qualitativen Inhaltsanalyse wird der Text in ein Kategorie-System zerlegt und neu angeordnet. Anhand dieses Systems sollen die wesentlichen Informationen im Rahmen der Untersuchung herausgefiltert werden. Da eine Vielzahl von unstrukturiertem Datenmaterial für diese Arbeit ausgewertet werden muss, eignet sich dieses Verfahren am besten.

Die zentralen Erkenntnisse aus den vier durchgeführten Experteninterviews werden im nächsten Unterkapitel vorgestellt.

6.2 Ein hybrides Konzept (Auswertung)

Zur Erfassung des Potenzials des Scrum-mit-Kanban-Modells erscheint es aufgrund der Komplexität erforderlich, die Auswertung auf wenige, klar

abgrenzbare Teileigenschaften zu reduzieren. Da sich die jeweiligen Scrum- bzw. Kanban-Praktiken hinreichend voneinander abheben, sollen die drei folgenden Praktiken als Grundlage für die Auswertung dienen: die WiP-Limitierung, die Basis-Metriken und das Commitment in Form eines Sprintziels.

Zur Verdeutlichung werden diese drei Praktiken in ihren jeweiligen Ausprägungen nach Scrum und nach Kanban kurz gegenübergestellt. Eine ausführlichere Erläuterung der Praktiken fand bereits im Theorieteil dieser Arbeit statt.

Die Limitierung der parallelen Arbeit wird bei Scrum indirekt durch den Sprint vollzogen. Mit Hilfe des Sprint-Backlogs wird nur eine gewisse Menge an Arbeitspaketen in einen festgelegten Zeitraum zugelassen. Kanban hingegen limitiert die parallele Arbeit direkt durch eine Anzahllimitierung. Das WiP-Limit wird für jedes Stadium, das ein Arbeitspaket durchläuft, festgesetzt und in den jeweiligen Spalten des Kanban-Boards sichtbar gekennzeichnet.

Die Basis-Metrik bei Scrum ist die schon oft genannte Velocity. Dabei wird die durchschnittliche Anzahl an Story-Punkten pro Iteration gemessen, um anschließend die Team-Geschwindigkeit in einem sogenannten Burn-Down-Chart darzustellen. Kanban nutzt viele verschiedene Metriken, die zusammengefasst als Flow-Metriken bezeichnet werden. Exemplarisch soll für die Untersuchung die Durchlaufzeit (Cycle-Time) dienen. Hier wird die durchschnittliche Zeit von Arbeitspaketen gemessen, die sie benötigen, um den Workflow komplett zu durchlaufen. Die Messergebnisse werden anschließend in einem Cumulative-Flow-Diagramm visualisiert.

Das Engagement, also das Commitment als Wert ist bei beiden Vorgehensmodellen fester Bestandteil und deren Bedeutung kann als gleichwertig ausgelegt werden. Das Commitment in Form eines Sprintziels findet jedoch nur bei Scrum Anwendung. Bei Kanban ist das optional.

Der Product Owner beschreibt das Sprintziel als eine sehr „gute Erfahrung“, da es „einen Rahmen gibt, (…) eine Orientierung, eine gemeinsame Marschrichtung“. Außerdem gibt er zu bedenken: „Unser Ziel ist es nicht, die einzelnen Items umzusetzen, sondern das Sprintziel zu erreichen.“ Die Sichtweise des Scrum Masters lässt sich daran anknüpfen: „Ein Sprintziel festzulegen ist durchaus sinnvoll, wenn dies gut formuliert ist. Natürlich wird das Sprintziel in der Praxis nicht immer erreicht, ist aber dennoch ein wichtiger Fokus, eine Art Leitfaden für den Sprint.“ Der Entwickler teilt die Meinung, dass ein Sprintziel für das Management wichtig ist. „Aus technischer Sicht ist es aber nicht entscheidend. Jedes Ticket besteht bestenfalls für sich und damit ist es auch bearbeitbar.“, ergänzt er. Auch die Aussagen des agilen Coaches stimmen mit den Meinungen der anderen überein: „Das Sprintziel ist das Wichtigste, an dem wir hier arbeiten.“ Er gibt zusätzlich zu bedenken, dass es extrem wichtig ist, dieses Sprintziel auch zu erreichen, dies aber nicht gleichbedeutend damit ist, dass alle Items zwingend umgesetzt werden müssen.

Die direkte WiP-Limitierung nach Kanban sorgt laut Product Owner für einen „kontinuierlichen Flow durch den Prozess“, da Multitasking vermieden wird. Außerdem kann es helfen, die gegenseitige Unterstützung der Teammitglieder und eine Hilfestellung untereinander zu fördern. Der Experte gibt jedoch anhand eigener praktischer Erfahrung zu bedenken, dass es nicht in jedem Team gelebt werden kann und die Akzeptanz des Entwicklungsteams für eine erfolgreiche Praktizierung der WiP-Limits eine unbedingte Voraussetzung darstellt. Auch diese Ansicht teilt der Scrum Master: Er betont, dass das Einverständnis des Teams essentiell ist und die WiP-Limits an die konkreten Verhältnisse im Team adaptiert werden müssen. In der Theorie kann eine Einführung der WiP-Limits laut Scrum Master sehr hilfreich sein, um zu vermeiden, dass immer wieder Aufgaben angefangen aber nicht beendet werden. Der Entwickler hingegen ist der Auffassung, dass eine schnelle Erledigung angefangener Aufgaben im grundlegenden Interesse eines Entwicklers ist und dafür deshalb keine „neue Regel“ benötigt werde. Er sieht nicht nur keinen Vorteil, sondern vielmehr den Nachteil, dass die direkte WiP-Limitierung zu zusätzlichen Diskussionen und damit zur Verschwendung von

Zeitressourcen führen könne. Der agile Coach hält die direkte Begrenzung von paralleler Arbeit für sinnvoll, wenn die Entwickler sich drauf einlassen, nicht nur für ein Item allein zuständig zu sein. Denn dann werden die gegenseitige Unterstützung und das gemeinsame Arbeiten an einem Arbeitspaket gefördert, sodass die einzelnen Aufgaben schneller beendet werden. Wenn die Entwickler, wie es bei [redacted] der Fall war, jedoch nicht damit einverstanden sind, ist eine direkte Limitierung wirkungslos, da immer Gründe gefunden werden, um mit einer neuen Aufgabe zu beginnen.

Zu den Metriken konnte der Product Owner keine klare Entscheidung zwischen Scrum und Kanban fällen. Die Velocity ist seiner Meinung nach eine gute Fortschrittskontrolle, da das Team „über einen längeren Zeitraum eine konstante Geschwindigkeit“ benötigt. Aber er sieht auch die Vorteile der Cycle-Time als Messgröße. Der Scrum Master betont die Vorteile der Flow-Metriken im Hinblick auf die dadurch veränderte Ausrichtung bei den Scrum-Events. Der agile Coach vertritt zu diesem Thema eine klare Meinung: „Der Flow-Metriken-Kanban-Ansatz ist der modernere Ansatz, der im Grunde auch mehr mit Agilität und Scrum zusammenpasst, weil es ein empirischer Ansatz ist. Es hat nichts mit Vermutung zu tun, sondern es handelt sich um Fakten. Wenn man gemessene Zahlen hat, braucht man nicht mehr zu schätzen.“ Den größten Vorteil der Flow-Metriken sieht er darin, dass das Team viel zielgerichteter Diskussionen führen kann. Laut dem agilen Coach von [redacted] ist die Messung und Auswertung der Flow-Metriken das A und O von Scrum mit Kanban, es ist eine neue Fähigkeit, die ein Mitglied im Scrum-Team aktiv nutzen sollte, um bei erfolgreicher Nutzung öfter Releases durchzuführen und deren Termine genauer vorhersagen zu können. Nach anfänglicher Skepsis den Flow-Metriken gegenüber, sah das Entwicklungsteam die Vorteile ein und arbeitete gut mit.

Neben den Aussagen der Experten zu den drei Praktiken werden im Folgenden noch aussagekräftige Statements zusammengefasst, die für den thematischen Kontext dieser Bachelorarbeit sinnvoll erscheinen.

Für den Entwickler ist das Kanban-Board das zentrale Arbeitsmittel. Das Board stellt seiner Meinung nach die perfekte Dokumentation dar, da alle Teammitglieder zu jeder Zeit ablesen können, was im Projekt gerade passiert. Der Entwickler gibt für zukünftige Projekte zu bedenken, dass Scrum nicht die Kommunikation ersetzt. Es wird durch Scrum eine gewisse Form vorgegeben, wie alles stattzufinden hat, aber das Team muss weiterhin miteinander reden.

Eine prägnante Aussage des Entwicklers im Interview war folgende: „Man versucht, das, was in den Lehrbüchern steht über ein Projekt zu kippen und man sieht gar nicht die Projektunterschiede. Teamgröße ist entscheidend. Habe ich ein Produkt mit schnellen oder langsamen Iterationen? Habe ich zentrale Systeme oder bewege ich mich auf einer Web-Oberfläche? Aus Entwicklersicht ist es schön, wenn man sich vorher mehr Gedanken über die vorhandenen Gegebenheiten macht, bevor man eine neue agile Arbeitsweise ausrollt."

Der agile Coach fasst das Potenzial der Kombination von Scrum mit Kanban mit den folgenden drei Sätzen passend zusammen: „Scrum setzt den Rahmen, in dem die Arbeit abläuft und Kanban fokussiert den Arbeitsfluss. Deswegen wird die Kombination von Scrum mit Kanban verwendet. Du optimierst den Arbeitsfluss und hast den Rahmen vorgegeben."

6.3 Mögliche Erfolgsfaktoren (Schlussfolgerung)

Im Zentrum von Kanban steht das Flusskonzept. Mit Fluss, im Sinne von Arbeitsfluss, ist hier die Bewegung eines Geschäftswertes durch ein Entwicklungssystem gemeint. Scrum gründet auf der Theorie empirischer Prozess-Steuerung. Der Schlüssel liegt dabei in der Frequenz der Zyklen, in denen Transparenz hergestellt wird und Überprüfung und Anpassung stattfinden. Werden Kanban-Praktiken in Scrum angewendet, liegt der Fokus darauf, den Fluss durch die Rückkopplungsschleifen zu verbessern, also sowohl die Frequenz als auch die Transparenz von Überprüfung und Anpassung zu optimieren.

Scrum-Teams, die Kanban anwenden, können eine solche Optimierung des Arbeitsflusses mithilfe der folgenden vier Praktiken erreichen: Visualisierung des Arbeitsablaufs, Work-in-Progress-Begrenzung, aktives Management von laufenden Arbeiten mithilfe der Flow-Metriken und das Überprüfen und Anpassen des vom Team definierten Arbeitsablaufs.

Über die Visualisierung mit dem Kanban-Board macht das Scrum-Team seinen Arbeitsablauf transparent. Das Board kann zur Anregung der richtigen Gespräche zum richtigen Zeitpunkt sowie zum Vorschlag von proaktiven Verbesserungsmöglichkeiten verhelfen.

Der Haupteffekt der WiP-Begrenzung besteht darin, dass die Tätigkeit an einer Arbeitseinheit nur begonnen wird, wenn klar ist, dass die entsprechenden Kapazitäten verfügbar sind. Die Begrenzung von WiP kann demnach den Arbeitsfluss fördern und die Selbstorganisation des Scrum-Teams verbessern.

WiP zu begrenzen ist notwendig, jedoch nicht ausreichend. Die dritte Praktik besteht im aktiven Management laufender Arbeitseinheiten. Beispielsweise kann das Team mit Hilfe der Metrik *Work-Item-Age* sicherstellen, dass Arbeitseinheiten nicht unnötig altern. Mit der Metrik *Cycle-Time* kann kontrolliert werden, dass nur Arbeitseinheiten mit ungefähr der gleichen Geschwindigkeit in den Arbeitsablauf gezogen werden, mit der sie den Arbeitsablauf auch verlassen. Das schnelle Reagieren auf blockierte oder wartende Arbeitseinheiten kann durch die Metrik *Throughput* unterstützt werden.

Um den Arbeitsablauf zu überprüfen und anzupassen verwendet das Team die bestehenden Scrum-Ereignisse. Um den Wert, den das Scrum-Team ausliefert, zu optimieren, ist das Commitment des Teams auf ein Sprintziel essenziell. Optimistische, hoch motivierte, dem Produkt verpflichtende Projektmitglieder sind ein absolutes Muss.

Wichtig ist, dass die ergänzenden Kanban-Praktiken die Rollen, Artefakte und Events in Scrum nicht ersetzen. Teams, die Scrum mit Kanban verwenden, sollten die Scrum-Ereignisse im Sprint als Rhythmus oder als eine Art regelmäßigen Pulsschlag für die Überprüfung und Anpassung des Produkts und des Prozesses nutzen. Die Kanban-Metriken können bei den Scrum-Events helfen, den Fluss zu verbessern. Beispielsweise können im Sprint-Planning historische Durchsatz-Werte genutzt werden, um zu verstehen, was die Kapazität des Scrum-Teams für den nächsten Sprint ist.

Scrum ist ein Rahmenwerk zur Realisierung komplexer adaptiver Aufgabenstellungen. Die Teammitglieder werden in die Lage versetzt, produktiv und kreativ Produkte mit höchstmöglichem Wert auszuliefern. Die verschiedenen Kanban-Praktiken zur Optimierung des Arbeitsflusses bieten Scrum-Teams zusätzliche Möglichkeiten, das Richtige zum korrekten Zeitpunkt zu überprüfen und somit nach Bedarf anzupassen. Die extreme Fokussierung von Kanban auf Transparenz, Visualisierung und Fluss kann diese Auslieferung von Wert maximieren.

IV Schlussteil

7 Fazit

Die vorliegende Bachelorarbeit hatte das Ziel, die Verbindung von Scrum mit Kanban anhand des praktischen [redacted]-Projekts auf Grundlage der Theorie und der empirischen Erhebung zu untersuchen. Die Ergebnisse werden im letzten Kapitel dieser Arbeit zusammengefasst, bevor diese mit einem Ausblick beendet wird.

7.1 Zusammenfassung

Agile Softwareentwicklung mit dem bekanntesten Vertreter Scrum ist mittlerweile aus der Projektlandschaft nicht mehr wegzudenken. Das vergleichsweise unbekannte oder teilweise in Vergessenheit geratene, weil ältere Kanban verspricht eine höhere Flexibilität, lässt jedoch einige Detailfragen offen. In Kanban wird der Fokus auf den Arbeitsfluss gerichtet. Diese Eigenschaft kann das Scrum-Rahmenwerk und dessen Einführung verbessern und ergänzen. Eine Kombination der beiden Vorgehensmodelle erscheint daher als sinnvoller Schritt für die Erreichung der gestellten Ziele.

Eine Addition aller Elemente beider Arbeitsweisen ist allerdings nicht praktikabel. Da die Vorteile des Scrum gegenüber denen des Kanban überwiegen, sollten einige Bausteine des Kanban im Scrumkosmos verwendet werden, um damit die statischen Strukturen zu beleben.

Die in dieser Arbeit erörterten Stärken und Schwächen der beiden Projektmanagementmethoden, offenbarten sich in Form der SWOT-Analyse, die im Theorieteil aufgeführt worden ist.

Das diskutierte Hybrid-Konzept verbindet Elemente von Scrum und Kanban, um Stärken zu vereinen und Schwächen zu reduzieren. Es wurde festgestellt, dass Kanban und Scrum nicht nur theoretisch sinnvoll kombiniert werden können, sondern die Integration der Stärken von Kanban (Visualisierung und Limitierung

der WiP) die Schwächen von Scrum (starr festgelegte Strukturen) auch in der Praxis ausgleichen kann. Dazu dienten die Aussagen und Meinungen praxiserfahrener Scrum- und Kanban-Experten, die bei den im Rahmen dieser Bachelorarbeit durchgeführten Interviews getroffen wurden.

Folglich erscheint die Kombination gewinnbringend, da mit Scrum zunächst die reine Entwicklung betrachtet wird, und Kanban zusätzliche Prozessbausteine der Wertschöpfungskette verbessern möchte. Das Einnehmen einer flussorientierten Sichtweise und die Verwendung von Flussmetriken in Scrum-Ereignissen stärken den empirischen Ansatz von Scrum.

Aber die Ergebnisse der Experteninterviews haben auch gezeigt, dass es nicht nur positive Auswirkungen und Einschätzungen gab. Nicht alle Teammitglieder waren mit der agilen Arbeitsweise mit Scrum und Kanban gleichermaßen zufrieden. Eine entscheidende Voraussetzung für eine erfolgreiche Einführung von Scrum und Kanban ist die Offenheit für Veränderung und Anpassungsbereitschaft der Projektmitglieder. Daher sollten die Teammitglieder aktiv in den Entscheidungsprozess eingebunden werden, um deren Mitbestimmung und Freiräume gezielt zu fördern. Leider konnte das bei [redacted] nicht immer erreicht werden. Für zukünftige Projekte ist es daher essentiell zu analysieren, wie die bestehenden Gewohnheiten die neue Kultur blockieren könnten, bevor der Änderungsprozess gestartet wird.

Die Praxistauglichkeit und Übertragbarkeit des dargestellten Konzepts muss sich noch bei weiteren Untersuchungen und Durchführungen mit anderen Fallbeispielen und unter verschiedenen Rahmenbedingungen beweisen. Es erscheint wahrscheinlich, dass sich die Instrumente der Arbeitsweisen weiter verändern werden und sich neue Kombinationen als sinnvoll herauskristallisieren. In jedem Fall sollte eine Adaption an die konkreten Anforderungen eines Projektes und an die Eigenschaften des Teams erfolgen.

