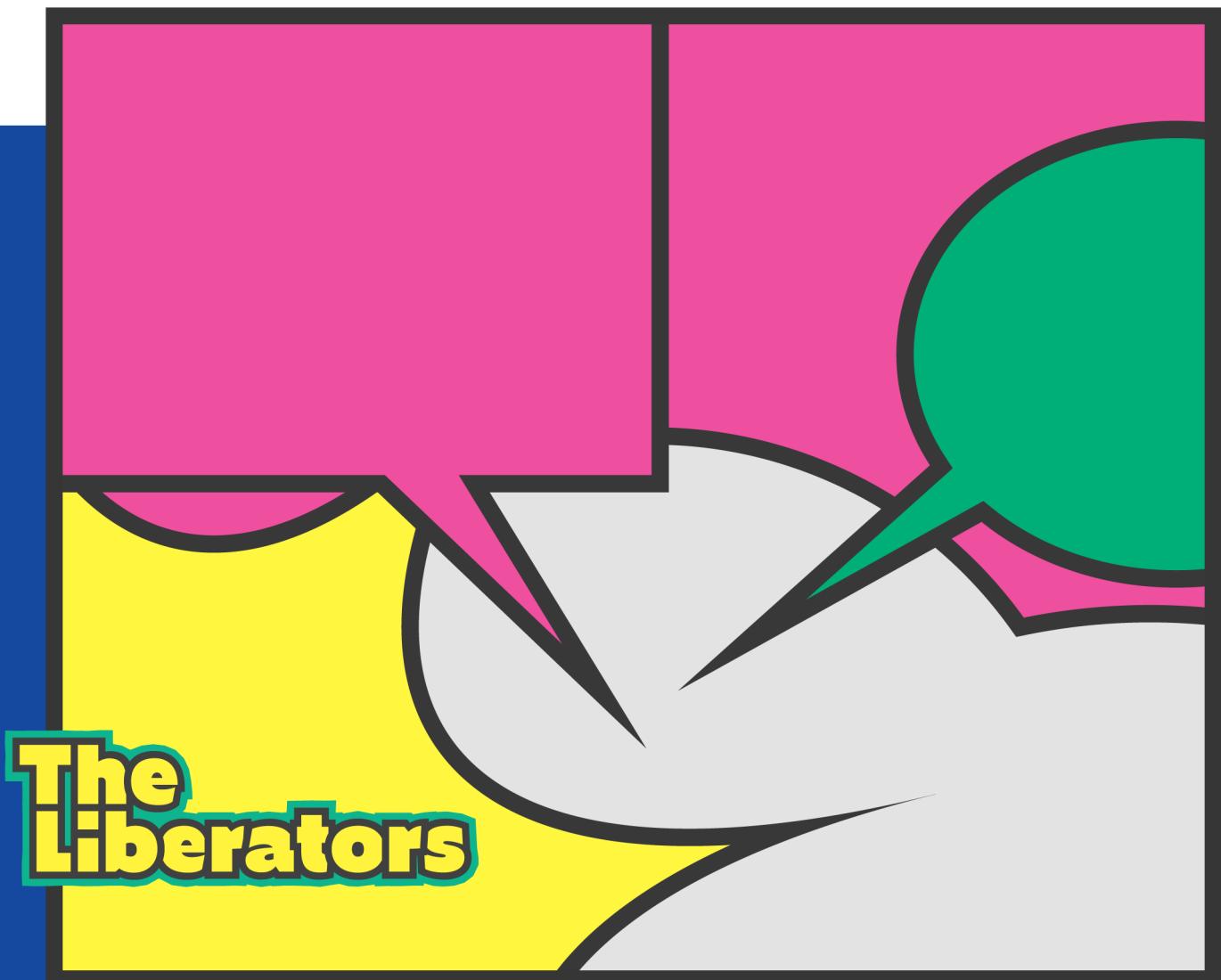


Scrum: Ein Rahmenwerk Für Risikoreduzierung Und Schnellere Wertschöpfung

Ein Überblick über das Scrum-Framework – für Scrum-Neulinge und all jene, die ihre Kenntnisse und ihr Verständnis von Scrum etwas auffrischen möchten.



In diesem Whitepaper



Vorwort

Ein wenig Geschichte

Die Säulen der Empirie

Scrum: Ein Rahmenwerk für empirische Prozesskontrolle

Die Scrum-Artefakte

Die Scrum Events

Die Verantwortlichkeiten [Accountabilities] in Scrum

Zwei zentrale Prinzipien

Fünf Schlüsselwerte für mehr Empirie

Gemeinsam: Ein einfaches Rahmenwerk

Vorwort



"Scrum ist uns egal", sagen wir den Leuten manchmal. Das verursacht sicherlich einige zweifelnde Blicke. Aber es ist unsere Art zu sagen, dass es uns nicht um das Scrum-Framework geht – sondern darum, was es ermöglicht. Wenn man diesen Blickwinkel einnimmt, werden viele theoretische Fragen selbsterklärend oder sogar sinnlos. Wie zum Beispiel die folgenden: "Sollten sich alle Einträge im Sprint Backlog auf das Sprintziel beziehen?", "Sollten auch Bugs ins Product Backlog aufgenommen werden?" oder "Sollten wir das Daily Scrum nach genau 15 Minuten stoppen?" Es ist leicht, sich in den Details zu verlieren, wenn man das Gesamtbild vergisst (oder nicht sieht).

Wir lieben das Scrum-Framework für das, was es ermöglicht. Der offizielle Scrum-Guide ist bereits hervorragend dazu geeignet, es kurz und bündig zu erklären. Dieses Dokument wollen wir um eine Erläuterung in unseren eigenen Worten ergänzen, entsprechend unserer Arbeit mit Scrum-Teams und als professionelle Scrum-Trainer für Scrum.org. Unser Ziel war es, praxisnah und bodenständig vor dem Hintergrund all dessen zu schreiben, was das Scrum-Framework ermöglicht. Wir hoffen, dass sich unser Text gut lesen lässt, mögliche Missverständnisse aufklärt und Ihr Verständnis von Scrum vertieft.

Wir haben dieses Whitepaper zusammen mit Johannes Schartau für unser Buch 'The Zombie Scrum Survival Guide' geschrieben. Diese Übersetzung in die deutsche Sprache wurde von Antje Lehmann-Benz erstellt.

Ein wenig Geschichte



Das [Scrum Framework](#) wurde von Ken Schwaber und Jeff Sutherland in den 1990er-Jahren entwickelt. Es wurde 1995 erstmals formalisiert, um der innewohnenden Komplexität in der Produkt- und Softwareentwicklung Rechnung zu tragen. In jüngerer Zeit wird das Scrum -Framework erfolgreich auf komplexe Probleme in einer Vielzahl von Bereichen angewendet: Vom Marketing bis zur organisatorischen Veränderung, und von der wissenschaftlichen Forschung bis zur Softwareentwicklung. Egal wo Sie es einsetzen, baut das Scrum-Framework auf drei Säulen auf, die eine empirische Prozesskontrolle ermöglichen.