7.2 Ausblick

Agile Ideen, wie zum Beispiel die feedbackgetriebene, iterative Herstellung von Produktvarianten, setzen sich neben der Softwareentwicklung zunehmend auch in anderen Branchen durch. Agilität stellt für die Priorisierung der Arbeiten den Kundennutzen in den Vordergrund. Von den Unternehmen wird dabei noch immer sehr stark die monetäre Komponente gesehen. Ob sich diese kaufmännische Fokussierung vermehrt zu einem gesamtgesellschaftlichen Nutzen hin verschieben wird, wird die Zukunft zeigen.

Das Verständnis von einem ‚fertigen' Produkt wird sich weiter wandeln. Microsoft, Adobe und andere Softwarehersteller sind schon lange vom Verkauf eines fertigen Softwarerelease abgerückt. Nutzungsbasierte Modelle, Miete und ‚Software as a Service' rücken in den Vordergrund. Der Nutzer schreibt dabei das Produkt fort. Der Besitz dieser Produkte ist für den Anwender bereits heute weniger relevant als die Nutzungsmöglichkeit. Jeder möchte lieber die aktuelle, neuste Produktversion nutzen, als ein fertiges Produkt zu kaufen, das sich nicht mehr weiterentwickelt.

Aus diesem Grund werden sich dynamische, kleinere Organisationseinheiten, die direkt am Kunden Wertschöpfung erzeugen und direktes Feedback erhalten, gegenüber starren, großen und unbeweglichen Unternehmen behaupten. Das erfordert natürlich ein grundlegendes Umdenken in den Unternehmen. Dass gerade dieses Umdenken in der in dieser Bachelorarbeit beschriebenen Arbeit nicht einfach war, wurde hinreichend aufgezeigt. Aber nur so bleibt man wettbewerbsfähig.

Agilität ermöglicht den Unternehmen, durch effektivere Konzentration auf die wertschöpfenden Anteile eines neuen Produkts schneller an den Markt zu gehen und frühzeitig mit der Realisierung der Einnahmen zu beginnen.

Die Arbeitswelt befindet sich in stetem Wandel. Digitalisierung und Automatisierung haben diesen Prozess enorm beschleunigt. Dadurch stehen Unternehmen und ihre Mitarbeiter unter großem Anpassungsdruck.

Sobald sie die Notwendigkeit sehen, Veränderung anzustoßen und Verbesserungsprozesse einzuleiten, können agile Arbeitsmethoden wie Kanban, Scrum, Design Thinking oder auch die Kombination von deren Einzelelementen die Unternehmen sehr gut darin unterstützen, schneller Entscheidungen zu treffen und sich den Bedürfnissen des Marktes anzupassen.

Danksagung

Im Rahmen meiner Bachelorarbeit möchte ich mich bei allen bedanken, die mir in den letzten drei Jahren des Studiums zur Seite gestanden und mich in jeder Hinsicht unterstützt haben.

Ein besonderer Dank geht an Herrn [redacted], der meine Arbeit als akademischer Gutachter betreut hat und immer hilfreiche Anregungen und konstruktive Kritik für mich parat hatte.

Herzlichen Dank vor allem an Herrn [redacted], der als betrieblicher Betreuer viel Mühe und Zeit investiert hat, um mich während der Bearbeitung meiner Bachelorarbeit zu unterstützen und fachlich zu beraten.

Darüber hinaus möchte ich mich bei allen Kolleginnen und Kollegen von [redacted] bedanken, die mir in den drei Jahren des Studiums viel Wissen vermittelt und stets versucht haben, mir auf jede Frage eine Antwort zu geben. Ich wurde in allen Abteilungen immer herzlich willkommen geheißen und rasch integriert. Danke an die Experten meiner Interviews - ihre Beiträge und Antworten auf meine Fragen trugen maßgeblich zum Gelingen des Praxisteils bei.

Ein Dankschön sage ich all meinen Freunden, die mich in stressigen Phasen immer auf andere Gedanken bringen und mir immer eine große Hilfe sind. Ohne euch wären die letzten drei Jahre nicht so besonders gewesen.

Mein größter Dank gilt meiner Familie, die während des Studiums eine große Stütze für mich waren. Ihr habt mir immer mit Rat und Tat zur Seite gestanden und mich positiv bestärkt.

Anhang

A1: Transkript des Experteninterviews mit dem Product Owner von [redacted]

Interview geführt am 29. April 2020 von 11:00 bis 12:00 Uhr

Interviewer: Beim Commitment, ich hatte gestern auch nochmal nachgelesen und es war von dir auf jeden Fall ein richtig guter Hinweis, dass noch mal genauer zu definieren, was man eigentlich genau damit meint, weil Commitment als Wert an sich eben auch bei Kanban fester Bestandteil ist genauso wie bei Scrum. Also es wird nicht ganz genau gleich als Wert definiert, aber kann man auf jeden Fall so auslegen. Aber bei meiner Frage geht's um die Praktik, dass man eben im Scrum das Sprintziel als Commitment festlegt und da eben diese Prognose getroffen wird und da eben wie du auch gestern meintest es nicht darum geht, dass man das wirklich einhält, aber dass man das hat und probiert alles Mögliche dafür zu tun das man das schafft. Was Kanban so nicht hat und da ist meine Frage quasi jetzt an dich: Inwiefern du das jetzt als sinnvoll siehst so ein Commitment in Form eines Sprintziels einzugehen, auch wenn es dann vielleicht nicht unbedingt eingehalten wird und ob es dann nicht quasi der gleiche Effekt ist, wenn ich kein Sprintziel festlege?

Product Owner: Also ein Sprintziel gibt mir für den Sprint … ist ja so ein übergreifendes Ziel für die nächsten zwei Wochen, was wir erreichen wollen und ist unabhängig erstmal von den Backlog-Items, die wir da machen. Im Planning haben wir dann das Sprintziel schon oder erarbeiten es. Wir hatten es im Laufe des aktuellen Sprints haben wir immer das Sprintziel für den nächsten Sprint erarbeitet, im Refinement auch. Und haben dann z.B. gesagt wir würden gern, dass Anwälte mehr Artikel schreiben. Wir würden gerne, dass wir z.B. 10 neue Autoren gewinnen und die Autoren, die wir haben sollen im Monat jeweils ein Artikel mehr schreiben, sowas. Und dann kann ich mir im Planning überlegen, welcher meiner Backlog-Items, die ich jetzt zur Verfügung hab, zahlen denn irgendwie auf das Sprintziel ein. Also das kann zum Beispiel sein, wenn irgendein neuer Autor sich registriert, dass man ihn besonders hervorhebt, dass er hier Artikel schreiben kann und wie das funktioniert, das man das den Leuten erklärt – so ein Onboarding macht. Es kann sein, dass man denen sagt, okay den ersten Artikel … wenn du den Artikel schreibst, bekommst du einen zweiten von uns geschenkt. Dann hast du schon zwei online, das ist gut für dich. So ein Anreizsystem schafft. Und für die bestehenden Autoren kann man sagen, okay wir etablieren hier so ein Ranking Top 5 Autoren dieses Monats und das publizieren wir auf der Frontseite. So das die auch einen Anreiz haben mehr zu schreiben. Genau, dann kann man gucken, was hat man schon vorbereitet, welche Items zahlen auf dieses Ziel ein und die macht man dann. Und so hat man halt ein Commitment in der Form, dass man sich einen Rahmen gibt, okay das ist unser übergreifendes Ziel. Das ist also…habe ich bei anwalt24 zum ersten Mal so gemacht, vorher haben wir immer nur gesagt welche Backlog-Items wollen wir da so machen und das fand ich jetzt eine gute Erfahrung, will ich weiterhin beibehalten.

Interviewer: Okay, weil du quasi dann auch gesagt hast, dass das eben so eine Orientierung dann auch irgendwie schafft.

Product Owner: Genau, eine Orientierung, eine gemeinsame Marschrichtung. Ja.

Interviewer: Okay und man sagt ja auch oft, dass man die Ziele von vornerein bisschen höherstecken soll, damit man dann eben auch die Leistung noch mehr fordert. Würdest du das hier beim Scrum auch sagen, dass es schon sinnvoll ist, das immer so ein bisschen höher zu stecken?

Product Owner: Also das … da habe ich letztens was zu im Buch gelesen. Also das kommt so ganz auf den Führungsstil drauf an, den man hat und wenn man selbe hohe Ziele steckt, muss man auch selber fähig sein die zu liefern. Also wenn ich irgendwie ein hohes Ziel stecke, dann muss ich als Product Owner auch selbst … mein nötiges dazu tun um das zu erreichen oder selber schaffen so das mein Team idealerweise motiviert ist und dann auch sagt, okay hey da ist einer der schafft viel, das inspiriert uns, das wollen wir auch schaffen. Das funktioniert aber soweit ich das erfahren habe nur bei Teams die schon gut eingespielt sind und an sich schon eine gute Performance haben, die lassen sich von so Wettbewerbssituationen oder sowas so Challenges halt motivieren. Andere, die vielleicht noch nicht so gut eingespielt sind, die bisschen jünger sind, sind womöglich von sowas auch schnell überfordert oder frustriert.

Interviewer: Okay, also das würdest du dann quasi Teamabhängig entscheiden.

Product Owner: Teamabhängig und situationsabhängig. Auch haben wir vielleicht ein festes Ziel, was wir erreichen wollen, ein Releasedatum oder sowas. Ja, muss man immer individuell gucken.

Interviewer: Okay alles klar. Dann können wir zur zweiten Praktik gehen. Das hatten wir gestern ja auch schon besprochen, das eben bei Scrum das an sich vermieden soll während einer Iteration neue Arbeitspakete und Anforderungen hinzuzufügen. Aber da meintest du j auch, dass das Team von innen raus schon Änderungen natürlich anpassen kann, indem es natürlich die Items bisschen detaillierter ausarbeitet. Genau, und Kanban aber eben, dort

jederzeit Pakete hinzugefügt werden können. Und meine Frage wäre dann, welche Vor- und Nachteile du bei diesen beiden Vorgehensweisen siehst und ob man dann nicht eben teilweise diese natürliche Priorisierung, die vielleicht so gegeben ist, missachtet, wenn man eben wie in Scrum sagt, wir müssen jetzt erstmal zwei Wochen warten, bevor wir damit starten können.

Product Owner: Ja…also wie gesagt, ich kenn ja Kanban nicht so wirklich aber soweit ich das verstanden hab, haben die zwar ein Board wo ein Item durchfließt und haben auch ein Backlog aber sie haben keine Trennung zwischen einem Product Backlog und einem Sprint Backlog, weil sie keine Sprints haben, ne?

Interviewer: Ja, genau.

Product Owner: Ja, bei Scrum haben wir ne ich sag mal wir haben im Product Backlog 100 Items und davon nehmen wir uns 10 raus, die wir jetzt in diesem Sprint Vorhaben zu machen. Und darauf einigen wir uns erstmal für die nächsten zwei Wochen die zu tun, außer wir haben irgendwo während des Sprints haben wir neue Learnings – irgendwas erfahren wir – wo wir sagen okay teilweise macht‘s keinen Sinn mehr dieses Items umzusetzen um das Sprintziel zu erreichen. Weil unser Ziel ist es ja nicht diese Items umzusetzen, sondern das Sprintziel zu erreichen. So, also können wir auch während des Sprints Änderungen an den Items vornehmen, wir können welche reinnehmen, welche rausnehmen oder innerhalb des Sprints Items abändern solange diese Änderungen darauf abzielen, dieses Sprintziel zu erreichen. Und so lang diese Änderungen halt mit dem Developement-Team abgestimmt sind. Also die müssen sagen, ja, es ist okay, dass wir das tun. Und in der Praxis, also es gab…vielleicht machen wir das…haben wir gestern schon besprochen…irgend so ein Bug, irgendwas super dringendes, wo auf der Produktion was echt schief läuft und unsere Endkunden sind davon betroffen negativ, jetzt gerade, dann ergibt das halt schon Sinn, das man aus der Praxis heraus sowas noch mit in den Sprint reinnimmt, umsetzt und ausliefert, weil wir entwickeln die Produkte für unsere Kunden und das ist ja auch nicht so sinnhaft irgendwelchen wirklichen Kunden zu sagen, nein, ihr müsst jetzt zwei Wochen leiden, weil unser Sprint dauert jetzt noch so lange. So. Und da haben wir es in der Praxis oft so gemacht, wenn wir irgendein Item reingenommen haben in den Sprint, dann haben wir dafür was Anderes rausgenommen. Wir haben vorher ja schon die Kapazität geschätzt, wie viel können wir denn die nächsten zwei Wochen machen. Und wenn ich da immer nur wieder was reintue dann muss ja logisch irgendwo anders was raus, weil ich habe ja nicht plötzlich schwuppdiwupp mehr Leute oder mehr Zeit oder so.

Interviewer: Und das habt ihr dann immer im Team abgesprochen und dann immer zusammen entschieden?

Product Owner: Ja genau. Und genau deswegen kannst du halt bei Kanban immer neu was hinzufügen, weil die haben halt…in unseren Product Backlog kann ich ja auch immer neu was hinzutun. Das hat ja keine Auswirkungen erstmal auf den Sprint und weil Kanban kein Sprint hat, kann ich da einfach irgendwie…die haben da auch 100 und da tue ich, wenn ich irgendwas Wichtiges hab das ganz oben hin und dann ziehen sich das die Leute. Weil die bedienen sich immer aus dem Backlog. Uns bei uns muss man dann erstmal immer abstimmen, okay in den Backlog pack ich es, aber kommt es jetzt auch in den Sprint. Da müssen wir dann drüber reden. Wir haben quasi eine Mischung gemacht dann in Absprache was reingenommen wird.

Interviewer: Okay alles klar. Bei der WiP-Limitierung hatte ich nochmal ergänzt, dass bei Scrum ja indirekt durch die Sprints stattfindet und bei Kanban direkt durch die Anzahllimitierung in den verschiedenen Stadien, die eben so ein Arbeitspaket durchläuft, also quasi bei den Spalten vom Kanban-Board. Genau, und da hatten wir gestern ja auch schon drüber geredet, dass ihr bei anwalt24 das Limit pro Person festgelegt habt, was ja so bei Kanban nicht angedacht ist. Und da jetzt die Frage an dich, würdest du jetzt sagen, dass du, wenn ihr noch weiterentwickeln würdet bei anwalt24 dass du das mal ausprobieren würdest noch mehr in Richtung Limitierung pro Spalte zu gehen und gar nicht so sehr pro Person.

Product Owner: Ja, also pro Person hatte halt immer den…der Sinn von so einer WiP-Limitierung ist ja, Multitasking zu vermeiden. Weil man ja schon ganz lange weiß, wenn ich an ganz vielen verschiedenen Sachen arbeite dann bin ich in Summe nicht schneller, weil ich habe dann immer einen Kontextwechsel. Ich arbeite hier schnell etwas und dann denk ich mir, ich muss hier dran arbeiten. Dann muss ich mich erst einfinden, was war jetzt hier mein Sachstand. Dann tue ich das. Dann geh ich wieder dahin. (…) Es gibt irgendwann dann null Fortschritt mehr oder viel, viel weniger. So ein Grundprinzip ist ja, Start Finishing, Stop Starting…also Getting things done. Das Was du anfängst, bring das so schnell wie möglich zu Ende, sodass du das nächste machen kannst. Sodass wir einen kontinuierlichen Flow durch unseren Prozess haben wollen. Das ist erstmal so mal von meinem Verständnis der Hintergrund warum wir überhaupt die Arbeit limitieren. Jetzt hatten wir es bei uns im Team, dass wir das pro Person limitiert haben. Wir

haben gesagt, okay ein Entwickler darf nur an zwei Dingen arbeiten, also faktisch am Jira-Board an zwei Items designt sein und wenn bei Items in der Ablage lagen, sodass ich mir die angucken muss, bevor wir die auf Done schieben, dann kam es zu so einer Wartezeit bei den Entwicklern. Dann haben die natürlich immer gefragt, was mach ich jetzt mit dieser Zeit. Eigentlich war die Idee, also von mir und dem Scrum Master, also ich habe das unterstützt, da hätten die sich fortbilden können, die hätten mal gucken können, ob sie was am Code refecktern oder neue Ideen entwickeln oder irgendwie eine Schulung machen oder sowas. Aber das ist bei denen nicht so gut angekommen, die wollten einfach implementieren, programmieren. Vor allem auch die externen, die wir extern eingekauft haben, da hatten wir auch immer schon den Verdacht, dass da vielleicht der Manager drauf guckt, okay wie viel Zeit stecken in die Implementierung, wie viel Zeit stecken die in Meetings oder sowas. Wir haben da immer drauf hingewiesen, ihr könnt das ruhig machen…naja. Auf jeden Fall ist das nicht so gut angekommen, das wurde nicht so gut gelebt. Eigentlich war ja dann auch so der Hintergrund, wenn man sowas wirklich pro Spalte macht und dann knubbeln sich Items in der Akzeptanz zum Beispiel oder beim Testen, dass dann alle vom Team hergehen und sagen, dann unterstützen wir jetzt im Test und gucken, dass wir diesen Knubbel aufgelöst bekommen. Dann akzeptiert mal nicht der Product Owner, sondern vielleicht können wir ja mal was akzeptieren. Da sind wir irgendwie nicht hingekommen und vielleicht lag's daran, dass wir das pro Person assignt haben, und nicht pro Spalte. Ich finde wir hätten das ruhig nochmal ausprobieren können.

Interviewer: Wie seid ihr darauf gekommen, dass pro Person zu limitieren? Also war das eine Absprache von euch allen oder…weißt du noch, wie es dazu kam?

Product Owner: Ja, wir hatten mal…also unser Scrum Master war sehr hinterher diesen Durchfluss…weil ganz, ganz am Anfang haben Items echt lange gebraucht, also auch mal einen ganzen Sprint oder zwei Sprints oder drei Sprints und bis die mal fertig wurden. Deswegen war es immer ein Anliegen so ein Item spätestens nach einem Sprint durch ist. Dann hat er immer darauf geschaut, dass Leute nicht plötzlich was Neues anfangen, weil das jetzt bei jemand anderem liegt. Da kam von ihm die Idee die Limitierung einzubauen, sodass nochmal ein stärkerer Anreiz da ist, da halt doch zu unterstützen oder mir halt immer wieder auf die Nerven zu gehen, hey Andreas akzeptierst du das mal, ich komm da nicht voran. Oder hey, wann machst du mal das Code Review oder hey, wann bringst du das mal auf die Testumgebung. Sodass man, worauf man wartet, dem auf die Nerven geht. Also es hat sich schon insgesamt verbessert aber ich denke aber ich denke es hätte noch besser sein können – auf jeden Fall.

Interviewer: Okay, du würdest auf jeden Fall sagen, dass so eine direkte Limitierung zusätzlich zu der indirekten, die halt durch diese Sprints schon gegeben ist, dass das auf jeden Fall noch sinnvoll ist.

Product Owner: Ja.

Interviewer: Okay, weil das hört man ja auch so ein bisschen raus, dass das dann auf jeden Fall was gebracht hat. Okay, cool. Genau, dann bei der nächsten Praktik, da hatten wir ja gestern auch schon drüber geredet, die Planung. Die bei Scrum doch auch einen sehr großen Teil einnimmt und auch sehr viel Zeit. Genau, dass eben diese Items oder auch User Stories die verschiedenen Reifegrade durchlaufen und bei Kanban diese Planung ja eigentlich komplett außen vorgelassen wird, sondern das ist ja mehr das Motto, wir fangen mit dem an, was gerade so ansteht und haben den Backlog, wo wir alle Aufgaben sammeln und dann ziehen wir uns eben einfach immer die Oberste raus. Bei Scrum ist es ja auch noch zusätzlich so, dass die Aufgaben geschätzt werden und das ist bei Kanban…das wird da ja auch komplett weggelassen. Dann die Frage wieder an dich, was der Grund ist oder wieso sollte man dann überhaupt so viel Zeit investieren in die Planung und was ist da der Vorteil? Was bringt das überhaupt?

Product Owner: Naja, also mein Hauptjob als Product Owner ist ja den Wert des Produktes zu maximieren und das Risiko zu minimieren. Risiko minimieren heißt, dass ich möglichst schnell kleinere Inkremente liefere, um zu gucken, ob die eine positiven Effekt haben, auf die Ziele, die ich probiere damit zu erreichen. Also wenn ich jetzt möchte, dass die Autoren mehr Artikel schreiben, dann ergibt es Sinn, irgendwie, wenn ich jetzt sage, dieses Anreizsystem die Top-Autoren auf der Frontseite darzustellen, wenn ich da vielleicht jetzt mit einem sehr einfachen Algorithmus diese Informationen zusammenziehe und da was präsentiere und dieses Feature rudimentär entwickle und kann gucken ob es schon einen positiven Effekt hat bevor ich es irgendwie super groß entwickle, mit einem komplexen Algorithmus, wo ich dann ermittle was sind die Top-Autoren. Deshalb Risiko minimieren. Was Kleines erst liefern, gucken ob es in die richtige Richtung geht. Den Produktwert zu maximieren heißt ja, dass ich die Items entwickle, die mir den höchsten Mehrwert bringen. Dafür muss ich das irgendwie schätzen und dann halt auch die Items entwickeln. Idealerweise schätzen wir in Scrum nicht nur die Komplexität, also das ist ja nicht der Aufwand, sondern wie viel Business Value bringt es, und dann sollte ich als nächsten die Items entwickeln, die einen hohen Business Value

liefern aber eine geringe Komplexität haben. Und dann noch auf mein Sprintziel einzahlen. Sprintziel leite ich ja aus meiner Roadmap ab und aus meinem nochmal übergreifenderen Businessziel, die ich habe. Aus meinem übergreifenden Jahresziel oder Quartalsziel leite ich meine Roadmap ab und daraus leite ich dann meine Sprintziele ab und immer mit diesem Fokus den Produktwert zu steigern. Deswegen ergibt es für mich schon Sinn, diese Planungsstufen zu haben. Warum es die beim Kanban nicht gibt, ist…also soweit ich das weiß kommt Kanban ja aus der Fertigung. Fertigung heißt, da wird ein Auto gebaut bei Toyota und da steht diese Produktionsstraße schon fest. Das sind immer wiederkehrende Schritte. Es gibt verschiedene Arbeitsschritte, die da zu tun sind und ich versuche halt möglichst einen schnellen Durchfluss und einen gleichmäßigen Durchfluss durch diese Produktionsstraße zu haben. Es gibt ja nicht wechselnde Ziele, die ich habe oder worauf ich auslege, sondern es wird immer dieses eine Auto gebaut…und hinten kommen zigtausende von diesen Autos raus. Das hat man jetzt halt versucht in die Softwareentwicklung zu adaptieren und deswegen macht Kanban meiner Meinung…also reines Kanban da Sinn, wo es um…wo einfach Aufgaben zu erledigen sind, wo man nicht vorher überlegt muss ich die machen, muss ich die nicht machen, welche ist jetzt wichtiger, weniger wichtig…ja doch die haben ja auch einen Backlog, man kann ja auch da priorisieren. Also ich glaub ich sehe es vor allem im Betrieb. Ich habe ein Softwaresystem das zu betreiben ist, da muss ich warten, muss regelmäßige Updates machen. Da brauch ich einen neuen Server, da gibt es mal ein Bug, eine Störung. Also…So ein Update kann ich jetzt machen oder später irgendeins so ein Sicherheitsupdate muss ich vielleicht innerhalb zwei Wochen machen und Bug muss ich jetzt sofort irgendwie lösen, deswegen gibt es schon Sinn, dass das eine Priorität hat aber ansonsten…das höhere Ziel ist, die Maschine am Laufen zu haben. Und daran ändert sich erstmal nix, und dann muss ich nur gucken, dass meine Items gut durchfließen. Also für sowas sehe ich das gut. Wo Aufgaben einfach zu erledigen sind und wo es nicht so sehr um strategische Entscheidungen geht, will ich jetzt in die Richtung entwickeln oder in die Richtung entwickeln.

Interviewer: Okay sehr gut. Ich würde vielleicht noch mal mehr bei der Aufwandschätzung mit einsteigen. Ihr hattet ja bei anwalt24 dann auch mit den Storypoints geschätzt. Wie siehst du da den Vorteil gegenüber das in Personentagen zu schätzen?

Product Owner: Ja, also wir haben uns Storypoints geschätzt. Die basieren ja auf den Fibonacci-Zahlen. Die immer so eine Abfolge haben. Ich glaub es war 1,3,5,8,13 … also die Schritte sind immer unterschiedlich groß. Die sollen Komplexität ausdrücken relativ zu anderen Items. Also wenn ich irgendwann mal ein Item hab, das muss ich festlegen, das ist so ein Item, das hat vielleicht einen Aufwand oder eine Komplexität, die ist üblich bei uns. Dann gebe ich dem eine 8. Und dann kann ich gucken, wenn ich ein anderes Item habe, was denkt das Team, ist das komplexer als dieses Referenz-Item? Dann kriegt es eine 13 oder ist es weniger komplex, dann kriegt es eine 5. Dann habe ich das nächste Item. Ist das jetzt eine 5 oder ist es weniger komplex. Na es ist weniger komplex, dann bekommt es eine 3. Oder es ist mehr als eine 5, dann bekommt es eine 8. So kann halt eine Relation herstellen, irgendwann und habe dann eine Komplexität. Die Komplexität sagt halt nichts darüber aus, wie lange etwas dauert, weil ein Super Senior Entwickler, der in diesem Bereich schon tätig ist, der braucht für dieses Item vielleicht kürzere Zeit…für ihn ist das ja aber auch nicht mehr so komplex…das Team einigt sich halt in der Schätzung auch auf eine Schätzung aus dem Gesamtteam heraus. Man spielt ja dann auch dieses Planning-Poker…

Interviewer: Das habt ihr auch gemacht bei anwalt24?

Product Owner: Ja, das haben wir auch gemacht. Ich stelle ein Item vor, und dann überlegt sich jedes Teammitglied selber, was ist für ihn die Komplexität, geheim. Und dann hält man so Karten hoch, also die Zahlen oder schreibt sie irgendwo auf. Und das ist für den Super-Senior-Entwickler…der sagt das ist eine 2 und der Neuling, der Junior, der gerade erst eingearbeitet wird, der sagt das ist eine acht. Und dann können die zwei… Die anderen sagen das ist eine fünf. Dann kann man sich drüber austauschen, wieso sieht der eine das so unkomplex und der andere so komplex. Da geht man schon in eine Lösungsdiskussion rein. Man schafft gegenseitiges Verständnis, der Junior lernt dann vom Senior, aber auch der Senior lernt, warum andere das vielleicht immer noch als komplex ansehen. Und dann kann man sagen, okay…wenn der Junior sagt, ah ja das habe ich verstanden, wenn das so und so ist, dann würde ich das für mich auch auf eine fünf reduzieren. Und der Senior sagt, ja okay das ist ja eine Teamschätzung. Wenn ich das machen, kriege ich das super schnell hin, aber wenn das jemand anders macht, naja dann einigen wir uns mal in Summe im Team auf eine 5. Also so hat man auch schon mal ein Teamagreement erreicht und man setzt die Leute nicht unter Druck. Weil wenn dann dastehen würde, okay das dauert 3 Tage, naja dann könnte jemand ja auch nach drei Tagen auf die Idee kommen, naja, warum ist das denn jetzt noch nicht fertig. Und in Tagen schätzen ist auch schwierig, sind es drei Arbeitstage, aber drei Arbeitstage sind ja nicht 24 Arbeitsstunden, weil es gibt ja auch Meetings und sowas. Dann müsste man sich erstmal drüber unterhalten, was ist denn ein Arbeitstag? Sind das acht Stunden?

Und dann kommt ein Feiertag dazwischen oder jemand wird krank. Es hat ganz viele Nachteile und sorgt für viele Konflikte und wenn man den Leuten einfach vertraut, dass derjenige, der an was arbeitet, so konzentriert daran arbeitet, wie er kann und so flott dran arbeitet, wie er kann, dann ist das eine viel vertrauensvollere Basis für die Zusammenarbeit. Das, die lange Antwort, zum Thema warum in Zahlen schätzen und nicht mit Arbeitsaufwand.

Interviewer: Ja ist gut, also ich habe da jetzt auch…einfach nochmal zur Wiederholung…rausgehört, dass es bei dem Schätzen gar nicht nur darum geht für die Planung, sondern das man schon ins Gespräch kommt und von vornerein schon Sachen klären kann, die jeder so ein bisschen anders aufgefasst hat oder da in den Austausch zu kommen. Okay super, dann zur letzten Praktik…

Product Owner: Wir haben tatsächlich aber irgendwann damit aufgehört.

Interviewer: Wieso? Das klang doch jetzt voll positiv.

Product Owner: Ist auch positiv, ich war auch nicht so begeistert davon, dass wir das aufgehört haben. Wir haben dann halt nicht mehr unsere…das ist ja dann auch ein Punkt dann zum Velocity messen. Kommt das noch oder sind wir da jetzt schon?

Interviewer: Das wäre jetzt quasi…da hätte ich jetzt hingeleitet.

Product Owner: Wir haben am Anfang auch die Velocity gemessen. Also wie viele Storypunkte schafft denn ein Team in einem Sprint und können gucken, dass wir es schaffen, immer eine gleichmäßige Anzahl hinzukriegen. Weil auch wenn das ja immer Sprints heißt, sind wir ja keine Sprint-Läufer, sondern Marathon-Läufer. Wir brauchen über einen längeren Zeitraum eine konstante Geschwindigkeit. Und deswegen ist es gut, wenn sich die Velocity irgendwie einpendelt. Man schafft mal mehr, dann freut man sich. Dann schafft man weniger, das hat dann einen Grund gehabt und dann ist es auch okay. Und über Monate hinweg, sollte die ähnlich sein…kann sich natürlich aber auch steigern. Und dann haben wir angefangen, von unserem Scrum Master initiiert, die Cycle Time zu messen. Also wie viel Tage braucht ein Item bis es fertig ist, nachdem es angefangen wurde. Und das war dann so unsere Messgröße, woran wir gemessen haben, ob wir besser werden. Und dann im Zuge dessen, hat die Velocity nicht mehr so eine Rolle gespielt. Dann haben wir es vereinfacht und einfach nur noch geschätzt…naja dann haben wir doch irgendwie wieder den Aufwand geschätzt. Wir haben dann geschätzt, brauche ich für ein Item eine, zwei oder drei Wochen. Weil unsere Sprints drei Wochen gingen. Also das sollte dann einfach nur noch ein Indikator sein, passt das Item in den Sprint. Weil wir das Ziel hatten, jedes Item soll innerhalb eines Sprint erledigt werden, sodass die Cycle Time unter 21 Tage kommt. Ich fand es komisch davon wegzukommen, von der Storypunkte-Schätzung, weil ich auch dieses Diskutieren darüber, diesen Austausch halt wichtig fand. Der ist dann halt doch noch stärker weggefallen als man sich dann nur noch drauf geeinigt hat, sind es eine, zwei oder drei Wochen. Dann war halt auch wieder das in Wochen schätzen, ist wieder so ein bisschen wie Arbeitsaufwand. Also ich würde das im nächsten Projekt nicht mehr so machen und würde wieder bei den Storypunkten bleiben.

Interviewer: Und dann quasi auch wieder bei der Velocity?

Product Owner: Genau, ich würde die messen. Die Cycle-Time fand ich auch gut. Also die kann man ja trotzdem messen. Es ist ja relativ einfach. Irgendwann geht so ein Item in Progress oder Development und dann fang ich an die Tage zu zählen. Das kann man ja dann trotzdem machen und kann ja aber auch immer noch die Velocity messen.

Interviewer: Okay, eigentlich hast du mir die Frage zu der Velocity und Cycle-Time auch schon damit beantwortet. Deswegen würde ich das jetzt einfach mal überspringen. Und zu anwalt24 habe ich allgemein noch zwei, drei Fragen, zum Verständnis. Man liest ja zu den Vereinbarungen auch von der Definition of Done und of Ready. Habt ihr das auch definiert?

Product Owner: Ja. Also im Scrumguide gibt eine Definition of Done, glaube, keine Definition of Ready. Die Definition of Done haben wir definiert. Die war schon „fertig“ definiert, als ich in das Projekt kam. Und da stehen halt…also man hat ja…ich fang mal mit den Items an. Bei den Items schreibe ich mir ja Akzeptanzkriterien auf, die so ein Item erfüllen muss, damit ich es abnehmen kann, es auf Produktion bringe und es ausliefere. So. In der Definition of Done stehen Dinge drin, die auf jedes Item zutreffen, immer wieder. Sodass ich die halt nicht immer in die Akzeptanzkriterien reinschreiben muss. Da steht sowas drin wie, das Item ist getestet. Es ist vielleicht eine Dokumentation erstellt worden. Es wurden Test erstellt, die Tests laufen grün. Wenn wir was auf Produktion bringen, haben wir immer schon die neusten Updates installiert. Also da können auch so Tätigkeiten drinstehen, die ich nicht

immer wieder priorisieren möchte. Das fand ich gut, dass wir nicht drüber diskutiert haben ist das neuste PHP-Update zu installieren und das neuste Server-Update hier oder das neue Feature zu entwickeln. Sondern wir haben einmal gesagt, wir wollen immer auf dem neusten Stand der Software sein und wenn das nötig ist, dann machen wir das einfach. Da legen wir auch kein Item für an, das tun die Entwickler. Dann wissen sie halt, wenn das ansteht, müssen sie im Planning das im Hinterkopf haben. Nein, dieses mal nehmen wir ein paar weniger Items rein, weil wir wollen auch noch die und die Updates machen. Also so wiederkehrende Sachen und wo man nicht so priorisieren möchte, die kann man gut in die Definition of Done packen. Und die kann man aber auch immer wieder anpassen. Wenn man sagt, dass ist nicht mehr sinnvoll, dass das drinsteht, kann man es rausnehmen. Wenn man was neu hinzufügen will, kann man immer wieder drüber reden. Die Definition of Ready, sowas haben wir nicht formal festgelegt, aber was ja mindestens der Fall ist, dass ein Item, bevor es in den Sprint kommt muss vom Team verstanden werden, es muss geschätzt sein. Das ist ja mal mindestens was ready heißt. Bei uns… wir haben es refinet, das heißt es hatte Akzeptanzkriterien. Wenn es darum ging, dass am Frontend was zu tun ist, war visuell erklärt, was im Frontend zu tun ist. Design war mit dem Item verknüpft. Benötigte Texte und Übersetzungen sollten da sein. Genau. Abhängigkeiten sollten geklärt worden sein. Ja auf sowas haben wir uns im Groben geeinigt, aber das wurde nirgendwo richtig festgeschrieben.

Interviewer: Okay, alles klar. Ich habe nur noch eine abschließende Frage zur Kommunikation. Scrum soll ja super sein, dass durch die festen Events die Kommunikation unterstützt wird. Und gerade ihr hattet ja auch durch die verschiedenen, verteilten Standorte…da ist die Kommunikation gar nicht so einfach. Wie hast du das wahrgenommen? Hilft Scrum da total bei der Kommunikation?