Die Säulen der Empirie



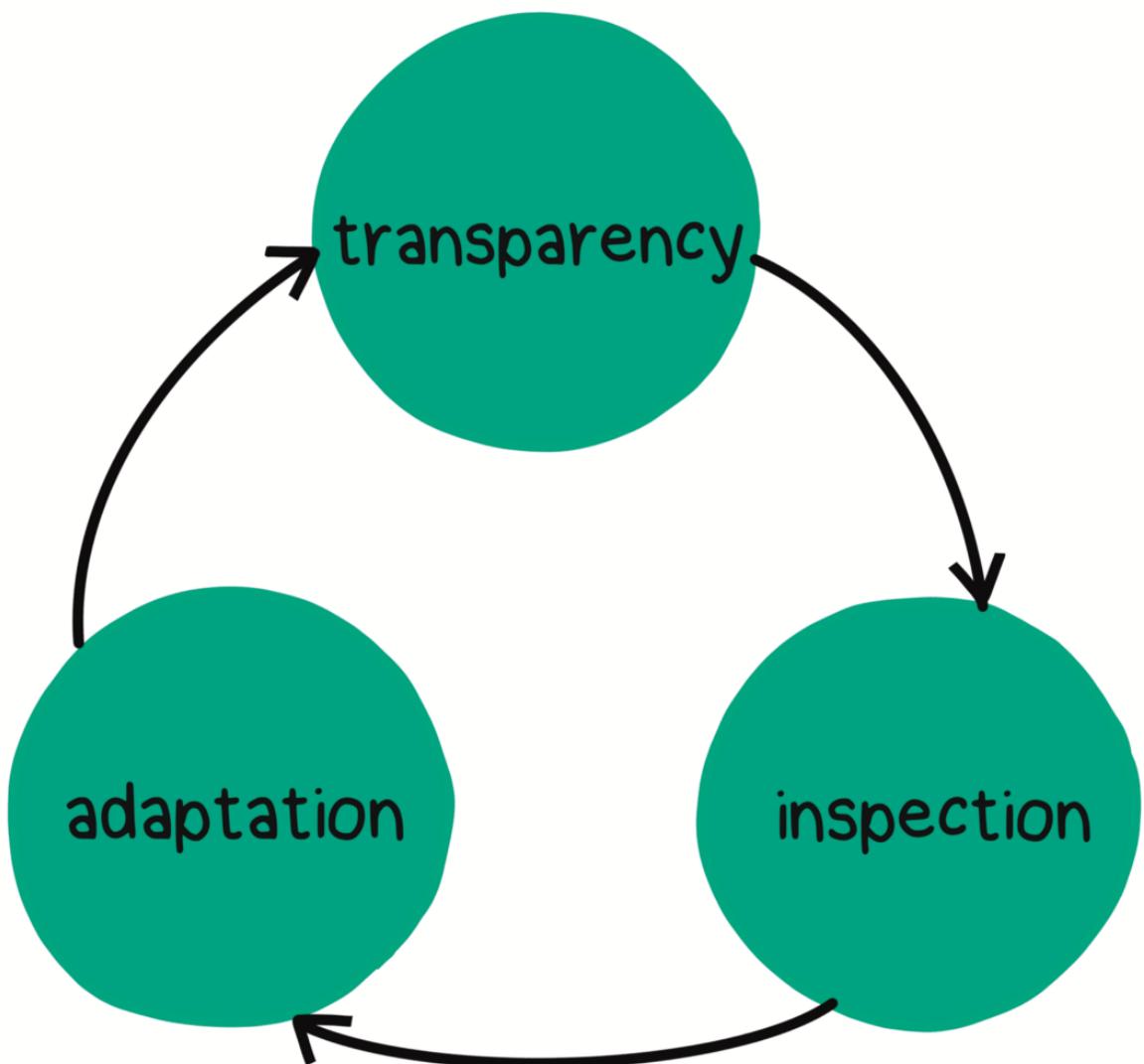
Das Scrum Framework baut auf drei Säulen auf, die eine empirische Prozesskontrolle ermöglichen:

- **Transparenz:** Sie sammeln Daten – wie Metriken, Feedback und andere Erfahrungen – um herauszufinden, was momentan geschieht;
- **Inspektion:** Sie untersuchen den Fortschritt mit allen Beteiligten und entscheiden, was das für Ihre Vorhaben bedeutet;
- **Anpassung:** Sie nehmen Veränderungen vor, von denen Sie hoffen, dass sie Sie der Verwirklichung Ihrer Vorhaben näher bringen;

Dieser Zyklus wird so oft wie nötig wiederholt, um Abweichungen, unerwartete Neuentdeckungen und potenzielle Chancen zu entdecken, die sich während der Arbeit ergeben. Dies geschieht nicht einmal im Jahr oder bei Abschluss des Projekts, sondern kontinuierlich auf täglicher, wöchentlicher oder monatlicher Basis. Anstatt Entscheidungen aufgrund von Annahmen über mögliche Zukunftsszenarien zu treffen, treffen Sie sie auf Grundlage der Daten, die Sie bis zu diesem Zeitpunkt gesammelt haben. Das ist Empirie. Und in diesem Beitrag werden Sie herausfinden, wie alles im Scrum-Framework um diese Säulen herum konzipiert ist.

"Anstatt Entscheidungen aufgrund von Annahmen über mögliche Zukunftsszenarien zu treffen, treffen Sie sie auf Grundlage der Daten, die Sie bis zu diesem Zeitpunkt gesammelt haben.



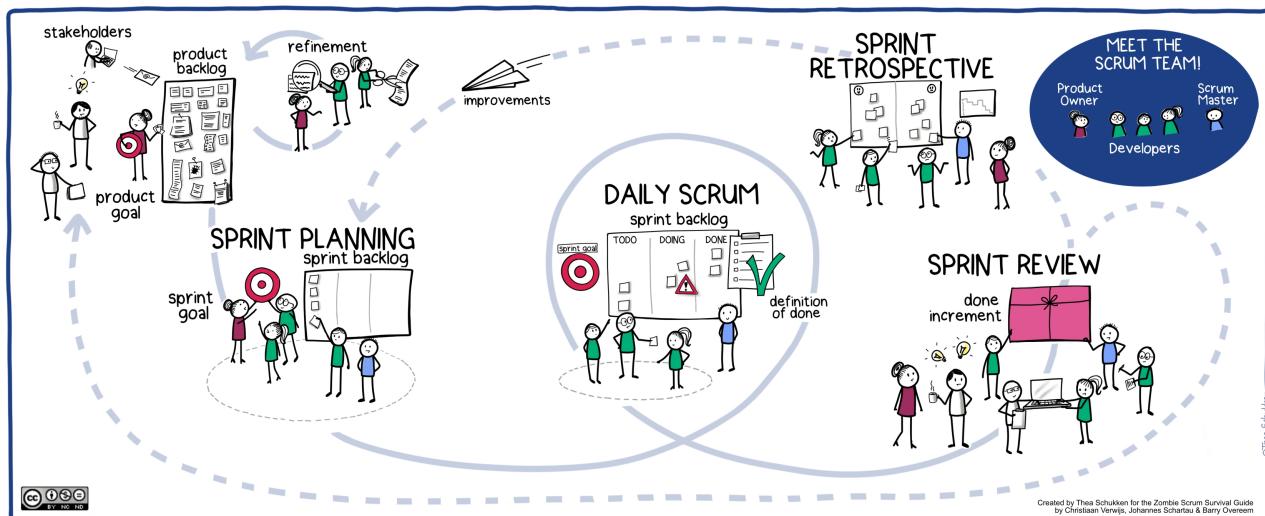


Created by Thea Schukken for the Zombie Scrum Survival Guide
by Christiaan Verwijs, Johannes Schartau & Barry Overeem



Scrum: Ein Rahmenwerk für empirische Prozesskontrolle

Dass Sie bei Ihrer Arbeit an einem Produkt Transparenz brauchen, dass Sie diese Arbeit häufig überprüfen müssen und dass die daraus resultierenden Erkenntnisse die Anpassung vorantreiben – all das ist immer nur die eine Seite. Diese Aussagen auf eine greifbare, konkrete Art und Weise in die Praxis umzusetzen, ist noch einmal etwas anderes. Genau das macht Scrum zu einem Rahmenwerk – es bietet fünf sich wiederholende Events, um an drei Artefakten zu arbeiten. Es gibt drei Rollen, die dies unterstützen, sowie mehrere Prinzipien und Regeln, um alles zu einem zusammenhängenden Ganzen zu machen.



Die Scrum-Artefakte

Die erste Säule der empirischen Prozesskontrolle ist die "Transparenz". Um häufig den Fortschritt Ihrer Arbeit an einem Produkt zu überprüfen und Entscheidungen darüber zu treffen, was (noch) benötigt wird, brauchen wir etwas, das wir überprüfen können. Wenn wir die Arbeit an einem Produkt "transparent" machen, meinen wir, dass wir es in einer Form zur Verfügung stellen, die es den an der Produktentwicklung Beteiligten ermöglicht, das Produkt zu betrachten, mit seiner Hilfe Annahmen zu validieren und neue Ideen daraus zu generieren.

Das Scrum Framework setzt von Teams voraus, dass sie mindestens drei Elemente ihrer Arbeit an einem Produkt transparent machen. Wir nennen diese drei Elemente "Artefakte" in Scrum. Sie sind die wichtigsten Vehikel für das Sammeln von Daten und Erfahrungen, die wir brauchen, um unsere Entscheidungen über die Zukunft zu treffen.

Das Product Backlog

Das erste Artefakt ist das Product Backlog. Es macht all die Arbeit transparent, die für das Produkt notwendig ist, um ein Produktziel zu erreichen. Das Produktziel gibt der Arbeit am Product Backlog einen Fokus, indem es dem Scrum Team erlaubt, sich auf ein einziges langfristiges Ziel für das Produkt festzulegen. Während Scrum Teams auf das Produktziel hinarbeiten, ändert sich das Product Backlog kontinuierlich. Dadurch reflektiert es neue Erkenntnisse und Möglichkeiten, die während der Arbeit ans Licht kommen. Ein neues Produktziel sollte nur dann gesetzt werden, wenn das vorherige erreicht (oder aufgegeben) wurde, da sonst der Fokus und das Engagement leiden.