Product Owner: Ja, auf jeden Fall. Wir haben ja mehrere Events. Das Daily hilft jeden Tag…das wir uns mindestens jeden Tag einmal austauschen und drüber reden, wie ist der Fortschritt bezüglich unsers Sprintziels. Bzw. das ist ja ein Event für die Entwickler, also die Entwickler setzen sich zusammen und überlegen sich was haben wir gestern oder heute erreicht bezogen auf unser Sprintziel. Was wollen wir bis morgen erreichen, gibt es Probleme, die uns im Wege stehen? Müssen wir unseren Plan anpassen, dieses Sprintziel zu erreichen. Müssen wir an den Items was ändern, brauchen wir neue? Müssen welche raus? Das können die sich alles im Daily erzählen. Und als Product Owner bin ich ja eigentlich gar nicht Teil des Dailys. Und wenn das Zuhörer bzw. steh stumm da und warte auf Fragen. Dann hat man eine Kontinuität drin und eine Routine. Die Refinements bieten mir immer die Möglichkeit gemeinsam am Produkt zu arbeiten. Was ist das nächste, was wir entwickeln wollen. Was ist sinnvoll, wie wollen wir das ausgestalten? Das ist wieder ein Kommunikationspunkt. Im Planning machen wir uns als Team gemeinsam einen Plan. Worauf wollen wir uns die nächsten zwei Wochen konzentrieren. Haben dann ein gemeinsames Verständnis davon, wo es hingehen soll. Das ist schon auch wichtig, dass man gemeinsames Verständnis entwickelt. Und diese Plannings haben bei mir auch bei anwalt24 teilweise sehr kurz. Wir hatten das Sprintziel in den Refinements vorbereitet, wir hatten die Items vorbereitet. Ich habe sie in eine Ordnung gebracht. Dann sind wir in das Planning rein, das Sprintziel stand fest, die Items standen eigentlich auch schon fest. Wir mussten uns nur noch darauf einigen, wie viele davon nehmen wir denn. Passen die jetzt zum Sprintziel oder nicht? Und dann war das bei uns auch in 15 Minuten auch mal erledigt. Weil alles schon vorbereitet war. Also wir haben ja Planning 1 gemacht, also definiert, was zu tun ist. Und Planning 2 ist ja das im Scrum Guide, wo die Entwickler die Items runterbrechen auf Task-Ebene. Das haben wir nicht gemacht. Aber kenn ich von anderen Projekten, wo man das gemacht hat. Fand ich auch immer ganz gut. Aber das ist ja von den Entwicklern ein Thema. Brauchen die das oder nicht? Im Review habe ich dann nochmal die Möglichkeit unseren Stakeholdern zu zeigen, was wir getan haben. Feedback zu sammeln. Wir können uns selber kritisch draufgucken, können schauen, ob mir…ob es schon Indizien gibt, was wir entwickelt haben, ob wir das Sprintziel erreichen. Das ist manchmal schwer, oder eigentlich ist es sehr oft schwer im Review schon zu gucken, ob das Sprintziel schon erreicht wird. Weil, wenn ich als Sprintziel habe, wir wollen das mehr Autoren mehr Artikel schreiben. Jetzt entwickle ich was. Und nach ein, zwei Wochen schaffe ich was auf Produktion zu liefern. Dann habe ich eine Woche Zeit, wo ich analysieren kann, ob sich schon was tut. Wenn ich jetzt erst gegen Ende vom Sprint was in Summe auf Produktion ausliefere, dann habe ich halt bis zum Review immer weniger Zeit, wo sich was zeigen kann. Deswegen hatten wir das dann auch einfach so gemacht, dass wir jetzt im Review geguckt haben, haben wir das Sprintziel von dem Sprint davor…hat sich da was getan. Sodass wir dann 3 Wochen Zeit hatten, wo sich was hätte tun können. Also nicht nur jetzt im Review schauen, tut sich da was, sondern auch im nächsten. Also je nachdem, wie es halt so ist. Manchmal können sich Sachen auch sehr schnell zeigen. Wenn man sehr viele Nutzer hat und man ändert irgendwas Kleines, dann kann man schon direkt vielleicht auch einen positiven oder negativen Trend sehen. Aber wir hatten halt nicht so viele Autoren. Deswegen hat es halt auch immer eine Weile gebraucht, bis man das sehen kann. Aber ich finde die gut, das sind ja auch keine Meetings, sondern es sind Arbeitsevents, Workshops, wo man gemeinsam am Produkt arbeitet und wenn man die ordentlich gestaltet sind das auch keine Laber-Rhabarber-Meetings.

Interviewer: Okay, cool. Sehr gut, ich glaube, dann bin ich durch mit meinen Fragen. Dann vielen Dank, dass du dir die Zeit nochmal genommen hast.

A2: Transkript des Experteninterviews mit dem Scrum Master von [redacted]

Interview geführt am 30. April 2020 von 14:00 bis 14:40 Uhr

Interviewer: Zurück zu den 5 Praktiken und ich starte mit Commitment, nicht als Wert, sondern als Sprintziel, welches eingehalten wird. Inwiefern ist ein Commitment in Form eines Sprintziels sinnvoll? Wo siehst du die Vorteile in der Praxis?

Scrum Master: Was wir gemacht haben, ist Scrum mit Kanban. Es war nicht optional, ein Sprintziel zu formulieren, es war vorgeschrieben, und wenn dies gut formuliert ist, ist es durchaus sinnvoll. Natürlich ist es in der Praxis nicht immer so, dass diese Sprintziel auch erreicht wird. Aber, wenn das Ziel gut formuliert ist, finde ich ist es eine Art Fokus oder eine Art Leitfaden für den Sprint ist. Ich kann dir auch etwas ziehen aus dem Jira, welche Sprintziele erreicht wurden. Das habe ich schon versucht, aber noch nicht erreicht. Ich habe mal mit dem ehemaligen Scrum Master eine Tabelle geführt, wo diese Sprintziele drinnen stehen, wann wir die Ziele erreicht haben, bei welchem Sprint und welchem Sprint nicht. Das kann man auch deutlich sehen, dass man nach der Einführung von Scrum mit Kanban die Ziele deutlich öfter erreicht wurden. Vielleicht kann man nicht 100%ig sagen, dass die Ziele nur deshalb erreicht wurden, dass Srum mit Kanban eingeführt wurde, aber das hat auf jeden Fall einen großen Impact. Dazu gibt es auch ein Diagramm, was ich dir auch zuschicken kann, wo das dann aufgezeigt wird, dass man vor Kanban nur 25% der Ziele erreicht hat und nach Kanban 31% der Fälle. Ich finde das schon sehr wichtig, dass man eine Sprintziel hat. Wenn man überlegt, dass man im wahren Leben ist es genauso wichtig, dass man sich kleine Ziele setzt, dass man den Fokus darauf setzt und im Auge behält, was wollen wir eigentlich erreichen. Ohne das, das wäre dann nur eine Art User Story abzuarbeiten.

Interviewer: Also als übergreifende Orientierung ist es auf alle Fälle gut, ein Sprintziel zu haben. Ihr habt ja bei anwalt24 auch diese zusätzliche WiP-Limitierung eingeführt. Ich spring dann einfach mal einen Punkt weiter. Bei Scrum ist das ja automatisch so, dass das durch die Sprints vollzogen wird und nur eine gewisse Anzahl von Items in einem Sprintziel und bei Kanban wird das dann noch einmal extra mit einer Anzahl limitiert. Also das ich sage, in einem bestimmten Stadium, also zum Beispiel beim Entwickeln dürfen nur 3 Items liegen und beim Testen dann nur 2 und ihr hattet bei anwalt24 das glaube ich pro Person eingeführt? Wie kam es dazu, also wie habt ihr dieses Limit festgelegt?

Scrum Master: Das ist ein schwieriges Thema, bisschen über der Bachelorarbeit. Das war halt so: wir hatten zwei Teile vom Team, ein Teil in Deutschland, einen Teil in Armenien, das war ein bisschen schwierig. Dann hatten wir ein Team-Meeting, wo alle da waren. Dann haben wir alle gemeinsam den Workflow visualisiert. Da wurde dann auch diese WiP-Limit festgelegt. Das war auch eine sehr leidenschaftliche Veranstaltung, weil die Entwickler da schon gar nicht damit einverstanden waren, dies Limits einzuführen. Am Ende wurde sich doch darauf geeinigt. Ob das dann tatsächlich die richtige Vorgehensweise war, weiß ich jetzt nicht. Es hat schon geholfen, dass man die Arbeit in einer Zeit begrenzen sollte. Ob es tatsächlich sinnvoll war, könnte der ehemalige Scrum Master besser sagen. Ich hatte auch die Vorstellung davon, dass die Entwickler damit unzufrieden waren, das wir das ändern. Wir haben das am Ende, als feststand, dass anwalt24 nicht mehr weiterentwickelt wird, haben wir die WiP-Limits rausgenommen. Aber in dieser kurzen Zeit konnte man mit den Metriken nicht rausbekommen, ob es eine tatsächliche Auswirkung hatte, aber ich kann mir auch vorstellen, dass es besser ist, die Limits nicht pro Person zu benennen, sondern insgesamt auf das Team bezogen oder auf Arbeitssplitter, so und so viele Items in Modification. Also schwieriges Thema. Es gibt nicht die richtige Lösung. Für jedes Team wäre es unterschiedlich. Damit kann man aber vermeiden, dass man immer wieder Aufgaben anfängt und diese dann nicht fertig macht. Wenn man immer etwas Neues anfängt erhöht sich auch die Cycletime vom Anfang bis zum Ende und das hat auch eine ganz schöne Auswirkung insgesamt auf den Workflow.

Interviewer: Ja, also bei Kanban ist es glaube ich auch so in der Theorie, dass man das auch in den verschiedenen Bearbeitungsstatus limitiert, damit man bei dem Workflow auch sieht, wo entstehen da Warteschlangen. Das ist auch der Sinn, das die Leute da eingreifen, dass man zum Beispiel mit testet, wenn da Arbeit angesammelt ist, auch wenn man dafür nicht verantwortlich war. Das findest du generell auch sinnvoll, bloß bei euch hat es nicht ganz so geklappt?

Scrum Master: Definitiv, dass es bei uns nicht so geklappt hat, hat bei uns auch mehrere Gründe, aber deswegen würde ich nicht sagen, dass es nicht sinnvoll ist. Das ist definitiv sinnvoll. Das ist auch eine der wichtigsten Prinzipien von der Verbindung von Scrum und Kanban. Ich würde es empfehlen, das so zu machen, nur vielleicht irgendwie anders. Es ist superwichtig, dass das Team dann damit einverstanden ist. Bei uns waren deshalb viele Konflikte. Und das Team, also die Entwickler haben sich angegriffen gefühlt dadurch und es war ein Gefühl wie in der Kita, wo die kleinen Kinder sagen: „Nee ich mach das nicht“ und auch wenn du aufzeigen kannst, dass es doch sinnvoll ist, ist die menschliche Komponente noch stärker.

Interviewer: Du hattest ja jetzt auch mal mit der Cycle-Time angefangen. Ich würde da jetzt mal zu diesen Metriken springen. Also bei Scrum ist es ja üblich, dass man die Velocity misst und die dann im Burn-Down-Chart darstellt und bei Kanban ist es eher üblich, die Cycle-Time zu messen. Wo siehst du diese beiden Metriken, gibt es bestimmte Vor- und Nachteile, wann man das eine oder andere anwendet?

Scrum Master: Ja, lass mich mal kurz überlegen: natürlich hat irgendwie beides Vor- und Nachteile, Nicht nur mit der Cycle-Time sondern auch mit den Flowmetriken kann man bessere Voraussagen treffen. Bei Scrum kannst du die Metriken nutzen und Vorhersagbarkeit hervorrufen und aufzeigen, wie viele Aufgaben, wie viele Items das Team in dem nächsten Sprint schaffen könnte. Was sind die Vor-und Nachteile? Das ist eine sehr gute Frage. In diesen Metriken bin ich auch nicht der Megaprofi. Der ehemalige Scrum Master ist da viel besser, Lass mal nochmal überlegen.

Interviewer: Allgemein hattest du ja jetzt schon gesagt, dass diese Metriken gut sein können um zu planen und auch zu schauen, was man zukünftig anpassen kann.

Scrum Master: Aber es ändert ja zum Beispiel auch sehr große Effekt beim Daily Scrum. Man schaut auf die fast fertigen Items. Da bleibt man dann nicht immer bei den drei vorgeschriebenen Fragen von Scrum: Was habe ich geschafft? Was muss ich noch tun. Es wird darüber gesprochen, wie das Team es hinkriegt eine fast fertige Aufgabe fertig zu bekommen. Es hat geholfen, dass Leute um Hilfe gebeten haben, dass sie untereinander Hilfe geleistet haben. Ich überlege noch, was mir noch einfällt. Hast du noch Fragen?

Interviewer: Nee, es ist ausreichend. Lohnt es sich beides zu messen: Velocity und Cycle-Time?

Scrum Master: Ja doch, kann man beides machen. Wir haben beides gemacht. Das schließt sich nicht aus. Es ist natürlich eine Sache noch: Das war ja vorhin bevor man Scrum mit Kanban eingeführt hat, hat das Team die Items geschätzt, in Storypoints. Die Estimations machen keinen Sinn mehr, aber das DEV-Team wollte die Estimation (Schätzung) nicht komplett weg lassen und zu 1-2-3 gewechselt, schaffe ich das Ziel in einer, zwei oder drei Wochen. Das kann man zur Diskussion stellen, ob es gut ist oder nicht. Es hat ein Gefühl gegeben, wie groß ein Item ist. Wie viel Arbeit man reinstecken sollte.

Interviewer: Jetzt würde ich nochmal zurückspringen zu der Praktik mit den Arbeitspaketen und Anforderungen also im Scrum Guide steht ja, dass es vermieden werden soll, während eines Sprints neue Anforderungen hinzuzufügen und auf jeden Fall auch keine Änderungen stattfinden dürfen, die das Inkrement gefährden und bei Kanban ist das ja genau andersherum, da können ja jederzeit neue Anforderungen zugefügt werden. Kannst du nochmal sagen, wie du das in der Praxis siehst. Wie weit sollte man die Regeln strikt befolgen oder wann sollte man es lockerer sehen?

Scrum Master: Es kommt darauf an. Erstens ist es Aufgabe des PO, ob man eine neue Aufgabe annimmt. Aber wenn man eine neue Aufgabe annimmt, dann muss man das bewusstmachen, damit man das Sprintziel nicht verändert und gefährdet. Das ist ganz wichtig. Das Sprintziel ist wieder wichtig zu erwähnen. Sonst hätte man bloß die Items, an denen man arbeitet ohne irgendwelche Ziele zu verfolgen! Dann könnte man sagen, das eine Item nehme ich rein, arbeite ich ab und fertig. Man kann Items reinnehmen, ich würde das nur in Ausnahmefällen machen, denn es unterbricht schon ein bisschen deinen Workflow. Du hast geplant, du hast gearbeitet, du verfolgst dein Sprintziel, es kann natürlich auch sein, dass die neue Aufgabe zu deinem Sprintziel passt. Ich würde jetzt nicht grundsätzlich nicht ausschließen. Haben wir auch immer wieder mal gemacht. Ich weiß auch nicht, ob es 100%ig richtig ist oder nicht. Wenn das Team eher fertig war, wenn jemand nix mehr zu tun hätte, hätte man neue Items reingeholt. Dabei finde ich es sehr wichtig, dass diese neuen Items nicht in großen Ausmaßen reingenommen werden und wenn sie oder eine reingenommen wird, das jeweilige Sprintziel nicht gefährdet.

Interviewer: Okay, dass sollte dann immer noch im Team besprochen werden, bevor es dann reingenommen wird.

Scrum Master: Ja. Wie groß ist die Aufgabe, nimmt das nur nach unserer Schätzung nur eine Woche in Anspruch oder drei Wochen. Das nimmt dann schon einen großen Einfluss auf die Arbeit. Zum Beispiel können Leute an ihrer eigentlichen Aufgabe nicht mehr weiterarbeiten, weil sie jetzt an der neuen Aufgabe arbeiten. Die Gefahr ist: es gibt immer irgendwelche adhoc-Aufgaben und jetzt beliebig viele Aufgaben hinzuzufügen, finde ich nicht richtig, denn eigentlich hat man im Backlog festgelegt und priorisiert. Und eigentlich sollte man nichts hinzufügen, nur, wenn es unbedingt notwendig ist und dann muss man darauf achten, was für ein Impact das auf die ganze Sache hat und, ob es das Sprintziel gefährdet. Das ist so meine Einschätzung.

Interviewer: Okay und als ihr dann Scrum mit Kanban bei anwalt24 eingeführt habt, habt ihr die WiP-Limitierung dazu genommen und was habt ihr noch geändert?

Scrum Master: Es wurde ja auch, wir haben unseren Workflow aufgezeigt, aber ich erinnere mich nicht mehr, ob wir daran etwas geändert haben. Vielleicht kann der Product Owner dazu etwas sagen. Sonst außer WiP-Limitierung und der Einführung der Metriken haben sich die Scrum-Events verändert. Bei dem Planning wurden andere Ansätze benutzt zur Diskussion. Ich überlege noch… Es kommt so eine Art, am Anfang hat der vorherige Scrum Master diese Aufgabe übernommen die Rolle als Flowmanager dann ich die die ganzen Metriken und den Flow verfolgt. Das ist auch eine Veränderung der Rollenverteilung vom Scrum Master.

Intervierer: Du meintest ja, dass sich die Events bisschen anders ausgerichtet haben durch die Erweiterung mit Kanban und würdest du sagen, das hat dann auch die Kommunikation bei euch im Team verbessert? Also vielleicht zielgerichteter?

Scrum Master: Also zielgerichteter auf jeden Fall. Die größte Auswirkung hatte es auf Daily Scrum und auf das Sprint Planning. Ob das jetzt eine positive Auswirkung auf die Kommunikation hatte, kann ich schwierig sagen. Ich hatte die ganze Zeit das Gefühl, dass das Team doch ein bisschen dagegen war ist vielleicht nicht direkt dagegen, hat aber nicht so richtig die Vorteile gesehen, warum wir das überhaupt machen. Und dadurch kann ich dir die Frage nicht beantworten. Die Kommunikation ist auf jeden Fall besser geworden als am Anfang, aber ich weiß es nicht, ob es eine Auswirkung von Kanban war. Bestimmt nicht 100%ig. Da haben viel mehr Sachen reingespielt. Es sind auch neue Leute ins Team gekommen und das hat sehr geholfen, dass wir irgendwie sehr offen über alles sprechen konnten, irgendwie hatte ich das Gefühl, dass die Entwickler viel offener, ehrlicher und mutiger ihre Meinung gesagt haben. Einige, die vielleicht früher teilweise vielleicht unüberlegt und leidenschaftlich gesprochen haben – mmmh ist die Kommunikation besser geworden, insgesamt hat Kanban ein bisschen geholfen, aber nicht nur.

Interviewer: Okay, und dann noch mal zur Retrospektive, wie habt ihr das bei anwalt24 durchgeführt. Habt ihr da immer verschiedene Methoden angewandt?

Scrum Master: Am Anfang haben wir das immer online gemacht. Das war auf jeden Fall eine Herausforderung, das kann ich dir sagen, das remote zu machen. Alleine wegen der Kultursache. In jedem Land arbeitet man ein bisschen anders. Das hat man auch in Armenien gemerkt, da hatten wir Herausforderungen mit der Pünktlichkeit und so. Aber das hat sich auch verbessert. Retrospektiven haben wir nur über Video gemacht und jeder hat erzählt, was lief gut und was war schlecht. Es wurden auch keine Sachen festgehalten oder transparent gemacht. Also jemand hat es aufgeschrieben und dann haben wir ein Tool eingeführt, das heißt RETRIUM, das war cool. Das ist eine Retrospektive, wo Teilnehmer auf ein Board live wie auf Post-Its ihre Gedanken aufschreiben können. Die sind erstmal verdeckt, also andere können nicht sehen, was die einzelne Person schreibt und da gibt es dann unterschiedliche Methoden, wie du das machst. Wir haben von dem ganz einfachen: was war gut, was war schlecht bis zu Scrum-Events oder Scrum-Werte bewertet so ist ein Spinnennetz entstanden, wo man sehen konnte, welches Event das Team als bestes oder welches das schlechteste war und dann konnten wir gezielt diskutieren, wie man das Review verbessern konnten. Das ist ein cooles Tool, kann ich dir schicken. Kostenfreie Version ist möglich. Ich habe vom Product Owner Tipps bekommen, wie wir unsere Arbeit verbessern können um unsere Ziele zu erreichen. Was sind Sachen, die uns zurückwerfen und Sachen, die uns motivieren und uns helfen. Da haben wir einmal eine Retrospektive gemacht. Ich habe so eine Art Schiff gebastelt, kann ich dir auch schicken: wie so eine Art Symbol, wo so der Anker von dem Schiff…der Wind ist das, was uns vorwärts bringt und das ist sehr spielerisch, aber das macht mehr Spaß. Da kann man schon kreativ sein. Was ich noch ganz witzig fand, um herauszufinden, wie insgesamt die Laune im Team ist, so eine Art „Happiness-Index“. Dann habe die Leute gebeten, dass sie sich für die Retrospektive für sich ein Bild überlegen oder aussuchen zur Veranschaulichung, wie sie sich im letzten Sprint gefühlt haben. Ich weiß noch, dass vom Prdouct Owner eine Schildkröte kam, die auf einem Berg hochklettern möchte… sind witzige Sachen rausgekommen. Es ist bei der Retrospektive sehr wichtig, dass man guckt, was ist im Sprint passiert und worauf sollte man achten, du kannst das auch lenken und am Ende sollte man darauf achten, dass diese Actionpoints auch festgelegt werden. Und dass man nicht nur diskutiert und dass man sich auch einigt am Ende, was man tatsächlich verbessern möchte. Das haben wir manchmal nicht gemacht. Von den Punkten zu denen man sich geeinigt hat, sollte man wenigstens eine Sache in den Backlog vom nächsten Sprint geben. Sonst habe ich das im Confluence festgehalten, aber dann wurden die Sachen vergessen. Das war auch meine Schuld. Ich sollte da auch häufiger drauf gucken und das Team darauf hinweisen. Aber es ist auch schwierig, wenn es im Confluence auf irgendeiner Seite steht. Aus meiner

Sicht wäre es aber auf jeden Fall wichtig, dass das Team zu mindestens eine Sache davon tatsächlich in den Sprint reinpackt und aktiv daran arbeitet.

Interviewer: Aber bei der Retrospektive haben dann alle teilgenommen?

Scrum Master: Ja, aber ich habe mich manchmal von dem Inhaltlichen zurückgezogen, aber meistens habe ich mitgemacht. Als Scrum Master ist es eine Herausforderung, dass du mitmachst und gleichzeitig moderierst und aktiv zuhörst. Es hat am Anfang geholfen, dass wir das auch manchmal mute, anonym gemacht haben. Manchmal haben die Leute das dann vorgelesen und manchmal war es auch besser die Sachen unpersönlich, anonym aufzuschreiben und sagen. Das hat bei uns geholfen. Manchmal haben die Leute auch von ganz allein ihre Namen genannt.

Interviewer: Jetzt zum Abschluss nochmal die Frage, wenn du noch einmal Scrum Master bei einem neuen Projekt wärst, würdest du es noch einmal probieren mit Scrum und Kanban? Oder würdest du reines Scrum einführen?

Scrum Master: Also so 100%ig kann ich dir das gar nicht sagen. Ich finde schon, dass es eine gute Sache ist, aber es war mein erstes Team, dadurch habe ich auch nur diese eine Erfahrung und auch noch keine reine Scrum-Erfahrung. Aber ich würde auf jeden Fall sagen, dass es vom Team abhängig ist. Man muss die Umstände abwägen.

Interviewer: Okay, alles klar. Dann bin ich durch mit meinen Fragen. Danke dir!

A3: Transkript des Experteninterviews mit dem Agilen Coach von [redacted]

Interview geführt am 04. Mai 2020 von 15:00 bis 15:45 Uhr

Interviewer: Genau, vielleicht können wir da ja auch starten. Wie kam es denn überhaupt dazu, dass Scrum praktiziert wurde? Was waren die Gründe dafür, dass Scrum mit Kanban eingeführt wurde? Kannst du da vielleicht ein bisschen was zu den Anfängen erzählen?

Agile Coach: Also als ich da hinkam, Mitte 2017, hatten die ja eine sehr große Migration hinter sich. (…) Also ich habe ja bei Wolters Kluwer angefangen als Agile Coach. Und dann hat mein Chef gesagt, guck dir mal anwalt24 an, das läuft nicht so gut, was die da eigentlich machen. Und dann haben die mir alle gesagt wir machen Scrum, aber eigentlich hatte das mit Scrum nicht wirklich viel zu tun. Es gab sehr viele interne Teamkonflikte, an denen man arbeiten musste, um die aufzulösen. Damit man sich so erstmal besser verstanden hat, hat schon eine ganze Weile gedauert. Und die hatten extreme Schwierigkeiten, und haben es auch nicht eingesehen, häufiger zu releasen. Und wenn wir releast haben, hat es halt immer gekracht und es ging halt ziemlich viel schief und in Produktion waren ziemlich viele unerwartete Fehler auch. Lass mal zeitlich überlegen. Genau, so dann bin ich ja Professionell Scrum Trainer. Und dann ungefähr, das war glaube Anfang 2018 gab es so ein Angebot an dem Professionell Scrum with Kanban Training teilzunehmen, das war damals noch ein Beta-Training. Naja, und dann habe ich halt viele Vorteile gesehen und war auch von dem Ansatz total begeistert und hatte natürlich auch mein Spielfeld das mal auszuprobieren und dann habe ich damit erstmal so undercover angefangen. Und dann so suggsezive, das nach und nach in dem Team implementiert. Ja und dann haben wir das ja auch tatsächlich geschafft, irgendwie diese ganzen Teamkonflikte dann aufzuräumen. Wir haben den gordischen Knoten dann auch irgendwie geplatzt bekommen. Dazu hat uns das glaube auch viel geholfen, weil wir viel, also die Diskussionen die wir hatten waren dann nicht mehr so abstrakt und subjektiv, sondern konnten objektiv geführt werden auf Grund der Zahlenlage, die wir hatten. Und wie schon vorhin erwähnt, wir hatten ja echt Schwierigkeiten zu releasen und das auch sicher zu releasen. Und am Anfang war der Releasezyklus aller drei Monate vielleicht mal und nachher war das eher so Low-Event geworden, so einmal die Woche oder mindestens einmal im Sprint. Das hat sich schon extrem verbessert.

Interviewer: Okay krass. Und da würdest du jetzt sagen, dass es an dem Scrum mit Kanban lag oder würdest du sagen, das hättet ihr auch erreicht, wenn ihr reines Scrum eingeführt hättet.

Agile Coach: Das Scrum mit Kanban hat da auf jeden Fall definitiv geholfen.

Interviewer: Okay ich hatte bei den Interviews mit dem Scrum Master und dem Product Owner schon erfahren, dass ihr eine WiP-Limitierung eingeführt hattet. Bei Scrum ist es ja normalerweise so, dass nur indirekt durch die Sprints eine Limitierung vollzogen wird und das ist ja gerade so eine Kanban-Praktik das man die WiP-Limits genau festlegt. Wie seid ihr zu der Entscheidung gekommen, die WiP-Limitierung pro Person einzuführen? Eigentlich soll ja kein Entwickler ein Item besitzen.

Agile Coach: Das war ja immer voll die heiße Diskussion mit den WiP-Limits und war auch eine der Hauptschwierigkeiten das dir damit hatten, dass die mit allen möglichen Items angefangen haben und die Sachen dann nicht zu Ende gebracht haben. Und als ich damit angefangen habe, lagen da auch Items in dem Product Backlog, wo die irgendwie schon über 100 Tagen dran „am Arbeiten" waren. Das waren so die ersten Diskussionen, die wir hatten. Und dann gab es immer irgendwelche Gründe mit was Neuem anzufangen. Da dieses Team auch so geteilt war, in Münster und in Jerewan, und es in Münster noch ein Team-Leader gab und in Jerewan ein Team-Lead gab. Es war super schwierig…es hat sich zwar gebessert aber es war immer noch so, das einer für ein Item zuständig war. Wir haben das hinbekommen, dass sie sich dann mehr unterstützt haben gegenseitig, aber die wollten halt nicht davon abrücken, dass jemand für ein Item zuständig ist. Naja ich kann es ihnen ja auch nicht alles aufoktroyieren, wenn sie es partout nicht wollen. Und diese zwei Items pro Person war dann irgendwann deren Entscheidung gewesen. Und dann dachte ich mir, okay, dann ist das jetzt erstmal so. Es ist ja auch gut, dass mal eine Entscheidung getroffen wurde und zwei ist ja jetzt auch nicht mal besonders viel. Von daher war ich mit deren Entscheidung dann auch ganz zufrieden.

Interviewer: Hat das dann aber geholfen? Also hat es das verbessert, dass so viele Items angefangen werden und nicht beendet werden?

Agile Coach: Sagen dir diese Flow-Metriken was? Cycle-Time, Throughput?

Interviewer: Ja, also ich hatte mir das schon mal angeschaut, aber ich bin noch lange kein Profi darin.

Agile Coach: Okay, also Cycle-Time ist ja, wenn du angefangen hast, wie lange es dann dauert bis du es fertig hast. Das kannst du ja pro Item messen, empirisch. Und nachher hast du dann eine Wahrscheinlichkeit von so und so viel

Prozent, 85 Prozent normalerweise, in denen du dann sagst, 85 Prozent unserer Items, historisch gesehen, werden in 21 Tagen oder kürzer fertig. Wir haben angefangen von einer Cycle-Time von 85 Prozent mit irgendwie 124 Tagen. Und das endete dann bei 21 Tagen. Und der Throughput, das ist ja, da kannst du ja auf den Sprint sehen, wie viele Items werden im Sprint fertig oder normalerweise, wie viele Items werden dann pro Tag fertig. Und am Anfang hast du gesehen, dass immer in so Batches gearbeitet wurde, das, was du in Kanban vermeiden willst. Das auf einen Schwung dann mal 10 Items oder 20 Items oder so fertig waren. Und mit der Zeit hat sich das dann so verbessert, das es gleichmäßig war. Das dann jeden Tag, naja nicht unbedingt gleich viel Items fertig wurden, aber das fast jeden Tag immer irgendwas fertig wurde. Und das hat auf jeden Fall dazu geführt, auch immer diese Diskussionen, die wir dann führen konnten, warum das nicht fertig wurde und was wir jetzt tun können, um das fertig zu kriegen und so. Das wir die Dinge auch einfach früher fertig gekriegt haben. Und dann gibt es ja auch dieses Work Item Age. Das ist die nächste wichtige Metrik, in diesen Flow-Metriken. Das heißt, wenn du damit angefangen hast, weißt du ja jeden Tag wie lang du halt schon dran bist. Und das haben wir uns immer im Daily Scrum angeguckt und üblicherweise werden ja immer diese drei Fragen im Daily Scrum gestellt. Was hast du gestern gemacht? Was machst du heute? Was hast du für Hindernisse? Hindernisse hat eh nie jemand, weil er kein Bock auf Daily Scrum hat, so in die Richtung. Und wir haben das dann halt geändert. Da gibt es so ein Chart, das Work-Item-Age-Chart. Da kannst du alle Items drauf sehen und wie lang die Leute jetzt schon dran am Arbeiten sind. Und dann sind wir immer von rechts nach links durchgegangen. Rechts ist halt möglichst nah an fertig und links ist gerade erst angefangen. Und haben dann in der Regel, angefangen. Was ist in Produktion neues aufgetreten? Und dann die Items halt von diesem fast fertig und haben schon besonders hohes Work-Item-Age angefangen zu besprechen. Dann zu klären im Team, was können wir dafür tun, um diese möglichst schnell fertig zu kriegen. Das hat die Dynamik halt komplett verändert.

Interviewer: Okay, und diese Metriken habt ihr dann nur für das Daily genutzt? Oder habt ihr das auch beim Planning genutzt?

Agile Coach: Nein, die kannst du für alles Mögliche benutzen. Die kannst du für den Forecast benutzen, die kannst du für das Planning benutzen, in der Retrospektive, im Review haben wir die auch benutzt. Damit kannst du halt zum Beispiel so eine Prognose darauf machen, wann jetzt die nächsten 50 Items im Product Backlog fertig sind. Oder da kannst du im Sprint Planning sagen, im nächsten Sprint kriegen wir, mit dem Throughput, den wir historisch gemessen haben, im nächsten Sprint 10 Items fertig. Und damit brauchst dann eigentlich auch nicht mehr zu schätzen.

Interviewer: Und das hast du dann immer gemessen und dann mit dem Team besprochen. Und die haben das dann auch gut angenommen, mit den Metriken zu arbeiten?

Agile Coach: Es gab ziemlich viele Diskussionen am Anfang, aber nachher haben eigentlich alle den Vorteil eingesehen und dann auch ganz gut damit gearbeitet. Dann habe ich auch ein Tool, mit denen kann man sich die ganzen Charts und so anzeigen lassen. Aus lizenztechnischen Gründen konnten wir das nicht ins Jira integrieren, deswegen haben wir so eine Excel-Liste gepflegt, manuell. Das hat nachher der Scrum Master gemacht und am Anfang ich. Das war jetzt bei dem kleinen Team nicht so viel Arbeit. Bei einem anderen Team haben wir das dann automatisiert, da haben wir dann ein Skript drüber laufen lassen und das hat die Zahlen dann von alleine rausgezogen. Du musst dann halt immer gucken das Datum, wenn sich eine Spalte ändert, wie das datiert ist, damit du das nachverfolgen kannst. Und dann kannst du auch sehen, wir haben die meiste Zeit immer im Test oder in der Integration oder wo auch immer dann verbraucht. Das kannst du dann alles nachvollziehen.

Interviewer: Okay und das macht dann aber der Scrum Master, der diese Metriken pflegt oder gibt es dafür dann eine neue Rolle?

Agile Coach: Ich würde sagen das ist eher eine neue Fähigkeit. Eine neue Rolle ist in Scrum ja so nicht vorgesehen. Könnten wir sagen, wir haben jetzt sowas wie einen Flow-Manager, da das Development-Team diesen Skill, diese Fähigkeit nicht so gerne annehmen wollte, habe ich sie dann übernommen. Und nachdem ich dann weg war, ist das der Adrienn auch schwerer gefallen, die dazu zu bewegen, das auch einzuhalten.

Interviewer: Also würdest du sagen, aus deiner praktischen Erfahrung ist es schon hilfreich eine neue Rolle mit einzuführen?

Agile Coach: Nein, aus meiner Überzeugung als professional Scrum Master, muss ich natürlich sagen, gibt es natürlich keine neue Rolle. Man braucht halt nur neue Fähigkeiten, die halt irgendjemand ausfüllen muss. Am besten

natürlich aus dem Development-Team. Wenn du jetzt natürlich pures Kanban machst, dann kannst du eine Rolle auch einführen, wenn du magst.

Interviewer: Okay und zu den Metriken. Bei Scrum werden die Items ja eigentlich in Story-Points geschätzt und dann die Velocity als Metrik genutzt. Ich habe schon gehört, dass ihr das bei anwalt24 am Anfang auch gemacht habt und dann nicht mehr. Magst du dazu noch was erzählen? Wieso habt ihr das dann nicht mehr gemacht habt?