Jede Idee, Hypothese, Funktion, Fehlerbehebung oder Aufgabe, die das Scrum-Team aktuell als für das Erreichen des Produktziels als notwendig empfindet, wird im Product Backlog durch einen einzelnen Eintrag repräsentiert. Einige davon sind eher grob und unklar formuliert, während andere kleiner und spezifischer sind. Alle Einträge sind geordnet nach ihrer Relevanz in Bezug auf die Erwartungen an das Produkt und seitens der Stakeholder.

Für ein Produkt gibt es nur ein einziges Product Backlog, unabhängig davon, wie viele Teams daran arbeiten. Andernfalls leidet die Transparenz – da es dann keine einzige "wahre Quelle" gibt, die besagt, welche Arbeit notwendig ist und in welcher Reihenfolge sie durchgeführt werden sollte.

Das Sprint Backlog

Das zweite Artefakt ist das Sprint Backlog. Es besteht aus einer Auswahl an Einträgen aus dem Product Backlog – denjenigen, die die Entwickler für notwendig halten, um das Sprintziel zu erreichen, zu dem sie sich für diesen Sprint verpflichtet haben. Das Sprint Backlog macht die gesamte Arbeit transparent, an der ein Scrum Team im aktuellen Sprint arbeitet oder arbeiten wird. Jedes Scrum Team hat sein eigenes Sprint Backlog, auch wenn es mit mehreren Teams an einem Produkt arbeitet. Das Sprint Backlog ist nicht statisch und ändert sich immer dann, wenn die Teams während des Sprints etwas dazulernen. In enger Zusammenarbeit mit dem Product Owner können die Entwickler Einträge hinzufügen oder entfernen, je nachdem, wie viel Zeit im Sprint noch übrig ist und wie relevant oder notwendig diese Einträge für das Sprintziel sind.

Das Inkrement

Das dritte Artefakt ist das Inkrement. Jedes Inkrement stellt einen weiteren Schritt auf das Gesamt-Produktziel hin dar. In einem Sprint wird ein Inkrement immer dann erstellt, wenn ein für ein Sprint Backlog ausgewählter Eintrag aus dem Product Backlog fertiggestellt wird. Zur Vermeidung von Verwirrung und auseinandergehenden Erwartungen wird ein Eintrag immer nur dann als "fertig" betrachtet, wenn er gründlich verifiziert wurde und der Definition of "Done" entspricht, zu der sich das Scrum Team verpflichtet hat. Scrum Teams können während eines Sprints mehrere Inkmente erstellen und diese sogar vor dem Sprint Review an ihre Stakeholder liefern.

Der Zweck jedes Inkrement ist es, Annahmen über die bisher geleistete Arbeit zu validieren. Alle mit einem Interesse an dem Produkt können es betrachten und erörtern, ob es ihren Bedürfnissen entspricht, ob sie verstehen, wie es funktioniert und ob es ihre Erwartungen erfüllt. Das Inkrement ist ein Treiber für neue Ideen: Nicht zuletzt ist die Interaktion mit etwas Greifbarem ein sehr gut geeigneter Anlass für Beteiligte, neue Möglichkeiten erkennen zu können.

Das Inkrement ist das wichtigste Mittel für Scrum Teams, um empirische Prozesskontrolle für komplexe Probleme auszuüben. Jedes Inkrement beginnt mit einer Vermutung, einer Hypothese oder einer potenziell wertvollen Idee, wie das Produkt näher an seine Ziele gebracht werden kann. Das ist es auch, was im Sprintziel festgehalten wird. Die Arbeit, die zum Erreichen dieses Ziels notwendig ist, wird hingegen im Sprint Backlog dokumentiert.

Wenn mehrere Scrum Teams an einem Produkt arbeiten, integrieren sie ihre Arbeit in jedem Sprint in ein einziges Inkrement, damit eine effektive Betrachtung der Ergebnisse stattfinden kann.

Und außerdem?

Offensichtlich gibt es noch viele weitere Elemente der Arbeit eines Teams, die wir transparent machen können. Man könnte zum Beispiel eine Stakeholder-Map erstellen, um ein besseres Gefühl dafür zu bekommen, wer und wo die Menschen sind, die ein Interesse an der Produktentwicklung haben. Teams können des Weiteren Metriken auf einem Dashboard sammeln und diese häufig auswerten. Das sind sehr gute Werkzeuge, deren genauere Betrachtung sich auf jeden Fall lohnt. Um Produkte auf empirische Art und Weise zu erschaffen, empfiehlt das Scrum Framework jedoch, dass die Teams mit der Schaffung von Transparenz mit den drei Schlüsselartefakten beginnen: dem Product Backlog, dem Sprint Backlog und dem Inkrement.



Die Scrum Events



Damit Sie Entscheidungen darüber treffen können, was als nächstes zu tun ist, müssen Sie sich häufig mit allen Beteiligten über das Product Backlog, das Sprint Backlog und das Inkrement austauschen und Gedanken machen. Das nennen wir 'Inspektion' ('Überprüfung'). Inspektion gibt es in vielen Formen. Wir können dem Product Backlog einen Sinn geben, indem wir es anschauen und Schlussfolgerungen ziehen – wie z.B. "es ist zu lang", "wir sollten diese Arbeit nicht vor dieser anderen erledigen" oder "wann können wir dies einem Stakeholder übergeben?" Wir können dem Inkrement einen Sinn geben, indem wir Anwender einbeziehen und unsere Annahmen mit ihnen zusammen validieren – wie z.B. "Verstehen die Benutzer diese neue Funktion?" Und wir können dem Sprint Backlog einen Sinn geben, indem wir sicherstellen, dass es tatsächlich durchführbar ist und unseren Plan für den aktuellen Sprint widerspiegelt.

Welche Form eine solche Überprüfung auch immer annimmt – sie bietet die beste Grundlage für Entscheidungen, wenn sie auf etwas Greifbarem und Konkretem beruht. Im Falle der Produktentwicklung ist die Inspektion einer fertiggestellten und eingesetzten Funktion der beste Weg, um Annahmen dazu und ihre Verwendung wirklich zu validieren. Das Betrachten einer Präsentation oder eines Designvorschlags zu einer Funktion mag nützlich erscheinen, aber jeder Anwesende macht dann immer noch viele – und wahrscheinlich unterschiedliche – Annahmen darüber, wie sie aussehen wird, wenn sie einmal tatsächlich implementiert ist.

Das Scrum-Rahmenwerk schlägt vor, dass Teams mindestens fünf sich wiederholende Möglichkeiten zur Überprüfung zur Verfügung haben sollten, um empirisch arbeiten zu können. Jede von ihnen hat einen bestimmte Maximaldauer und bietet eine eigene Perspektive auf die durchgeführte Arbeit. Zusammen bieten sie ein vollständiges Bild. Dies sind die fünf Scrum Events. Obwohl sie oft als "Meetings" bezeichnet werden – auch vom Scrum Guide – sind sie nicht als Meetings im traditionellen Sinne des Wortes gemeint, bei denen eine Gruppe von Menschen gelangweilt an einem Tisch sitzt. In der Tat: Wenn die Scrum Events mit klarem Ziel vor Augen durchgeführt werden, verringert sich die Notwendigkeit anderer formeller Meetings und macht Platz für natürliche Kollaboration und Koordination im aktuellen Moment der Team-Zusammenarbeit.

Der Sprint

Das erste zeitliche Ereignis ist das des Sprints selbst. Ähnlich wie man ein komplexes Puzzle löst, indem man auf einer kleineren Fläche startet, besteht der Zweck jedes Sprints in dem Versuch, einen Teil eines komplexen Problems zu lösen. Diese Teillösung wird durch eine inkrementelle Version des Produkts – das Inkrement – dargestellt. Diese kann gemeinsam überprüft werden, um zu entscheiden, was als nächstes geschehen soll. Dieses Inkrement sollte sich zumindest in einem Zustand befinden, in dem es direkt nach dem Sprint mit dem sprichwörtlichen Knopfdruck an die Stakeholder übergeben werden kann, wenn der Product Owner dies beschlossen hat. Noch besser ist eine Situation, in der die Teams während des gesamten Sprints releases können, wodurch ihre Lernfähigkeit noch weiter beschleunigt wird. Obwohl der Erfolg von Sprints immer unterschiedlich ausfallen wird – selbst ein einzelner Sprint ist mit einer Menge komplexer, unvorhersehbarer Arbeit verbunden –, lernt man in jedem Fall viel Neues über das Rätsel.

"Ähnlich wie man ein komplexes Puzzle löst, indem man auf einer kleineren Fläche startet, besteht der Zweck jedes Sprints in dem Versuch, einen Teil eines komplexen Problems zu lösen.



Sprints sollten immer gleich lang sein, um Kadenz und Vorhersehbarkeit für das Team und seine Beteiligten zu schaffen. Wenn ein Sprint zu lang wird, verliert ein Team wertvolle Gelegenheiten zur Validierung von Annahmen und zur Sicherstellung, dass die Arbeit noch in die richtige Richtung geht. Mit zunehmender Länge der Sprints steigt auch das Risiko, dass man Zeit damit verschwendet, falsche Dinge zu bauen. Das Scrum-Framework gibt keine Länge für Sprints vor – mit der Ausnahme, dass sie weniger als einen Monat dauern sollten. Es ist Sache des Teams zu entscheiden, wie schnell es lernen kann und will. Im Allgemeinen sollten Sprints so kurz wie möglich sein, während sie es dem Entwicklungsteam dennoch ermöglichen, eine sinnvolle neue Version des Produkts zu liefern, die Sprintziel erfüllt.

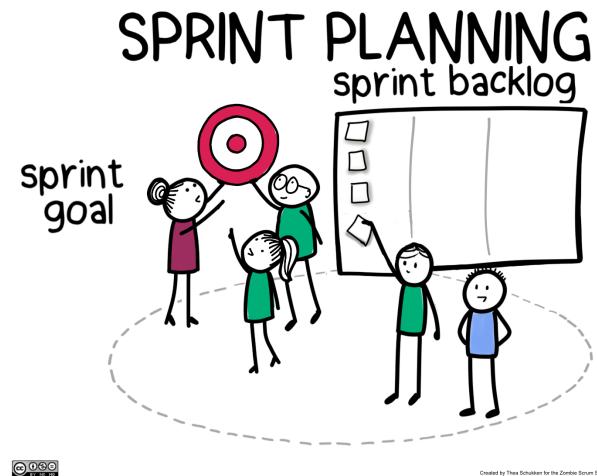
Während also der Sprint eine zeitlich begrenzte Gelegenheit ist, einen bestimmten Teil eines komplexen Produktentwicklungs-Problems zu erforschen, stellen die anderen vier Scrum-Events konkrete Möglichkeiten zur Förderung der Überprüfung und Anpassung innerhalb des Sprints dar.

Die Sprintplanung

Jeder Sprint in Scrum beginnt mit einer Gelegenheit für das Scrum-Team, den grundlegenden Plan für den kommenden Sprint zu erstellen. Dieser Anlass ist die so genannte Sprintplanung. Die erste und zentrale Frage ist dabei die nach dem Ziel für diesen Sprint. Ohne ein Ziel gibt es keine klare Orientierung, mit deren Hilfe alle Beteiligten während des Sprints ihre Kräfte bündeln und gemeinsam an Ideen arbeiten. Ein Ziel für einen Sprint kann beispielsweise sein, ein zusammenhängendes Set an Funktionen zu liefern, mit denen ein bestimmtes Problem gelöst wird. Es kann auch darum gehen, ein Bedürfnis von einer Gruppe Stakeholdern anzugehen. Vielleicht möchte aber auch ein Product Owner mit dem aus dem Sprint hervorgehenden Inkrement eine Hypothese oder kritische Annahme verifizieren.

Obwohl es für Product Owner empfehlenswert ist, mit einem Ziel im Hinterkopf in die Sprint-Planung zu gehen, arbeitet das Scrum Team zusammen, um dieses Vorhaben in ein Ziel zu verfeinern, das sowohl wertliefernd als auch innerhalb des Zeitrahmens eines Sprints machbar ist. Bei der Auswahl aus dem Product Backlog der Arbeit für den beginnenden Sprint zur Erreichung dieses Ziels arbeiten die Entwickler mit dem Product Owner zusammen. Das Product Backlog wird dabei bei Bedarf neu geordnet. Diese Planung wird zum Sprint Backlog. Da die Entwickler die eigentliche Arbeit durchführen, entscheiden sie auch darüber, was letztendlich in das Sprint Backlog kommt.

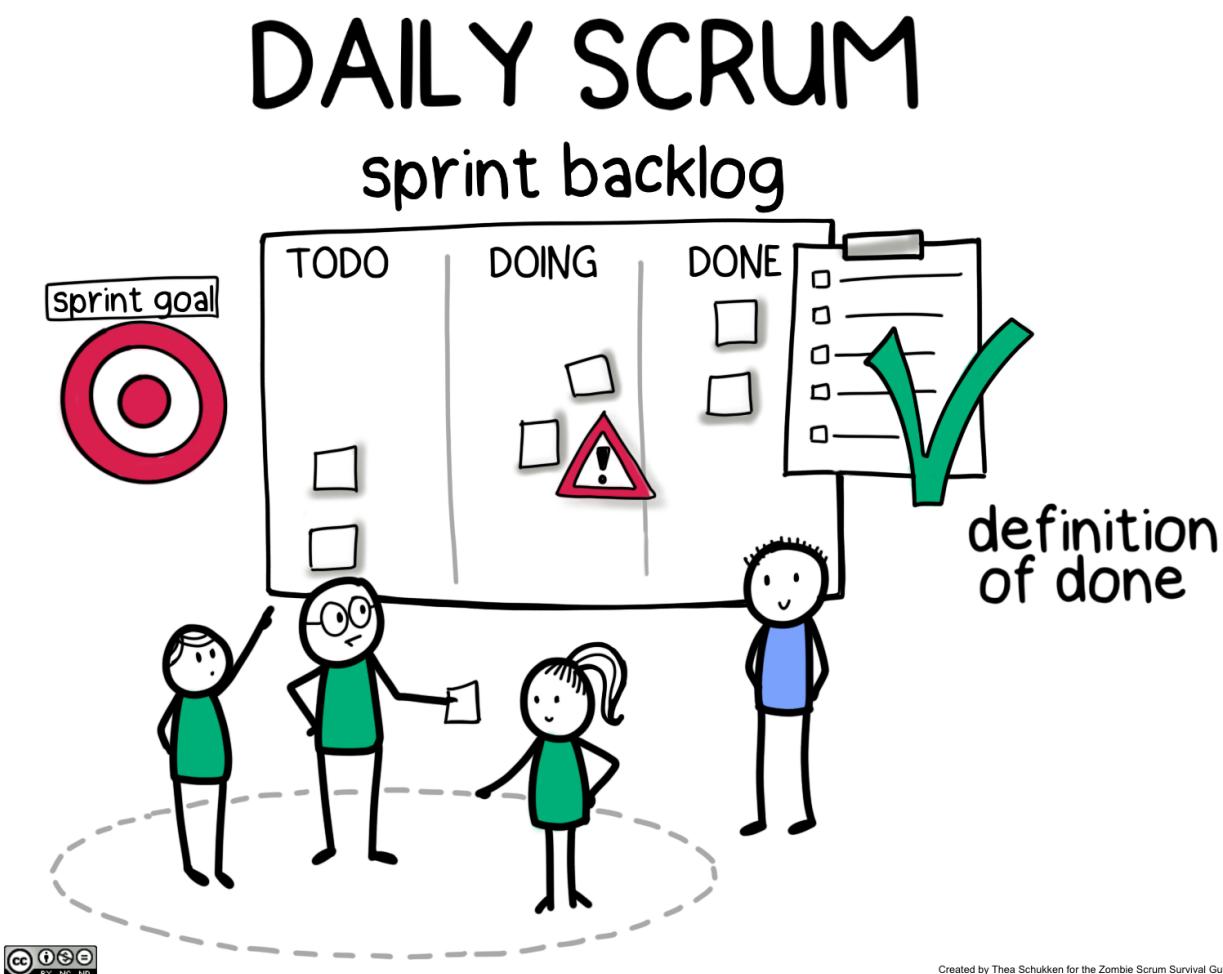
Bei Sprints mit einer Dauer von einem Monat sollte die Sprintplanung nicht mehr als acht Stunden in Anspruch nehmen. Je kürzer der Sprint ist, desto weniger Zeit darf sie in Anspruch nehmen. Die Sprintplanung ist abgeschlossen, wenn es eine grobe Skizze für den kommenden Sprint und einen detaillierteren Plan für die ersten paar Tage gibt. Dieser Plan wird normalerweise in Form einer Arbeitsaufteilung für die ersten Tage des Sprints festgehalten. Im weiteren Verlauf des Sprints übersetzen die Entwickler die grobe Skizze – wie sie durch das Sprintziel und das Sprint Backlog festgelegt wurde – in einen konkreteren Plan, wie sie gemeinsam daran arbeiten wollen. Mindestens eine Gelegenheit für diese Aktivität ist das Daily Scrum.



Daily Scrum

Das Daily Scrum findet alle 24 Stunden statt. So kann es dem Entwicklungsteam ermöglicht werden, die Komplexität selbst eines einzigen Sprints zu beherrschen. Das Team überprüft den Fortschritt seiner Arbeit im Hinblick auf das Sprintziel, der durch ihr Sprint Backlog transparent gemacht wird, und nehmen entsprechende Anpassungen vor. So können sie beispielsweise auf unerwartete Probleme stoßen, die eine enge Zusammenarbeit erfordern. Oder sie stellen fest, dass sie dem Sprint Backlog zusätzliche Aufgaben hinzufügen müssen, die zur Erreichung des Sprintziels erforderlich sind. Sie können auf Hindernisse stoßen, also auf Probleme, die sie bei der Erreichung des Sprintziels blockieren und die sie nicht lösen können.