Agile Coach: Diese Velocity…also dieses Schätzen mit relativen Punkten machst du ja einerseits halt um danach ein Instrument zu haben, um Forecast zu geben. Einerseits auf den gesamten Product Backlog, wann wird das fertig und andererseits für den Forecast für den Sprint, wie viele Items können wir uns im nächsten Sprint vornehmen. Es ist ja sehr gut, wenn du diese Schätzung dazu nutzt, um eine Unterhaltung anzuregen, um herauszufinden, ein gemeinsames Verständnis, was machen wir da eigentlich und was ist alles dabei zu bedenken. Was in der Realität aber häufig dabei passiert ist, dass die Schätzung dazu führt, dass genau das Gegenteil passiert. Dass die Unterhaltung nämlich unterdrückt wird, weil wir schnell irgendwas schätzen und uns dann nicht mehr unterhalten brauchen. Und die Entwickler sich dann wieder hinter ihren Computer setzen können und weiter hacken können. Und das mit Velocity und relative Schätzen und so, würde ich mal sagen, also das würde ich mal als traditionellen agilen Ansatz bezeichnen. Und diese Flow-Metrics-Kanban-Ansatz ist halt eher der, ich nenne es mal, moderner Ansatz, der aber Im Grunde auch mehr mit der Agilität und Scrum zusammenpasst, weil es auch ein empirischer Ansatz ist. Eigentlich halt nicht mit Vermutung zu tun hat, sondern mit Fakten zu tun hat. Wenn du die Zahlen hast, brauchst du eigentlich nicht mehr zu schätzen. Weil du, historisch gesehen, ja alle Daten hast und eine Wahrscheinlichkeit hast, mit der du weißt, wann die anderen Items fertig werden oder wie lang es dauert. Das trifft natürlich jetzt nicht auf jedes einzelne Item zu, aber bei der Schätzung ist es ja genauso, dass du dich grob verschätzen kannst.

Interviewer: Und ihr hattet dann die Schätzung dann komplett sein gelassen?

Agile Coach: Nein, wir haben das dann immer noch gemacht. Aber wir haben die Schätzung geändert, dahingehend, dass wir entweder 1,2 oder 3 Punkte vergeben haben. Und ein Punkt bedeutete, wir gehen davon aus, dass wir, wenn wir damit anfangen zu arbeiten, in einer Woche fertig sind. Das ist ja auch eine andere Aussage, als wenn ich sage ich gebe einen Punkt, brauche dafür 10 Minuten und bis das dann fertig ist, kann es dann trotzdem 10 Tage dauern.

Interviewer: Okay. Und was habt ihr…also ihr habt ja die WiP-Limits eingeführt, dann habt ihr mit den Flow-Metriken gearbeitet und hat sich noch was verändert, dadurch, dass ihr Scrum mit Kanban eingeführt habt. Oder waren das die größten Punkte, die sich bei euch im Projektmanagement geändert haben?

Agile Coach: Ich würde mal sagen, der größte Punkt, der sich dadurch geändert hat, ist das wir häufiger releasen konnten und das wir zielgerichteter Diskussionen haben. Und von daher war das halt ein ganz gutes Mittel, was wir eingesetzt haben. Und dadurch hatte der Product Owner dann häufiger die Möglichkeit, seine super Ideen auszuprobieren.

Interviewer: Ich hatte auch vom Scrum Master oder Product Owner gehört, dass ihr beim Sprint häufiger schon fertig gewesen seid und die Items alle schon abgearbeitet waren und ihr euch dann während des Sprints neue Items ziehen musstet. Wie kam es dazu?

Agile Coach: Grundsätzlich war es im Sprint früher so, dann war quasi das Sprintende gleich der Zeitpunkt, wo du releasen wolltest. Dadurch das wir dann halt häufiger releasen konnten, wurde das Sprintende auch weniger relevant dafür, dass wir mit der Arbeit fertig sein wollten. Und dann hat sich dieser Zyklus von dem Sprint, die drei Wochen waren das bei uns, entkoppelt von dem Release-Zyklus der halt seine eigene Dynamik bekommen hat. Naja und dadurch, dass wir in der Regel auch immer schneller fertig geworden sind mit irgendwas, und das was wir gelernt haben, bestand häufiger die Möglichkeit irgendwas Neues in den Sprint zu holen. Immer unter der Berücksichtigung, das Sachen fertig geworden sind. Darauf habe ich dann schon immer geachtet, weil naja die Tendenz war halt immer noch da, diese WiP-Limits zu ignorieren.

Interviewer: Okay. Ich habe noch eine Frage zum Commitment. Bei Kanban sind diese Commitments ja optional und bei Scrum wird ja dieser Forecast getroffen und auch das Sprintziel als Commitment festgelegt. Inwiefern siehst du das als sinnvoll an, so ein Commitment in Form eines Sprintziels zu setzen, auch wenn dann nicht immer das Sprintziel eingehalten wird?

Agile Coach: Das Sprintziel ist das Wichtigste, an dem wir hier arbeiten. Es ist natürlich extrem wichtig, dass wir unser Sprintziel auch erreichen. Das heißt jetzt natürlich nicht, dass wir alle unsere Items, die wir in den Sprint Backlog gesetzt haben, umgesetzt werden müssen. Das ist ein häufiges Missverständnis, was viele noch benutzen. Das viele sich darauf commiten, diese Arbeit fertig zu kriegen. Aber wir sagen, wir sind ja commitet, wir geben unser bestes, das Sprintziel zu erreichen. Und wenn wir merken, dass bei unserem Plan den wir gemacht haben und der jetzt in unserem Sprint Backlog definiert ist, mit dem wir unser Sprintziel erreichen wollen, irgendwas nicht stimmt. Dann passen wir halt den Sprint Backlog an, um trotzdem bestmöglich unser Sprintziel zu erreichen.

Interviewer: Der Scrum Master hatte mir auch schon das Diagramm gezeigt, dass ihr durch die Einführung von Scrum mit Kanban erreichen konntet, dass ihr mehr Sprintziele erreichen konntet.

Agile Coach: Ja, das hat sich schon stark verbessert. Ich habe es später dann nicht mehr nachgemessen, weil ich dann ja auch nicht mehr Teil von dem Team war, aber der Teil, den wir erreichen konnten, ist stark gestiegen.

Interviewer: Was meinst du, was waren die Einflüsse, die dazu geführt haben, dass das verbessert wurde?

Agile Coach: Naja, das wir halt mehr Fokus hatten. Und das wir tatsächlich darauf geachtet haben, dass wir die Dinge fertig kriegen. Und wir auch ein Instrument hatten, da ständig draufzugucken und festzustellen, dass es auch fertig wird. Wie ich eben meinte, dass wir uns im Daily Scrum halt nicht jeder sagt, was hast du jetzt gemacht und jeder findet Ausflüchte, warum ich halt doch noch einen Tag brauche. Sondern wir uns immer darauf fokussiert haben, was können wir jetzt tun, um das fertig zu kriegen.

Interviewer: Okay. Du meintest ja auch, dass als ihr das dann implementiert habt, dass viele Items da schon angefangen wurden, aber nicht beendet und dadurch lange rumlagen. Was habt ihr mit diesen Items gemacht? Wie seid ihr damit vorgegangen?

Agile Coach: Naja, die meisten von diesen Items, die die irgendwann mal angefangen haben, waren nicht mehr relevant und wir haben sie weggeschmissen. Und die paar Sachen, die dann übriggeblieben sind, haben wir dann fertig gestellt.

Interviewer: Okay. Und hattest du das Gefühl, dass Verständnis vom Entwicklungsteam kam? Das so ein Fokus, den du ja jetzt auch immer wieder beschrieben hast durch die Flow-Metriken und durch den allgemeinen Fokus auf den Workflow auf Verständnis bei den Entwicklern getroffen hat?

Agile Coach: Auf jeden Fall. Es gab so ein paar Brennpunkte, an denen wir uns immer aufgerieben haben.

Interviewer: Zum Beispiel?

Agile Coach: Diese WiP-Limits auf jeden Fall. Okay und ein anderer Punkt war noch gewesen, du darfst diese Items nicht mehr zurückschieben. Also es gibt ja das Kanban-Board mit den einzelnen Spalten und wenn das einmal weitergewandert ist, darfst du das nicht mehr zurückschieben. Und vorher haben die halt immer Pingpong gespielt. Dann war das im Test, dort ist dann irgendwas aufgefallen und dann ist das zurück zur Entwicklung gesprungen, und wieder vor und zurück. Das habe ich dann halt irgendwann unterbunden und das führte dann zu einer riesigen Diskussion. Weil ich denen jetzt irgendwie vorschreiben würde, wie ihr Arbeitsablauf ist, weil sie wären ja selbstorganisiert und so. Ja, haben sie natürlich nicht ganz unrecht mit. Wenn du das dann aber gestattest, dass die dann die Dinger wieder zurückschieben kannst, kommen halt deine ganzen Metriken durcheinander. Davon abgesehen, wenn wir jetzt wieder daran denken, dass du möglichst schnell sehen willst, dass es möglichst schnell fertig wird, ist es ungünstig, wenn du die Dinger immer wieder vor- und zurückschiebst.

Interviewer: Ja klar. Und bei den WiP-Limits, was war da die Diskussion?

Agile Coach: Ja, wir können daran jetzt halt nicht weiterarbeiten, und wollen hier jetzt nicht rumsitzen. Wir müssen jetzt mit irgendwas Anderem anfangen. Mal so grob zusammengefasst.

Interviewer: Dann nochmal eine Frage zu den Anforderungen, zu der Praktik. Also bei Scrum ist es ja so gedacht, dass man während eines Sprints möglichst vermeiden soll neue Arbeitspakete dazuzugeben und neue Items, sondern man soll ja eigentlich die Iteration abwarten. Bei Kanban ist es ja so, dass Änderungen jederzeit hinzugefügt werden können.

Agile Coach: Ja das ist auch eher klassisches Scrum von dem du jetzt sprichst.

Interviewer: Okay, also da würdest du jetzt sagen, dass muss man nicht so strikt befolgen, bzw. das ist in der Praxis auch sehr schwer, weil immer ad hoc dann irgendwelche Änderungen reinkommen?

Agile Coach: Wenn was anliegt, muss ich es natürlich machen. Wenn jetzt ständig irgendwas ändert und jemand ankommt der was geändert haben will, und der Product Owner das jedem Recht machen wollte, haben wir natürlich ein Problem. Aber natürlich kann es gute Gründe geben, dass wir mit irgendwas Neuem anfangen zu arbeiten. Zum Beispiel wir haben ein Fehler in Produktion. Na klar warten wir da jetzt nicht noch zwei Wochen mit, bis der Sprint zu Ende ist. Wenn es wichtig und gravierend ist. Wäre ja Blödsinn, wenn du damit wartest, nur, weil du dir was Anderes vorgenommen hast.

Interviewer: Ja klar. Aber an sich ist es ja so gedacht, dass eben das vermieden werden soll, das so viele Aufgaben gleichzeitig in Arbeit sind und nichts fertiggemacht wird. Sondern das man sich den Sprint dafür mal die Zeit nimmt.

Agile Coach: Ja…ja. Und wenn diese Arbeitspakete halt immer kleiner werden, ist es auch leichter abzuschließen, bevor du mit etwas Neuem anfängst, als wenn du immer so große Arbeitspakete hast. Da ist die Tendenz, dass du immer mit irgendwas Neuem anfängst höher. Und deswegen sollte halt auch das Sprintziel wohl überlegt sein, weil das das Wichtigste ist an dem wir arbeiten. Das heißt jetzt natürlich nicht, dass alles was wir im Sprint tun auf dieses Sprintziel einzahlt. Das du dadurch halt immer noch Flexibilität hast, irgendwelche anderen Dinge, die jetzt wichtig dazu kommen, auch noch erledigen kannst. Das kennst du ja wahrscheinlich aus deinem privaten Leben auch, du hast irgendein Ziel und dann passiert ständig irgendwas, was du doch noch dazwischenschieben musst.

Interviewer: Ja, das stimmt. Okay, dann nochmal eine Frage zur Planung. Da ist ja auch ein ziemlich großer Unterschied zwischen Scrum und Kanban. Scrum hat ja diese verschiedenen Planungsstufen: die Roadmap, im Product Backlog die Items dann schon detaillierter ausarbeitet bis sie dann in den Sprint Backlog gezogen werden. Bei Kanban gibt es ja nur die Aufgabenplanung durch den Backlog, wo die Items priorisiert werden und die oberste Aufgabe gezogen wird. Wo siehst du da die Vor- und Nachteile?

Agile Coach: Okay, also in Kanban gibt es ja eigentlich nur das Kanban-Board und da sind dann alle Spalten drin, die du benötigst in deinem Workflow. Da wäre dann aus Scrum-Sicht der Product Backlog noch drin. Da spielt die Lead-Time dann vielleicht auch wieder mit rein. Der Unterschied zwischen Lead-Time und Cycle-Time ist halt, bei Cycle-Time hast du wirklich angefangen und bei Lead-Time weißt du nicht, ob du das vielleicht doch nicht anfängst. Wenn du den Product Backlog da mit einbeziehen würdest, würdest du eher mit Lead-Time sprechen, weil du nicht genau weißt wann du damit anfängst. Kannst du jetzt deine genaue Frage nochmal wiederholen bitte?

Interviewer: Ja, meine Frage ist, ob man diese Planungsstufen, die Scrum hat, ob sich das wirklich lohnt oder ob das überspitzt gesagt „Waste of Time“ ist und man wie in Kanban nur rein mit dem Backlog arbeiten kann?

Agile Coach: Das würde ich nicht sagen, dass das „Waste of Time“ ist, weil wir bereiten das ja so gut vor, dass wir das gut genug verstehen, um mit der Entwicklung, also mit der tatsächlichen Umsetzung anzufangen. All das darüber hinaus, ist dann vielleicht „Waste of Time“, wenn man es zu sehr spezifiziert, aber das ist ja auch nicht der Plan in Scrum. Das soll ja immer noch leichtgewichtig sein. Und ich geh mal davon aus, in Kanban wirst du das natürlich auch machen, dass du, bevor du anfängst, überlegst, was du da eigentlich machst. Aber Kanban sagt da ja explizit nichts drüber aus über diese Phasen und so, das ist da ja viel freier. Bei Kanban ist ja in der Regel, dass du ein Kanban-Board hast mit der du deine Arbeit dann transparent machst. Und in Scrum hast du dann quasi zwei Kanban-Boards. Das eine ist der Sprint-Backlog und dann hast du noch den Product-Backlog, und wenn du den jetzt auch noch an dieses andere Board dranmachst, dann hat man ja im Prinzip ein Kanban-Board. Scrum setzt mir halt den Rahmen, in dem die Arbeit abläuft und Kanban fokussiert mehr konkret auf diesen Fluss, auf diesen Arbeitsfluss. Wie du den Rahmen dann bildest, indem das optimiert wird dieser Fluss, ist dir halt frei gelassen. Und deswegen halt diese Kombination von Scrum mit Kanban. Du optimierst den Fluss und hast den Rahmen vorgegeben. Aber in Scrum wird wiederrum nichts gesagt, wie du diesen Arbeitsfluss optimierst. Und da kann da halt Kanban helfen.

Interviewer: Mit den Fluss-Metriken.

Agile Coach: Genau.

Interviewer: Okay und der Scrum Master hatte auch erzählt, dass als ihr Scrum mit Kanban eingeführt habt, euch getroffen hattet und euren Workflow von anwalt24 visualisiert habt.

Agile Coach: Ja also der Scrum Master war noch gar nicht wirklich dabei, als wir damit angefangen haben. Du musst halt eigentlich, wenn du Scrum mit Kanban machst, wie du es eigentlich machen sollst, deinen Workflow visualisieren. Das solltest du halt eigentlich am Anfang machen. Das war eins der Dinge, die ich so undercover erstmal für mich gemacht habe und wo wir dann aber nochmal explizit im Workshop, wo wir alle zusammen waren, dann darüber gesprochen haben. Und dann auch sowas wie Policies festgelegt haben. Was in dem einen Schritt passiert, oder in dem anderen Schritt passiert. Was passieren muss, damit so ein Phasenübertritt stattfinden kann. Z.B. wann ist ein Item zum Testen fertig? Wer testet usw. Das haben wir dann alles definiert und in dem Zusammenhang kam dann auch das mit den zwei Items pro Entwickler als WiP-Limit. Dann hatten wir so drei oder vier Flipcharts nebeneinander geklebt und hatten da unser Board.

Interviewer: Okay und von den Flow-Metriken, die du für anwalt24 gemessen hast, hast du da zufällig noch welche, die du mir zuschicken kannst. Der Scrum Master hatte mir auch deinen Artikel bei scrum.org geschickt, da ist ja auch dieses Kuchen-Chart drin. Dürfte ich das bei meiner Bachelorarbeit nutzen?

Agile Coach: Ja, schon. Also damit habe ich natürlich irgendwann aufgehört, das weiter zu verfolgen. Ich weiß nicht genau, wie sich das entwickelt hat. Ich glaube aber mit der Zeit, dürfte diese Auswertung noch positiver ausfallen. Die anderen Daten, also ich weiß nicht genau, ich habe die nicht mehr. Und ich weiß auch nicht genau, was jetzt damit passiert, wenn man die in dieses Tool reinwirft. Weil wir haben jetzt Mai 2020 und wir haben aufgehört damit keine Ahnung wann letzten Jahres. Keine Ahnung was das Tool dann daraus macht. Man müsste dann vielleicht die Uhr irgendwie zurückstellen oder so. Ich weiß jetzt nicht genau, wofür du das brauchst, aber vielleicht kann man sich dafür auch irgendwas konstruieren. Wenn du irgendwie nur nach ein paar Charts suchst, um das zu erläutern wie das funktioniert, dann findest du bestimmt auch was im Internet oder wir könnten schnell irgendwas basteln.