Das Daily Scrum sollte nicht länger als 15 Minuten dauern. Es ist ein kurzer, als Mindestbasis stattfindender Austausch zur Koordination der Zusammenarbeit für die nächsten 24 Stunden. Falls mehr Koordination vonnöten ist, können die Entwickler dies natürlich den ganzen Tag über tun. Das Scrum Framework schreibt nicht vor, wie man ein Daily Scrum effektiv durchführt. Entwickler sollen sich dazu ermuntert fühlen, dass sie den für sie besten Weg zu einem gut funktionierenden Daily Scrum finden.



Sprint Review

Das Sprint Review findet am Ende eines Sprints statt, vor der Sprint-Retrospektive. Zweck ist es, die bisher geleistete Arbeit zu überprüfen und zu entscheiden, welche nächsten Schritte auf der Grundlage der daraus gewonnenen Erkenntnisse sinnvoll sind. Das Sprint Review ist eine Mindest-Gelegenheit während des Sprints, bei der diejenigen, die das Produkt entwickeln, mit den Stakeholdern zusammenkommen für eine gemeinsame Begutachtung der Ergebnisse des Sprints. Mit Blick auf die aktuellen Marktbedingungen, auf organisatorische Veränderungen, auf das Budget und auf den Zeitplan entscheiden sie gemeinsam über die nächsten Schritte.

Sprint Reviews sollten für einen Sprint mit einer einmonatigen Dauer nicht mehr als 4 Stunden lang sein. Je kürzer der Sprint, desto kürzer wird das Sprint Review in der Regel sein. Das Ergebnis des Sprint Reviews besteht aus Anpassungen des Product Backlogs auf Grundlage der gewonnenen Erkenntnisse. Dazu können neue Ideen gehören, die während der Sprint-Reviews auftauchen, entdeckte Fehler, Änderungen an sich bereits im Product Backlog befindlichen Einträgen oder die Neuordnung des Product Backlog selbst. In gewisser Weise geht es beim Sprint Review um die Beantwortung der Frage: "Was sind – basierend darauf, was wir in diesem Sprint gelernt haben – die nächsten Schritte?" Dies liefert wertvollen Input für die Sprintplanung und mögliche Sprintziele für kommende Sprints.

Eine reine, einseitige Präsentation durch die Entwickler, was sie fertiggestellt haben, stellt bei Sprint Reviews keine echte 'Überprüfung' dar. Das Inkrement tatsächlich zu 'inspizieren', bedeutet, es gemeinsam auszuprobieren und Feedback zu geben. Wir finden es richtig, wenn man Sprint Reviews als "Feedbackanlässe" statt als "Demos" betrachtet. Sprint Reviews sollten nicht die erste Gelegenheit sein, zu der ein Product Owner sieht, was die Entwickler fertigstellen konnten. Stattdessen ist das Sprint Review ein wichtiger und sich wiederholender Anlass für den Product Owner, die Stakeholder dazu einzuladen, damit sie das Produkt gemeinsam mit dem Scrum Team begutachten.

Eine reine, einseitige Präsentation durch die Entwickler, was sie fertiggestellt haben, stellt bei Sprint Reviews keine echte 'Überprüfung' dar.

SPRINT REVIEW



Created by Thilo Schüken for the Zombie Scrum Survival Guide
by Christian Verwoerd, Johannes Schäfer & Barry Owsley

Sprint-Retrospektive

Die Sprint-Retrospektive findet am Ende des Sprints statt, in der Regel direkt nach dem Sprint Review. Das gesamte Scrum Team nimmt an ihr teil. Ihr Zweck ist es, zu untersuchen, wie das Team bei der Erreichung des Sprintziels zusammengearbeitet hat – und was im nächsten Sprint verbessert werden kann. Während das Sprint Review mehr auf das Produkt und den Inhalt der geleisteten Arbeit fokussiert ist, zielt die Sprint-Retrospektive darauf ab, den Prozess zur Durchführung dieser Arbeit zu überprüfen.

Bei Sprints mit einer einmonatigen Dauer sollte die Sprint-Retrospektive nicht länger als 3 Stunden lang sein. Aber je kürzer der Sprint, desto kürzer ist in der Regel auch die Maximaldauer für dieses Ereignis. Ausgehend davon, was das Scrum Team aus der Überprüfung seiner Zusammenarbeit gelernt hat, kann es beschließen, Anpassungen für eine effektivere Arbeit vorzunehmen. Dies kann eine aktualisierte Definition of "Done", die Erforschung neuer Tools oder Technologien, eine Änderung der Vereinbarungen für die Zusammenarbeit oder eine neue Teamzusammenstellung sein. Mindestens eine umsetzbare Verbesserung darf direkt in das Sprint Backlog für den nächsten Sprint eingehen.



Created by Thea Schukken for the Zombie Scrum Survival Guide
by Christiaan Verwijs, Johannes Schartau & Barry Overeem



Product-Backlog-Verfeinerung [Refinement]

Wie sieht es mit der Verfeinerung [Refinement] des Product Backlog aus? Der Scrum Guide erwähnt dies als etwas, das irgendwo geschehen muss. Macht sie das nicht zu einem Event? Und ja, obwohl es definitiv ein wesentlicher Teil des Scrum-Rahmenwerks ist, ist es kein Event wie die anderen. Vielmehr handelt es sich um eine fortlaufende Aktivität. Und auch wenn dies wie ein Wortspiel erscheinen mag, so erfasst es doch einen wichtigen Unterschied. Alles beginnt mit dem Verständnis der vielen Variationen von Product-Backlog-Pflege:

- Klärung von Einträgen im Product Backlog, die zu unklar sind, um mit der Arbeit an ihnen zu beginnen. Dies geschieht vorzugsweise direkt mit den Personen, für die Sie die Einträge erstellen (die Stakeholder);
- Zerlegung der Arbeit am Product Backlog, die zu groß ist, um sie in einem einzigen Sprint abzuschließen (was im Allgemeinen auch bedeutet, dass sie zu unklar formuliert ist);
- Neuordnung der Arbeit im Product Backlog nach Bedarf, um die bevorstehenden Sprints so reibungslos und wertvoll wie möglich zu gestalten;
- Hinzufügen oder Entfernen von Einträgen aus dem Product Backlog, wenn sich neue Erkenntnisse ergeben;
- Abschätzung des Aufwands, der mit der Umsetzung bestimmter Punkte verbunden ist. Dies muss nicht so "formell" sein wie die Zuweisung von Story-Punkten (ein optionales Element in Scrum), T-Shirt-Größen oder welche Größenanpassungstechnik auch immer Sie verwenden. Ein Bauchgefühl ("Ja, wir wissen gut genug, was getan werden muss, und es fühlt sich in einem Sprint machbar an") ist auch in Ordnung;

In gewissem Sinne sind die Einträge in einem Product Backlog Erinnerungen an "Gespräche, die wir in Zukunft führen müssen". Und die Backlog-Pflege oder Verfeinerung ist der fortlaufende Prozess, diese Gespräche zu führen. Manchmal bedeutet dies, mit Stakeholdern über ein Thema zu sprechen, das im nächsten Sprint auftauchen könnte. Ein anderes Mal könnte es darum gehen kann, ein Thema zu klären, an dem das Team bereits arbeitet. Einige dieser Gespräche finden mit dem gesamten Entwicklungsteam statt, andere mit kleineren Gruppen. Einige Verfeinerungen können sogar individuell vorgenommen werden.

Anstatt also das Product Backlog Refinement als formalisiertes Meeting zu verstehen, die einmal während des Sprints stattfindet und an der jeder im Team teilnimmt, sollte es eine Reihe von fortlaufenden Aktivitäten sein, um die Inhalte im Product Backlog für die kommenden Sprints zu verfeinern. Es kann auf viele verschiedene Arten stattfinden mit unterschiedlich zusammengesetzten Beteiligten. Und es liegt ganz beim Scrum-Team, den besten Weg dafür zu finden und festzulegen.



Die Verantwortlichkeiten [Accountabilities] in Scrum

Eine gute Zusammenarbeit zwischen qualifizierten Fachexperten ist bei komplexen Problemen unerlässlich. Kreative und unkonventionelle Lösungen lassen sich leichter finden, wenn man die Perspektiven verschiedener Experten zusammenbringt. Um diese Zusammenarbeit zu erleichtern und die Komplexität in der Kommunikation und Entscheidungsfindung zu reduzieren, beschränkt sich das Scrum Framework bewusst auf drei Verantwortlichkeiten (oder "Rollen"). Anstelle von Rollen im hierarchischen Sinne, in denen eine Person die Autorität über die anderen hat, repräsentiert in Scrum jede der drei eine andere Perspektive, die bei empirischer Arbeit mindestens mit einbezogen werden sollte. Gemeinsam sind die drei ausreichend, um in jeder Umgebung empirisch arbeiten zu können. Weitere Verantwortlichkeiten oder Rollen sind demnach nicht notwendig (und ehrlich gesagt wären sie wahrscheinlich sogar etwas im Weg).

MEET THE SCRUM TEAM!

Product Owner



Scrum Master



Developers



Created by Thea Schukken for the Zombie Scrum Survival Guide
by Christiaan Verwijs, Johannes Schartau & Barry Overeem



Product Owner

Product Owner beziehen die Perspektive ein, was wertvoll ist (und was nicht), wenn es um die Ziele für und Erwartungen an das Produkt geht. Da das Scrum Team seine Zeit und sein Geld mit der Arbeit am Produkt verbringt, sind Product Owner dazu da, sicherzustellen, dass diese Investition den Stakeholdern einen Wert zurückgibt. Um zu entscheiden, was wertvoll ist und was nicht, ist eine enge Zusammenarbeit mit den am Produkt interessierten Personen sowie mit dem Entwicklungsteam wichtig.

"Product Owner beziehen die Perspektive ein, was wertvoll ist (und was nicht), wenn es um die Ziele für und Erwartungen an das Produkt geht."

Ein Produkt hat einen Product Owner und ein Product Backlog mit einem Produktziel. Um die Geschwindigkeit der Entscheidungsfindung hoch zu halten und eine rasche Anpassung zu ermöglichen, müssen Product Owner die volle Autorität über das Produkt haben. Sie haben das letzte Wort darüber, was die Ziele für das Produkt sind, was im Product Backlog steht und was nicht – und wie das Budget ausgegeben (oder sogar festgelegt) werden soll.

Product Owner stellen sicher, dass es ein Produktziel gibt, dass es ein geordnetes Product Backlog gibt und dass es sowohl dem Scrum Team als auch den Stakeholdern zur Verfügung gestellt wird. Das bedeutet nicht, dass der Product Owner die einzige Person im Scrum Team ist, die sich darum kümmert. Um den Wert der von den Entwicklern in jedem Sprint geleisteten Arbeit zu maximieren, ist es sinnvoll, dass Product Owner aktiv beim Schreiben, Verfeinern und Ordnen von Backlog-Einträgen mit ihnen zusammenarbeiten. Schließlich dreht sich alles um Kollaboration.

Entwickler

Entwickler sind alle Mitglieder eines Scrum Teams, die aktiv zur Lieferung eines Inkrementen beitragen, mit dem das Sprintziel erreicht wird. Es spielt keine Rolle, welche Art von Arbeit sie tun oder was ihre formale Berufsbezeichnung ist – für das Scrum-Framework sind sie alle "Entwickler". Zusammen repräsentieren sie die Perspektive, wie die für die Erfüllung der Erwartungen an das Produkt notwendige Arbeit zu erledigen ist und die Qualität dabei hochgehalten werden kann.

Da Produktentwicklung eine komplexe Arbeit ist, ist selbst die nahe Zukunft eines einzelnen Sprints schwer vorherzusagen. Es ist durchaus wahrscheinlich, dass Fragen, Herausforderungen und Probleme auftauchen, die das Team in seiner Fähigkeit behindern, das aktuelle Inkrement tatsächlich zu liefern. Es ist darüber hinaus ebenfalls wahrscheinlich, dass während des Sprints neue Ideen auftauchen, was in das Inkrement mit hinein genommen oder doch lieber weggelassen werden sollte. Aus diesem unvorhersehbaren Charakter selbst eines einzelnen Sprints ergeben sich drei wichtige Anforderungen daran, wie Entwickler arbeiten und sich organisieren.

"Entwickler repräsentieren die Perspektive, wie die für die Erfüllung der Erwartungen an das Produkt notwendige Arbeit zu erledigen ist und die Qualität dabei hochgehalten werden kann."

Die erste Anforderung ist, dediziert daran zu arbeiten, ihre Abhängigkeiten von anderen Personen, Abteilungen sowie Fähigkeiten außerhalb des Teams zu minimieren – also zu all jenen, die nicht zum Team gehören, aber dennoch häufig gebraucht werden, um fertige Arbeit liefern zu können. Jede Abhängigkeit ist ein Faktor, über den sie nur begrenzt Kontrolle haben und der sie in ihrer Fähigkeit, mit jedem Sprint ein fertiges Inkrement zu liefern, ausbremsen oder vollständig blockieren kann. Dies wirkt sich negativ auf ihre Fähigkeit aus, empirisch zu arbeiten. Abhängigkeiten können auf verschiedene Weise minimiert werden, von der Automatisierung (z.B. für Ausrollen und Testen) bis zum Erlernen neuer Fähigkeiten oder der Aufnahme von Personen mit notwendigen Kenntnissen in das Entwicklungsteam.



Die zweite Anforderung an Entwickler ist, dass sie Engpässe bei der Verteilung der Fähigkeiten und Kenntnisse innerhalb des Teams selbst verringern. Einen Tester im Scrum Team zu haben ist zwar besser, als wenn gar keiner da wäre. Aber wenn dieser Tester mit Arbeit überlastet ist, wird das gesamte Team in der Folge langsamer. Dasselbe gilt für andere Fähigkeiten wie Backend-Entwicklung, Design und Analyse. Das Team arbeitet deshalb aktiv an der Sicherstellung, dass auch andere Teammitglieder in der Lage sind, bestimmte Aufgaben zu erledigen. Das Ziel besteht hier nicht darin, dass jeder auf genau gleiche Weise dazu in der Lage ist – das ist offensichtlich unrealistisch und respektlos gegenüber Fähigkeiten, die Menschen jahrelang entwickelt haben –, vielmehr soll sichergestellt werden, dass man sich gegenseitig unterstützt oder bei Bedarf Aufgaben übernehmen kann. Das Scrum Framework fasst dies in der Aussage zusammen, dass "Scrum Teams funktionsübergreifend sind".

Die dritte Anforderung an Entwickler ist, dass sie sich oft auf ihre Intelligenz und Kreativität verlassen können müssen, um all das während eines Sprints zwangsläufig passierende Unerwartete zu bewältigen. Aus diesem Grund sollten Scrum Teams insofern selbstverwaltet sein, als dass sie Entscheidungen darüber treffen, wie sie arbeiten und was bei der Durchführung dieser Arbeit verbessert werden soll. In selbstverwalteten Teams gibt es keinen ernannten "Chef", der diese Entscheidungen für das Team trifft. Alle, die Arbeit erledigen – also die Entwickler – arbeiten zusammen, um Entscheidungen innerhalb der Grenzen des Scrum Frameworks zu treffen.

Scrum Master

Scrum Master beziehen die Perspektive der empirischen Prozesssteuerung ein sowie die der Qualität, mit der Transparenz, Kontrolle und Anpassung Gestalt annehmen – sowohl im Scrum Team als auch in seiner Umgebung. Der Scrum Master ist dazu da, die Elemente des Scrum Frameworks im Team und in der umgebenden Organisation mit Leben zu füllen. Dazu nehmen Scrum Master je nach Situation, in der sie sich befinden, unterschiedliche Positionen ein:

- **Lehrer:** sie lehren und erklären den Zweck des Scrum Frameworks als Mittel für empirische Arbeit. Sie geben sich viel Mühe beim Schaffen von Verständnis rund um die Artefakte, Ereignisse, Verantwortlichkeiten, Prinzipien – und wie diese Empirie und Agilität fördern;
- **Moderator:** sie machen die Säulen des Scrum Frameworks leichter verständlich, so dass Chancen für mehr Transparenz, Steuerung und Anpassung erkannt und maximiert werden können. Ein Beispiel dafür ist die Moderation von Scrum Events wie vom Scrum Team gewünscht oder benötigt. Ein weiteres Beispiel ist die Unterstützung des Product Owners bei der Anwendung von Techniken zur Verwaltung des Product Backlogs und dem Umgang mit Stakeholdern;
- **Problemlöser:** sie beseitigen Probleme, die das Entwicklungsteam daran hindern, seine Sprintziele zu erreichen – oder helfen zumindest bei der Beseitigung. Scrum Master helfen Scrum Teams, ihre Probleme selbstständiger zu lösen. Das müssen Teams erst lernen, und Scrum Master unterstützen sie bei diesem Prozess: Was während des ersten Sprints noch als echtes Hindernis galt, kann bei einem späteren Sprint zu einem Problem geworden sein, welches das Team leicht selbst lösen kann;
- **Change Agent:** sie helfen dabei, Hindernisse im Umfeld von Scrum Teams zu beseitigen, die der Empirie und der Agilität im Weg stehen. Beispielsweise teilen manche Organisationen "Testen" und "Ausrollen" oder "Inbetriebnahme" auf unterschiedliche Teams oder Abteilungen auf. Oder es gibt Praktiken im Personalmanagement, bei denen Einzelpersonen und nicht Teams belohnt werden. Die Beseitigung solcher Hindernisse kann ein Scrum Master in der Regel nicht alleine bewerkstelligen, daher arbeiten sie mit anderen Scrum Mastern, Product Owner und weiteren Beteiligten zusammen, um dies zu ermöglichen;
- **Coach & Mentor:** sie coachen das Scrum Team, indem sie starke offene Fragen stellen. Sie sind Mentoren für andere Scrum Master und helfen den Mitgliedern des Scrum Teams, Mentoren zu finden, welche die notwendige Erfahrung und die Fähigkeiten haben, um ihnen helfen zu können;