Interviewer: Okay, super. Dann würde ich eventuell nochmal auf dich zukommen, wenn ich genauer weiß, was ich brauche.

Agile Coach: Du kannst mal bei ActionableAgil gucken. Das ist das Tool von dem ich gesprochen habe, da kannst du dir eine Testlizenz machen und dir die ganzen Charts angucken. Das Tool ist quasi von dem Erfinder, naja oder dem, der das im Wesentlichen propagiert diese Flow-Metriken. Das ist so einer dieser Kanban-Gurus. Daniel Vacanti heißt der.

Interviewer: Okay, super. Weil das habe ich bei dir jetzt rausgehört, dass diese Flow-Metriken das Entscheidende waren, was sich durch Scrum mit Kanban geändert hat.

Agile Coach: Genau, das ist halt das A und O. Ja, genau. Auch viele sagen ja auch, sie machen Kanban und das einzige was sie haben ist das Kanban-Board. Und das hat dann genauso wenig mit Kanban zu tun, wie andere vielleicht sagen sie machen Scrum und es hat aber auch nicht viel mit Scrum zu tun.

Interviewer: Okay gut. Dann bin ich durch mit meinen Fragen. Vielen lieben Dank dir!

A4: Transkript des Experteninterviews mit dem Entwickler von [redacted]

Interview geführt am 14. Mai 2020 von 10:45 bis 11:25 Uhr

Interviewer: Okay, dann können wir ja gleich starten. Wie lange bist du denn eigentlich schon dabei?

Entwickler: Also in der Firma bin ich seit 15 Jahren. Anwalt24 mache ich glaube seit 7 oder 8 Jahren.

Interviewer: Also fast seit Beginn?

Entwickler: Ja, das ist ziemlich alt das Ding – also das ist 13 oder 14 Jahre alt und ich bin so in der Mitte eingestiegen.

Interviewer: Habt ihr denn von Anfang an Scrum gemacht?

Entwickler: Nein, eigentlich nie. Das war von den ganzen Anfängen, wo ich angefangen habe, war es Excel-Listen abarbeiten. Dann kam irgendwann Jira. Dann waren die Listen in Jira. Dann hat man die abgearbeitet. Ja in Prinzip hat man sich selber dann irgendwann ein Trello-Board gemacht, um selber seine Arbeit zu verfolgen und zu sehen was man gemacht hat. Und erst vor 3 Jahren, vermute ich, fing das Thema Scrum an. Da hatten wir auch Scrumban, Kanban, und dann wurde es mal wieder Scrum. Da war das Teamsetting halt auch ein bisschen anders. Da bestand das Team aus zwei Entwicklern und einem Product Manager, also sehr klein. Dann sollte es ein Relaunch geben. Dann merkte man so die Problemchen, was wollte überhaupt der Product Manager, was sollen wir umsetzen? Das lief nicht so richtig rund. Ja in der Analyse kam dann raus, dass man auf alle Fälle den Prozess verbessern muss. Könnte aber auch an der Menge der Leute gelegen haben. Und dann fing das halt an mit Scrum, und dann immer mehr Scrum. Und im Prinzip haben sich ja Alexander als erstes, dann Adrienn und dann Alexander, Adrienn und Andreas bei anwalt24 ausgetobt, so ungefähr. Dann hatte man irgendwann Scrum in Reinform. Das so der historische Punkt dabei.

Interviewer: Und wie würdest du jetzt aus Entwicklersicht diese Umstellung auf Scrum beschreiben? Fandst du es am Ende dann viel besser und die Arbeit hat am Ende viel besser funktioniert? Oder war das Quatsch mit den ganzen Meetings und ihr habt das vorher ja auch alles hinbekommen?

Entwickler: Geteilt. Also letztendlich finde ich es immer gut, wenn man, egal was man da macht, schaut welches Problem man tatsächlich hat. Und gerade beim Relaunch war unser Problem, dass wir keinen konkreten Entwicklungsplan hatten. Wir hatten sehr abstrakte Tickets. Das führte dann dazu, dass wir auch nur sehr abstrakt daran arbeiten konnten und auch nicht z.B. das so umgesetzt haben, wie der Product Manager das wollte, weil es aus dem Ticket nicht hervorging. Das war z.B. eines der zentralen Probleme für einen Entwickler. Die Ticketqualität ist ja relativ prozessunabhängig. Und die musste man halt optimieren. Wenn man dann eine gute Qualität darin hat, sieht das für einen Entwickler so aus, dass man ein Ticket nimmt und das über das Board schubst und irgendwann das dann fertig ist. Was bei Scrum ja dann noch hinzukommt ist, dass man mehr formale Meetings hat, um dann halt auch Refinement-Sachen – das war halt relativ entscheidend, um halt überhaupt mal die Ticketqualität zu verbessern. Früher in diesem kleinen Team gab es quasi keine Refinement-Sessions. Da hat der Product Manager für sich Tickets geschrieben und die dann bei uns aufs Board gekippt. Ob wir es verstanden haben oder nicht, sei mal dahingestellt oder ob das überhaupt Sinn ergibt, was er sich da gedacht hat. Dieses Refinement ist tatsächlich auch nochmal wichtiger geworden. Dann kam halt noch hinzu, dass wir mit einem internationalen Team noch größer wurden. Dann kamen noch drei Leute aus Armenien hinzu. Dann hatte man die Kommunikation auf Englisch. Dann musste man schauen, dass die das auch verstehen, um was es geht. Ich behaupte jetzt mal, wenn jemand inhouse ist, so wie ich und dann auch schon lange im Unternehmen ist, dann versteht der manche Sachen besser als jemand anderes. Das war eine neue Herausforderung. Damit der Prozess auch eingehalten wird, ist es entscheidend, dass die Tickets eine vernünftige Größe haben. Ganz am Anfang hatten wir auch Tickets…naja…hätte ungefähr drei Monate gedauert. Und verteilt über mehrere Leute kriegst du es auch nicht. Und nach vier Monaten siehst du dann, oh, es ist doch noch nicht fertig. Also kam dann – nachdem man die Ticketqualität verbessert hatte – zu sehen, was überhaupt…ob man die aufsplitten kann, ob es auch Unterpunkte gibt, die man losgelöst entwickeln kann. Das war dann halt auch ein entscheidender Punkt. Das hat aber ziemlich lange gedauert bis das im Verständnis von allen angekommen war, weil das natürlich immer nur vom Endergebnis gesehen wurde. Also was will ich haben, wenn dann alles fertig ist, damit die neue Funktionalität dann da ist. Aber es wurde nicht gesehen, dass man das auch in Teilpunkten entwickeln kann. Und klar, wenn man diese Teilpunkte dann hinkriegt, kann man unterschiedliche Aufgabe auf unterschiedliche Leute verteilen, parallel bearbeiten. Die Tickets haben dann auch irgendwann eine Größe bekommen, die leichter einschätzbar war, sodass man dann auch mal sicher sagen konnte, dass das im Sprint gemacht wurde. Das hat dann auch irgendwann funktioniert. Ja dadurch hatte man dann zu mindestens nach jedem Sprint die nächste Iteration seines Produktes. Man konnte den Fortschritt dann besser nachhalten oder planen. Gerade dieses „Ticket aufteilen" war ein sehr mühseliger Stepp.

Interviewer: Okay. Und weil du bei den Tickets meintest, dass die dann besser einschätzbar waren. Ihr habt ja am Anfang mit Story-Punkten geschätzt und zum Schluss dann wieder den Aufwand geschätzt mit 1,2,3. Magst du da nochmal was zu sagen? Was findest du besser? Wo siehst du die Vor- und Nachteile?

Entwickler: Ja, also Story-Punkte schätzen, das klingt zwar immer nice und dann ist da noch dieses Pokern dabei, das ist ein bisschen Fun-Faktor. Also wenn man gute Tickets hat, weiß man ungefähr was das ist und wie lange das dauert. Und 1,2,3 war dann nicht Schwierigkeit schätzen, sondern einfach Länge schätzen und dann wusste man auch, wie viele Tickets man dann in einen Sprint reinkriegt. Dadurch das die Ticketqualität dann gestiegen war, war es auch besser einschätzbar, das werde ich wohl in einer Woche schaffen. Im Prinzip, wenn man Scrum Poker spielt und da hohe Levels rauskommen, und die Komplexität zu hoch ist, kann das sowieso keiner einschätzen. Dann kannst du noch über die Gruppenstärke einen Median errechnen, wo der wohl liegen würde. Bei wenigen Leuten, naja, kommst du auch beim Median wirklich klar. Dieses Gespiele, okay, der, der am höchsten und niedrigsten war, sagen jetzt was dazu, dann hat man endlose Diskussionen und es bringt keinen wirklich weiter. Das dauert auch in meinen Augen zu lange, wenn man da ewige Diskussionen hat. In meinen Augen ist es halt wichtiger, dass man diese Probleme früh klärt, und zwar im Refinement. Bei der Ticketerstellung, die Größe, was gehört dazu, so gestaltet, dass es überschaubar ist. Und wenn es der Product Manager nicht überschauen kann, wird es ein Techniker auch nicht überschauen. Also im Prinzip, dass man die Probleme so früh wie möglich sieht und erkennt und löst und dann nicht erst in einem Sprint Planning. Also da kommt für mich das Sprint Planning einfach zu spät. Das ist früher schon offensichtlich, wo das Problem sein wird und das muss im Refinement gelöst werden. Und wenn das im Refinement gelöst ist, dann kann man mit einer sehr einfachen – kannst du dir ja vorstellen, 1,2,3, - da sind sich alle sehr schnell einig, du hast keine Diskussion und wenn du es gut gemacht hast – wir hatten ja drei Wochen Sprints, also 1 heißt 1 Woche usw. und bei einigen Tickets hast du das dann auch in einer Woche gelöst. Das war dann auch die Summe. Das ist gut bearbeitbar, gut planbar, gut verteilbar, gut verschiebbar – damit kannst du halt gut hantieren mit der Stückgröße. Und wenn die Stückgröße zu groß ist, hast du immer wieder ein Problem. Da hast du nachher im Sprint ein Problem, wenn es zu groß ist und von daher ist das für mich so das A und O, wo wir dann auch echt besser geworden sind. Wenn du die Statistiken vom Agile Coach gesehen hast, man auch irgendwann sehen konnte.

Interviewer: Okay, also du würdest sagen schätzen ist schon wichtig, aber es reicht so eine einfachere Methode der Aufwandschätzung, damit es zu keinen größeren Diskussionen kommt?

Entwickler: Ja, weil wenn dieses Schätzen sehr lange dauert, weil da komplexe Sachen dabei sind, dann hat man das Problem woanders. Es darf eigentlich nicht dazu kommen, dass das so kontrovers wird. Diese Sachen muss man halt vorher auflösen. Dazu kommt man dann eigentlich auch irgendwann, oder bis man zu mindestens hinarbeiten. Diese langen Diskussionen im Sprint Planning nützen eigentlich niemandem was. Da schreibt man lieber im Refinement einen Satz mehr oder splittet das Ticket auf, da hat man mehr gewonnen.

Interviewer: Okay, alles klar. Dann nochmal eine Frage zu dem Thema WiP-Limits. Ich habe schon gehört, dass es bei euch da viele Diskussionen gab, weil ihr eingeführt hattet, dass nur zwei Tickets pro Entwickler bearbeitet werden dürfen. Magst du deine Sichtweise zu dem Thema nochmal deutlich machen?

Entwickler: Also, in meinen Augen, ist der Prozess dazu da, dass man Tickets möglichst schnell von links nach rechts auf dem Board kriegt, dass man die Sachen im Sprint umsetzt. Letztendlich hast du dann…also wir hatten oft Tickets mit „external dependency“ gehabt. Das heißt dann einfach, das Ticket ist bis zu irgendeinem Status gekommen und dann ist es da. Und dann kannst du warten. Dann kannst du das Ticket wieder aus dem WiP-Limit nehmen und dann führst du es auch alles irgendwie ad absurdum. Du hast auch wieder viele Diskussionen und der Benefit ist, in meinen Augen, gleich null. Ziel jedes Entwicklers ist es, das Ticket von links nach rechts zu bekommen. Und in meinen Augen, ist es auch eher notwendig, dass z.B. dann…was bei uns im Prinzip ja auch jedem klar sein musste – wir haben dann auch im Refinement wieder gesagt, okay, das Ticket hat ein „external Dependency“, das wird also nicht einfach über das Board laufen, das wird irgendwo stecken bleiben. Das wissen wir schon vorher, das wissen wir auch schon Monate vorher. Dann haben wir halt irgendwann diese Rubrik „external Dependency“ eingeführt. Wenn du jetzt noch die Velocity misst, Tickets die jetzt da ewig rumliegen, da ist für mich ein Daily wichtig. Da muss man dann halt auch mal fragen. Scrum ersetzt halt keine Kommunikation. Man muss immer noch kommunizieren. Und dann erklärt sich das eigentlich auch. Und ein WiP-Limit hat von daher…es ist eine Regel, aber es führt nicht zu einer besseren Kommunikation. Also einerseits muss man halt schauen, dass Tickets, wo man schon weiß, dass sie andere Abhängigkeiten haben, klar analysieren, dass da Abhängigkeiten vorhanden sind, die vorher gelöst sein müssen oder erst gelöst werden, wenn ein Teil des Tickets gemacht wurde. Dann ist halt die Frage, wieso splitte ich das Ticket

nicht auf, bis zu dem Punkt wo die „external Dependency“ greift. Das muss man dann halt auch verstehen, dann versteht man auch eher, wie der Arbeitsablauf ist. Aber das WiP-Limit an sich, hat überhaupt nichts verändert oder verbessert. Also das ist einfach nur - gut, man hat es halt, dann redet man darüber, aber man redet nicht mehr über das Ticket an sich und das ist halt dann ein bisschen schade.

Interviewer: Okay, eigentlich soll das WiP-Limit ja helfen, dass man weniger Tickets anfängt, die man nicht beendet und schneller Tickets von links nach rechts bekommt. Aber was ich jetzt bei dir rausgehört habe ist, dass man das bei manchen Tickets nicht ändern kann und ihr habt das quasi sowieso im Kopf, dass ihr die Tickets schnellstmöglich beendet.

Entwickler: Das ist ja grundsätzlich das Ziel eines Entwicklers. Man legt die ja nicht gerne auf Fall. Also z.B. so ein Ticket mit „external Dependency“, da hörst du dann halt irgendwann auf und dann geht es irgendwann weiter, mitunter Monate später, das ist natürlich nicht so geil. Das ist dann halt auch weiterhin vorgekommen, weil das halt unumgänglich ist. Du hast halt keinen Einfluss auf die anderen Abteilungen und die haben halt auch ganz andere Prioritäten. Und dann musst du halt schauen, wie du damit umgehst. Wenn man dann besser drüber redet, kann man halt ein Ticket nehmen, was genau bis zum Anfang der anderen Abteilung geht und dann hat es halt ein Follow-Up-Ticket, was nachdem die andere Abteilung was gemacht hat, fertig wird. Und dann sind deine Tickets, mit denen du was getan hast, ebenfalls aber auch abarbeitbar. Und das ist dann hilfreich. In meinen Augen auch hilfreich für das Produktmanagement, weil man dann natürlich die Rückmeldung hat, okay, da muss ich mit der anderen Abteilung sprechen, da hängt es da und warum hängt es da. Das kann ja dann auch der Scrum Master machen, schauen wo die Abhängigkeiten überhaupt sind und dafür sorgen, dass es woanders dann mal vorwärtsgeht. Das ist ja keine zwingend technische Aufgabe zum Beispiel. Und von daher muss man immer schauen, welches Problem habe ich tatsächlich und dann sollte man das so früh wie irgend möglich lösen und nicht immer mehr Regeln und Regeln einbauen. Sondern schauen, was ist die Ursache dieser Probleme. Und gerade da, hilft ein WiP-Limit nicht wirklich. Dann wird immer so getan, dass es Entwicklern helfen soll. Man hat auch noch Dailys, da sollte man ja eigentlich die Kommunikation haben. Wenn man das zum Bespiel hat, dass sich ein Entwickler zehn Tickets nimmt und alle „In Progress“ setzt, dann kann man auch mal ein Wort sagen. Scrum sollte nicht Kommunikation ersetzen. Du siehst jeden Tag dein Board, du siehst jeden Tag was da passiert und du hast jeden Tag ein Daily, wo du über solche Sachen reden kannst. Das sollte man halt einfach nutzen. Wenn man das dann halt als Product Manager nicht versteht, wieso der Entwickler sich 10 Tickets genommen hat - okay, versteht normalerweise niemand – dann fragt man halt mal. Das ist ja auffällig, im Board siehst du, das eine Person, mit einem Blick siehst du, dass eine Person 10 Tickets hat. Und wenn man darüber gesprochen hat, also entweder hat der Produkt Manager Klarheit gewonnen oder der Entwickler hat gesehen, ja, das war vielleicht keine clevere Idee und dann macht er das normalerweise auch nicht wieder, aber da brauch ich nicht großartig viele Regeln für.

Interviewer: Okay, dann zum Daily. Der agile Coach hatte mir auch erzählt, dass als ihr Scrum mit Kanban gemacht habt und er die Flow-Metriken mit eingeführt hat (Cycle-Time, Throughput und solche Sachen), ihr im Daily nicht mehr diese drei Fragen abgearbeitet habt (Was habe ich gestern gemacht? Was mache ich heute? Was gibt es für Hindernisse?), sondern dass ihr auf diese Metriken geschaut habt. Fandst du das gut?