Scrum Master sind echte Führungs Personen. Anstatt die Aufmerksamkeit auf sich zu lenken, helfen sie anderen, so effektiv wie möglich zu sein. Sie managen oder leiten das Team nicht, indem sie ihnen sagen, was oder wie sie etwas tun sollen. Scrum Master sollten einzig dann eine starke Position einnehmen, wenn Entscheidungen getroffen werden sollen, die sich negativ auf den empirischen Prozess oder das Sicherheitsgefühl im Scrum Team beim Umgang mit anderen auswirken könnten.

"Scrum Master beziehen die Perspektive der empirischen Prozesssteuerung ein sowie die der Qualität, mit der Transparenz, Kontrolle und Anpassung Gestalt annehmen."

Gemeinsam: Das Scrum Team

Zusammen bilden die drei Verantwortlichkeiten ein Scrum Team. Sie haben die volle Autorität, alle für ihr Produkt relevanten Entscheidungen zu treffen. Product Owner entscheiden, wie das zur Verfügung stehende Budget verwendet wird, um den Wert für die Beteiligten zu maximieren. Die Entwickler entscheiden, wie das Produkt gebaut werden soll und verfügt über alle dafür erforderlichen Fähigkeiten. Und Scrum Master garantieren, dass dies alles so geschieht, dass die Empirie maximiert wird. Andere Rollen sind nicht erforderlich.

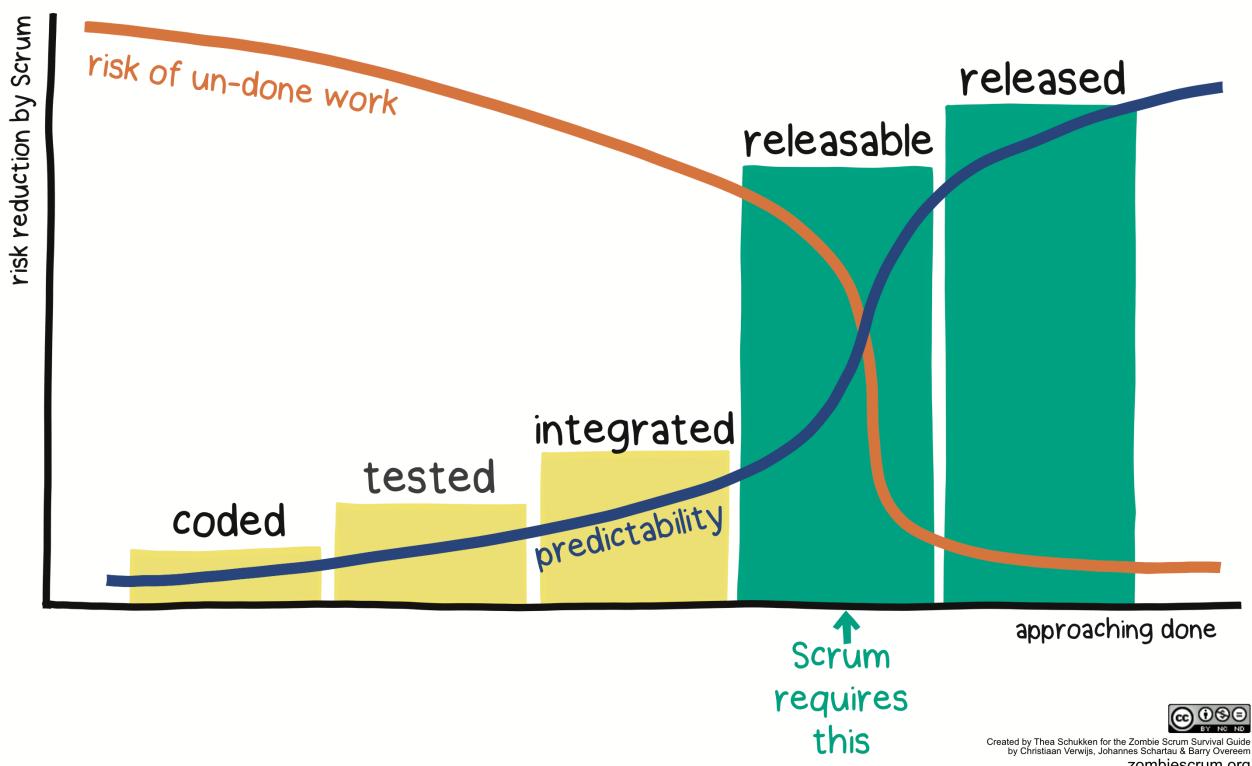
Das Scrum Framework ist auf eine Art konzipiert, die in diesem Zitat von [Gunther Verheyen](#) wunderbar beschrieben wird: "angesichts der Komplexität ist Einfachheit unser Weg". Es könnte verlockend sein, Rollen, Regeln, Praktiken und Strukturen zum Scrum Framework hinzuzufügen – vielleicht, weil Sie meinen, dass Ihre Organisation einfach anders ist. Und auch wenn einige Situationen das tatsächlich rechtfertigen mögen, ist es in komplexen Umgebungen eigentlich dennoch angebracht, sich auf eine so einfache Gestaltung der Arbeit wie nur möglich zu konzentrieren. Dies gilt auch für eine Skalierung. Das Hinzufügen weiterer Teams erhöht die Komplexität. Suchen Sie stattdessen doch nach Wegen, mehr mit weniger Leuten zu erreichen.



Zwei zentrale Prinzipien

Bei der Beschreibung des Scrum-Rahmenwerks erklären viele die Ereignisse und die Rollen darin. Diese Elemente sind sicherlich wichtig – aber ihre Wirksamkeit ergibt sich maßgeblich daraus, wie gut die Teams zwei zentrale Prinzipien verstehen, die alles vorantreiben.

In jedem Sprint ein fertiges Inkrement liefern



Wenn Sie den Zweck des Scrum Framework in einem einzigen Satz zusammenfassen würden, dann könnte dieser so lauten: Empirisch arbeiten durch das Liefern eines fertigen Inkrementen in jedem Sprint. Das erklärt auch, warum es so wichtig ist, dass Scrum Teams Zeit investieren in die Klärung, was zur Erstellung eines fertigen Inkrementen erforderlich ist. Welche Arbeit ist dafür genau notwendig? Welche Checks und Tests sind erforderlich, um unseren internen Qualitätsrichtlinien zu entsprechen? Wer muss daran beteiligt sein? Dieses gemeinsame Verständnis wird als "Definition of Done" bezeichnet. Es ist üblicherweise als Checkliste verfasst.

Die Arbeit im Product Backlog ist in der Regel voll mit Annahmen, bis die Arbeit abgeschlossen und die Ergebnisse in die Hände der Benutzer gelegt wird. Solche Annahmen haben zum Beispiel die folgenden Fragen zur Grundlage: "Wird die Realisierung dieses Backlog-Eintrags die Nutzererfahrung verbessern?", "Werden die Nutzer verstehen, wie es funktioniert?", "Ist die Performance gut genug?" Andere Annahmen beziehen sich auf die Arbeit, die für die Umsetzung eines Eintrags geleistet werden muss, wie z.B. "Wie einfach wird es sein, diesen Backlog-Eintrag zu testen und auszuliefern?", "Auf welche Probleme werden wir bei der Arbeit an diesem Eintrag stoßen?"