Entwickler: Das haben wir jetzt auch nicht jedes Mal gemacht. Letztendlich haben wir in die Metriken größtenteils in der Retrospektive glaube ich reingeguckt. Also nach dem Sprint haben wir analysiert wie lief eigentlich der Sprint. Wir haben nicht täglich analysiert was passiert. Wir haben uns das Board angeguckt und wenn alles normal läuft, erkennst du auf dem Board was passiert. Diese drei Fragen halte ich auch für totalen Blödsinn. Wie gesagt, du hast das Board vor dir. Da hast du verschiedene Steps – wir hatten Verification Step, Tests mussten laufen oder sowas in der Art – naja und das sollte die tägliche Arbeit sein. Man verfolgt also, was passiert auf dem Board. Man kann ja auch alles tracken, wo war das Ticket vorher, wann wurde es verschoben und dann sehe ich eigentlich okay das liegt da jetzt und wartet auf eine Verifikation vom Produkt Manager. Okay warum wartet das jetzt? Okay, weil der Produkt Manager auf einer Messe war oder so. Dann kann man darüber reden. Gibt es da Ersatz? Kann jemand anderes das abnehmen oder soll es weiter liegen? Also aus dem Board kann man eigentlich alle Probleme ablesen. Das man dann konkret darüber spricht und die konkreten Probleme angeht. Oder z.B. es sind zu viele Tickets von einer Person auf dem Board. Warum ist das so? Ja, dann stellt sich raus, ich habe fünf Tickets auf, weil drei darauf warten vom Produkt Manager abgenommen zu werden, weil er auf einer Messe ist. Dann redet man darüber. Okay und weil er morgen wiederkommt, ist das kein Problem und dann ist das Thema durch. Das man dann auch sieht, wo tatsächlich Probleme sind. Oder dann siehst du z.B. das ein Ticket lange steht, weil die Tests nicht mehr laufen, weil die Tests umgeschrieben werden müssen. Dadurch ist ein Arbeitsaufwand entstanden, der nicht vorher abgeschätzt wurde. Da

haben wir dann zum Beispiel Chap Cards eingeführt. Dann konnte man halt sehen, ah, da ist noch Zusatzaufwand entstanden, der vorher nicht gesehen wurde und dieses Ticket kann halt erst fertig gestellt werden, wenn diese Tests gefixt wurden. Das sind natürlich dann auch so Sachen, wo man immer besser wird, wenn man Tickets refined. Hat das eine Abhängigkeit zu Tests? Muss man da Tests umschreiben? Die Erfahrungswerte sagen einem das dann schon irgendwann. Und was du dann letztendlich hast ist, auf dem Board alle wesentlichen Sachen hast und auf dem Board letztendlich auch ergründen kannst, wieso was wie ist.

Interviewer: Und das ist dann auch die bessere Fokussierung als mit den drei Fragen?

Entwickler: Ja auf dem Board kann jeder jederzeit drauf gucken. Die Fragen beantwortest du einmal am Tag. Und die Frage was hast du gestern gemacht, ja, das siehst du auf dem Board was ich gestern gemacht habe. Die Tickets habe ich verschoben. Die sinnvollere Frage ist dann halt, du hast momentan gar kein Ticket, hast du vergessen dein Ticket zu verschieben? Ja, stimmt, ich habe vergessen mein Ticket zu verschieben und schon ist Bewegung drin. Das führte dann auch dazu, dass…wie schon gesagt, wir waren verteilt, dann noch die Zeitzonen ein bisschen unterschiedlich und „external Dependencies" die von Weißrussland bis nach Südamerika reichten. Da hat man dann Kommunikation manchmal auch am besten in den Tickets drin. Gerade wenn es zum Beispiel bei einer Verifikation um eine Nachbesserung geht, dass nicht nur mit der einzelnen Person gesprochen wird, sondern man einen Kommentar in das Ticket schreibt. Das ist offen und jeder der eine Frage hat oder irgendwas von dem Prozess nicht versteht, kann sich dann das Ticket angucken. Da ist die Besonderheit bei uns gewesen, dass wir die Tickets nicht zurückschieben durften. Das fand ich eigentlich gar nicht so schlecht. Also am Anfang ein bisschen irritierend. Es hat dann aber geholfen, als die Kommentare in den Tickets standen. Ohne Kommentare sieht man halt nicht, wieso das Ticket immer da steht und nicht weitergeht. Erst nach dem wir die Tickets geschrieben haben konnte man halt sehen, okay das muss noch nachgebessert werden, deswegen steht das da. Die Wichtigkeit des Boards ist sehr entscheidend, das alle Informationen auf dem Board verfügbar sind. Dann kann jeder jederzeit rauskriegen. Und wenn er es nicht rauskriegt, hat man noch das Daily um zu verstehen warum Sachen anders sind, als man das erwartet hat. Das hat dann eigentlich auch immer besser funktioniert. Gerade diese Kommentare waren bei der Historie wichtig und dann sieht man auch als Entwickler, das z.B. die Ticketnummer nutzt man ja auch einfach um das im Repository abzulegen und wenn man dann so eine Codeänderung anguckt oder so ein Blame über seinen Code macht, sieht man halt bei der Änderung wurde das gemacht an dem Punkt. Dann gucke ich wieder in das Ticket rein und sehe im Prinzip die ganze Historie, wie ist es dazu gekommen, warum sind da Änderungen drin und was natürlich auch mal passiert, dass man beim Bearbeiten des Tickets, der Produkt Manager sieht, die Kleinigkeit, die müsste da jetzt eigentlich auch noch mit drin sein. Die taucht bisher nicht in den Akzeptanzkriterien auf, aber bei der Verifikation habe ich gesehen, es ist zwingend notwendig, dass da jetzt was kommt. Dann kann man halt auch beim Daily drüber reden, wir ändern da nochmal die Akzeptanzkriterien. Dauert das jetzt zwei Wochen deswegen? Oder schaffen wir das noch unterzubringen? Muss ein neues Ticket aufgemacht werden oder was auch immer. Dann kann man halt über das Problem und alle möglichen Lösungen im Daily sprechen. Das sind eigentlich so allgemeine Themen, die relativ Scrum unabhängig sind, die aber relativ wichtig sind, das Scrum funktioniert. Diese Kommunikation, wo ist überhaupt das Problem und wie sieht die Problemlösung aus und wie kann ich das so früh wie möglich lösen.

Interviewer: Das stimmt. Okay, dann noch eine abschließende Frage. Ihr habt ja auch immer ein Sprintziel festgelegt. Was hältst du so allgemein von einem Sprintziel? Findest du das wichtig als Marschrichtung das zu haben?

Entwickler: Also aus technischer Sicht, ist es nicht entscheidend. Weil ich mache das Ticket, und jedes Ticket besteht bestenfalls für sich und damit ist es für mich auch bearbeitbar. Das reicht mir im Prinzip. Aber ich finde es letztendlich gut, wenn das Produktmanagement sowas macht, um sich selber klar zu machen, wo gehe ich mit dem Produkt gerade hin. Geht es darum, neue Features einzubauen oder worum geht es in diesem Sprint? Was mache ich? Will ich etwas verbessern, zum Beispiel für eine einzelne Personengruppe? Wie man das Produkt gestaltet, spiegelt sich dann auch im Sprint wieder. Lustig zu sehen ist dann nur, wenn man die Meinung hat, ich will jetzt was für den Customer Service verbessern. Und dann guckst du in den Sprint und entdeckst kein Ticket, was etwas für den Customer Service verbessert. Dann ist halt die Frage, sollten wir nicht etwas Anderes machen, um den Customer Service zu helfen, wenn wir sehen, dass der schlecht arbeiten kann? Dann sollten wir andere Tickets machen. Wenn der Produkt Manager auch im Scrum-Prozess aktiv ist, mitunter ja auch wie er zu seinen Tickets kommt, da kann er halt auch das gestalten und erkennt Probleme und muss sehen, dass er diese Probleme auch gelöst bekommt. Am besten schaut er dann, wie er diese Probleme clustern kann. Da gibt es ja auch wieder Möglichkeiten, wie man das im Board visualisieren kann. Das man Komponenten nutzt oder Epics, um zusammenhängende Themen zu clustern. Wir haben z.B. auch unterschiedliche Komponenten genutzt, da hatten wir dann auch den Customer Service als

Komponente. Danach kann man ja dann auch filtern, um zu sehen wie viele Tickets sind im Sprint. So aus Produktmanagersicht kann ich da auf jeden Fall sinnvolle Sachen sehen, aber technisch ist das nicht so entscheidend.

Interviewer: Okay, dann bin ich durch mit meinen Fragen. Vielen lieben dank dir auf jeden Fall für den Einblick. (…)

Entwickler: (…) Ja das einzige was man nicht vergessen darf ist halt, dass Scrum die Kommunikation nicht ersetzt. Es gibt eine gewisse Form vor, wie dass dann stattzufinden hat, es formalisiert Sachen aber man muss weiterhin miteinander reden. Das ist halt ein entscheidender Punkt. Und ist mitunter bei vielen Problemen auch schon vorher der Fall gewesen. Mit einer Excel Liste wirst du vermutlich auch arbeiten können, wenn du jeden Tag mit deinem Produkt Manager kommunizierst oder er neben dir sitzt. Außer du hast dann riesige Teams. Du musst halt immer gucken, welche Abhängigkeiten du hast. Dann funktioniert das halt nicht. Also man versucht, das war so meine Erfahrung, dass was eben in den Lehrbüchern ist über ein Projekt zu kippen und man sieht gar nicht die Projektunterschiede. Teamgröße ist entscheidend, was habe ich für ein Produkt, hat das schnelle oder langsame Iterationen? Es gibt auch Sachen, die in der Entwicklung sehr langsam sind, wo man auch langsam nur Entwicklungen ausrollte, weil sie mitunter ganz viele Abhängigkeiten haben. Wenn du zentrale Systeme hast, die laufen ganz anders ab als visuelle Sachen auf einer Web-Oberfläche. Und genau diese Sachen muss man halt berücksichtigen. Schön ist es natürlich, aus Entwicklersicht, wenn man sich vorher mehr Gedanken drüber macht, wie sind die Gegebenheiten hier, bevor man erstmal alles drüber kippt und dann kriegst du so einen Schock. Ich habe das jetzt zwar eingeführt, aber alle Probleme die ich vorher hatte, sind weiterhin vorhanden. Es hat nichts davon gelöst und das ist dann halt frustrierend an dieser Stelle. Die Anpassung ist halt das A und O.

Interviewer: Ja klar, Scrum ist ja auch kein Wundermittel, was dann plötzlich alles besser macht.

Entwickler: Wie gesagt, meistens sind es Menschen, die miteinander arbeiten und Menschen müssen miteinander reden und die Informationen müssen irgendwo geteilt werden und dafür hat man meistens ein Board. Das muss man machen, das ist halt Arbeit. Scrum hilft nur die Formen vorzugeben, dass das auch stattfindet. Einen Termin zu setzen für das Refinement oder so. Was man in diesem Termin macht, macht Scrum nicht.

Literaturverzeichnis

Anderson, David J.; Carmichael, Andy (2018): Die Essenz von Kanban kompakt. Heidelberg. dpunkt.verlag

Bartonitz, Martin; Lévesque, Veronika; Michl, Thomas; Steinbrecher, Wolf; Vonhof, Cornelia; Wagner, Ludger (2018): Agile Verwaltung. Berlin, Heidelberg. Springer Berlin Heidelberg

Boes, Andreas; Langes, Barbara; Kämpf, Tobias (2018): »Lean« und »agil« im Büro. Neue Organisationskonzepte in der digitalen Transformation und ihre Folgen für die Angestellten. Bielefeld. transcript-Verlag

Bogner, Alexander; Littig, Beate; Menz, Wolfgang (2014): Interviews mit Experten: Eine praxisorientierte Einführung. Wiesbaden. Springer VS

Böhm, Janko (2019): Erfolgsfaktor Agilität. Warum Scrum und Kanban zu zufriedenen Mitarbeitern und erfolgreichen Kunden führen. Stuttgart. Springer Fachmedien Wiesbaden GmbH

Dechange, André (2020): Projektmanagement – Schnell erfasst. Dortmund. Springer-Verlag GmbH Deutschland

Drähter, Rolf; Koschek, Holger; Sahling, Carsten (2019): Scrum – kurz & gut. O´Reilly Taschenbuchbibliothek

Epping, Thomas (2011): Kanban für die Softwareentwicklung. Heidelberg. Springer-Verlag Berlin Heidelberg

Fritz, Patrick (2017): Kanban vs. Scrum im Vergleich, in: https://www.fritz.tips/scrum-vs-kanban-im-vergleich/, Abruf: 07.05.2020

Frömling, Sabine (2020): Was Scrum von Kanban unterscheidet, in: https://www.computerwoche.de/a/was-scrum-von-kanban-unterscheidet,3548315, Abruf: 25.05.2020

Henkel, Paul (2018): Schnelleinstieg: Kanban im Projektmanagement, in: https://www.projectwizards.net/de/blog/2018/03/kanban#proscons, Abruf: 25.05.2020

Itemis AG (2016): Eigenschaften des Product Backlog, in:

https://blogs.itemis.com/de/scrum-eigenschaften-des-product-backlog, Abruf: 25.05.2020

Kanbanize (o.D.): Kanban-Analysen, in: https://kanbanize.com/de/kanban ressourcen/kanbananalysen, Abruf: 13.05.2020

Kanbanize (o.D.): 6 Agile Metrics to Boost Your Project Management, in: https://kanbanize.com/agile/project-management/agile-metrics, Abruf: 08.06.2020

Kanbanize (2017): Throughput-Run-Chart, in: https://kanbanize.com/blog/throughput-run-chart/, Abruf: 08.06.2020

Kiel, Christopher (2015): Scrum vs. Kanban: Managementmethoden im Vergleich, in: https://digitaler-mittelstand.de/trends/ratgeber/scrum-vs-kanban-managementmethoden-im-vergleich-7668, Abruf: 06.05.2020

Kusay-Merkle, Ursula (2018): Agiles Projektmanagement im Berufsalltag. Für mittlere und kleine Projekte. Wiesbaden. Springer-Verlag GmbH Deutschland

Linz, Tilo (2013): Testen in Scrum-Projekten. Leitfaden für Softwarequalität in der agilen Welt. dpunkt.verlag

Maximini, Dominik (2018): Scrum – Einführung in der Unternehmenspraxis. Berlin, Heidelberg. Springer Berlin Heidelberg.

Neng, Alexander (2018): Scrum und Kanban – wann nutze ich was?, in: https://www.methodpark.de/blog/scrum-und-kanban-wann-nutze-ich-was/, Abruf: 25.05.2020

Novkov, Alex (2019): Kanban vs. Scrum: What Are the Differences Between Scrum and Kanban, in: https://kanbanize.com/blog/kanban-vs-scrum-infographic/, Abruf: 25.05.2020

Pfeffer, Joachim (2020): Produkt-Entwicklung. Lean & Agile. München. Carl Hanser Verlag München

Preußig, Jörg (2018): Agiles Projektmanagement. Agilität und Scrum im klassischen Projektumfeld. Freiburg, München, Stuttgart. Haufe Gruppe

Rehkopf, Max (o.D.): Was ist ein Kanban Board?, in: https://www.atlassian.com/de/agile/kanban/boards, Abruf: 25.05.2020

Scaled Agile (2020): Leading SAFe®. Thriving in the digital age with Business Agility 5.0

Schwaber, Ken (2007): Agiles Projektmanagement mit Scrum. Unterschleissheim. Microsoft Press

Schwaber, Ken; Sutherland, Jeff (2017): The Scrum Guide, in: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-German.pdf, Abruf: 25.05.2020

StepStone GmbH (2020): Agile Unternehmen, in: https://www.stepstone.de/wissen/agilesarbeiten/, Abruf: 09.04.2020

Vacanti, Daniel; Yeret, Yuval (2019): Der Kanban Guide für Scrum-Teams, in: https://scrumorg-website-prod.s3.amazonaws.com/drupal/2019-10/2019-09-Kanban-Guide-for-Scrum-Teams-German_0.pdf?nexus-file=https%3A%2F%2Fscrumorg-website-prod.s3.amazonaws.com%2Fdrupal%2F2019-10%2F2019-09-Kanban-Guide-for-Scrum-Teams-German_0.pdf, Abruf: 25.05.2020

Vallon, Raoul; Müller-Wernhart, Michael; Schramm, Wolfgang; Grechenig, Thomas (2014): Kombination von Agil und Lean in der Softwareentwicklung, in: Informatik Spektrum 01/2014, S.28-35

Virgenschow, Uwe; Grass, Andrea; Augstin, Alexander; Hofmann, Michael (2015): APM – Agiles Projektmanagement. Anspruchsvolle Softwareprojekte erfolgreich steuern. Heidelberg. dpunkt.verlag

Weller, Kay (o.D.): Kanban und Scrum im Vergleich, in: https://sitegeist agile.de/blog/agile-methoden/kanban-und-scrum-im-vergleich.html, Abruf: 07.05.2020

[redacted] (2020): Portfolio für Anwälte. [redacted], in: [redacted], Abruf: 02.06.2020

[redacted] (2020): Screenshot Sprint-Board in Jira, in: [redacted], Abruf: 17.04.2020 [vertraulich]

[redacted] (2020): Screenshot Streudiagramm in ActionableAgile, in: https://analytics.actionableagile.com/, Abruf: 24.04.2020 [vertraulich]

Ehrenwörtliche Erklärung

Ich versichere hiermit, dass ich meine Bachelorarbeit mit dem Thema

„Agiles Projektmanagement mit Scrum und Kanban – Ergänzung oder Konkurrenz?“

selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die Übernahme wörtlicher Zitate aus der Literatur sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe und dass ich meine Bachelorarbeit bei keiner anderen Prüfung vorgelegt habe.

Köln
Ort

28.06.2020

Datum Unterschrift