All diese Annahmen stellen Risiken dar. Das Risiko, dass Sie missverstanden haben, was ein Nutzer wollte, und dass Sie es überarbeiten müssen. Das Risiko, dass Sie auf technische Einschränkungen stoßen, wenn Sie mit dem Schreiben des Codes beginnen. Das Risiko, dass die Funktion viel schlechter funktioniert als erwartet. Das Risiko, dass sich das Schreiben automatisierter Tests als viel schwieriger als gedacht erweist. Oder das Risiko, dass Sie Geld und Zeit für eine Funktion ausgeben, die am Ende nicht verwendet wird. Alle diese Risiken haben gemeinsam, dass sie irgendwo unterwegs zu mehr und unerwarteter Arbeit führen. Und diese Art von "Nacharbeit" neigt dazu, unsichtbar zu bleiben, bis sie auftaucht, wenn man es am wenigsten erwartet – und dann das, woran man in diesem Sprint arbeitet, durcheinander bringt. Dies nennt man "unerledigte Arbeit".

Der beste Weg, dem Risiko unerledigter Arbeit vorzubeugen, besteht darin, dafür zu sorgen, dass jeder Sprint – mindestens einmal – zu einem fertigen Inkrement führt, das potenziell ausgeliefert werden kann. Das bedeutet, dass das Inkrement vollständig getestet wurde und funktioniert. Texte und Bildmaterial sind in endgültigem Zustand. Dokumentations- und Auslieferbündelungen sind auf dem neuesten Stand. Performance und Sicherheit entsprechen den organisatorischen Standards. Von hier aus kann das Inkrement mit dem sprichwörtlichen Knopfdruck an die Benutzer freigegeben werden. Von diesem Zeitpunkt an wissen Sie, dass es wenig bis gar keine potentielle unfertige Arbeit geben wird, die den Fokus und die Vorhersehbarkeit zukünftiger Sprints durcheinander bringen kann. Noch wichtiger ist, dass Sie durch die Freigabe des Inkrementes eine empirische Prozesskontrolle ausüben können, um zu sehen, ob das Inkrement wirklich Ihr Produkt seinen Zielen näher bringt, anstatt einfach davon auszugehen, dass dies der Fall sein wird.

In dem Maße, in dem sich die Fähigkeit von Scrum-Teams, fertige Inkremeante zu liefern, nach rechts verschiebt, sinkt das Risiko von unerledigter Arbeit – und die Vorhersagbarkeit steigt (inspiriert durch die Arbeit von Gunther Verheyen). In jedem Sprint ein fertiges Inkrement zu erstellen, ist sicherlich eine Herausforderung. Aber diese wird durchaus absichtlich gestellt. Denn wenn man so viel Druck auf das das Produkt erzeugende System ausübt – also den Druck, die ganze für die Erreichung dieses Ziels notwendige Arbeit zu erledigen –, wird deutlich, wo Verbesserungen notwendig sind. Wo Fähigkeiten, Werkzeuge und Technologien fehlen. Wo die Bürokratie im Weg steht. Wo Hindernisse beseitigt werden müssen, um tatsächlich empirisch arbeiten und das inhärente Risiko komplexer Arbeit reduzieren zu können. Wenn der Fokus darauf gerichtet ist, mindestens einmal pro Sprint ein fertiges Inkrement zu liefern, wird das System irgendwann an den richtigen Stellen anfangen zu schmerzen und Möglichkeiten zur Überprüfung und Anpassung schaffen.

"Wenn der Fokus darauf gerichtet ist, mindestens einmal pro Sprint ein fertiges Inkrement zu liefern, wird das System irgendwann an den richtigen Stellen anfangen zu schmerzen und Möglichkeiten zur Überprüfung und Anpassung schaffen."

Verwendung eines gemeinsamen Produktziels und Sprintziels für mehr inhaltlichen Zusammenhang

Der Scrum Guide erwähnt 40 Mal das Wort "Ziel" – mehr als jedes andere Element aus dem Rahmenwerk. Wir wollen an dieser Stelle nicht vorschlagen, aus solchen Beobachtungen heraus Entscheidungen darüber zu treffen, wie mit Scrum gearbeitet werden soll. Aber es kann uns schon einen Hinweis darauf geben, wie wichtig gemeinsame Ziele wirklich sind.

Die Erklärung, warum das Produktziel und das Sprintziel so wichtig sind, geht darauf zurück, wofür das Scrum Framework entwickelt wurde: für eine wirksame Erkundung komplexer Probleme. Ziele helfen uns auf drei sich ergänzende Arten bei der Navigation durch diese Komplexität.



Erstens bieten sie dem Scrum Team eine Orientierungshilfe beim Treffen von Entscheidungen, wofür die vorhandene Zeit genau aufgewendet werden soll: Einige Aufgaben aus dem Sprint Backlog könnten für das Erreichen des Sprintziels wichtiger sein als andere. Der zweite Punkt ist, dass Sprintziele dem Team ermöglichen, sich auf das Wesentliche zu konzentrieren. Wenn es hart auf hart kommt – wie oft der Fall – kann das Team entscheiden, bestimmte Arbeiten fallen zu lassen und andere höher zu priorisieren. Oder, wenn es die Zeit erlaubt, können sie während des Sprints Aufgaben hinzufügen, die ihnen zusätzlich zur Zielerreichung notwendig erscheinen. Und drittens fördern Sprintziele die Zusammenarbeit, indem sie dem Entwicklungsteam eine klare und gemeinsame Vorgabe machen, sich selbst zu organisieren, anstatt an unterschiedlichen Vorhaben zu arbeiten. Zusammenarbeit ermöglicht die Art von unkonventionellem Denken und Teamgeist, die bei der Lösung komplexer Probleme voneinander nötig ist.

Aber das Produktziel und das Sprint-Ziel geben auch den Fokus auf die drei Verantwortlichkeiten des Scrum-Frameworks. Sie geben den Entwicklern Anleitung und Richtung, worum sie sich selbst organisieren sollen. Sie erlauben Product Ownern, sich auf ihre Ambitionen für das Produkt zu konzentrieren und darauf, wie sich das in Ziele für die verschiedenen Sprints übersetzt, anstatt sich mit den Details der Umsetzung zu beschäftigen. Und schließlich können Scrum Master die Effektivität der empirischen Prozesssteuerung erhöhen, indem sie das Produktziel und das Sprintziel als den sprichwörtlichen Kanarienvogel in der Kohlenmine verwenden.

Wenn Teams Schwierigkeiten bei der Festlegung von Sprintzielen haben, ist das ein guter Grund für Scrum Master, ihre Detektivhüte aufzusetzen und zu untersuchen, was die Ursache dafür sein könnte. Die Teams könnten zu groß oder zu klein sein, oder es könnten ihnen bestimmte Fähigkeiten oder Personen fehlen. Der Product Owner hat vielleicht keine Autorität oder keine Vision. Oder es findet keine ausreichende Backlog-Pflege statt.

Diese Erklärung sollte deutlich machen, warum es nur ein Produktziel pro Produkt und ein Sprintziel pro Sprint geben sollte. Mehrere Ziele verwirren nur den Fokus, verringern das Engagement und schränken die Transparenz ein. Natürlich kann ein neues Ziel formuliert werden, wenn das vorherige erreicht oder aufgegeben wurde.

Kurz gesagt: Bei komplexer Arbeit sind gemeinsame Ziele wie Produktziele und Sprintziele die Leuchttürme, die Teams helfen, durch dichten Nebel den Hafen zu erreichen. Ohne sie wird man sich wahrscheinlich verirren und auf Land auflaufen.



Fünf Schlüsselwerte für mehr Empirie

Bis zu diesem Punkt haben wir die Mechanik und die Prinzipien des Scrum-Rahmenwerks behandelt. Zwar können sie die Ermöglichung empirischer Prozesskontrolle auf fast wundersame Weise unterstützen. Sie würden aber fast nichts bewirken, wenn ein dies unterstützendes Verhalten der Beteiligten fehlt.. Wie ehrlich kann die Überprüfung des Inkrements während eines Sprint Reviews sein, wenn Scrum Teams Angst haben, offen über die technischen Herausforderungen zu sprechen, die sie sehen, und stattdessen lieber zu der Aussage greifen, es sei "alles in Ordnung"? Wie effektiv kann ein Scrum-Team sein, wenn der Product Owner Angst hat, Einträge im Product Backlog zurückzuweisen, die nicht zum Ziel passen? Wie viel Wert und Nutzen wird von einem Entwicklungsteam generiert, das sich ständig mit den neuesten und glanzvollsten Technologien ablenkt? Wie wirkt sich eine Änderung der Teamzusammensetzung ohne deren Wissen, Zutun oder Zustimmung auf ihre Bereitschaft aus, sich für die Arbeit im Team einzubringen?

Empirische Arbeit ist schwierig – besonders in Organisationen, die nicht daran gewöhnt sind. Um den Teams und ihren Stakeholdern etwas an die Hand zu geben, mit dem sie solche Entscheidungen vorantreiben können, legt das Scrum-Rahmenwerk besonderen Wert auf fünf Leitwerte. Bei jeder anstehenden Entscheidung können sich die Teams fragen, wie sich die ihnen offenstehenden Möglichkeiten auf diese fünf Werte auswirken:

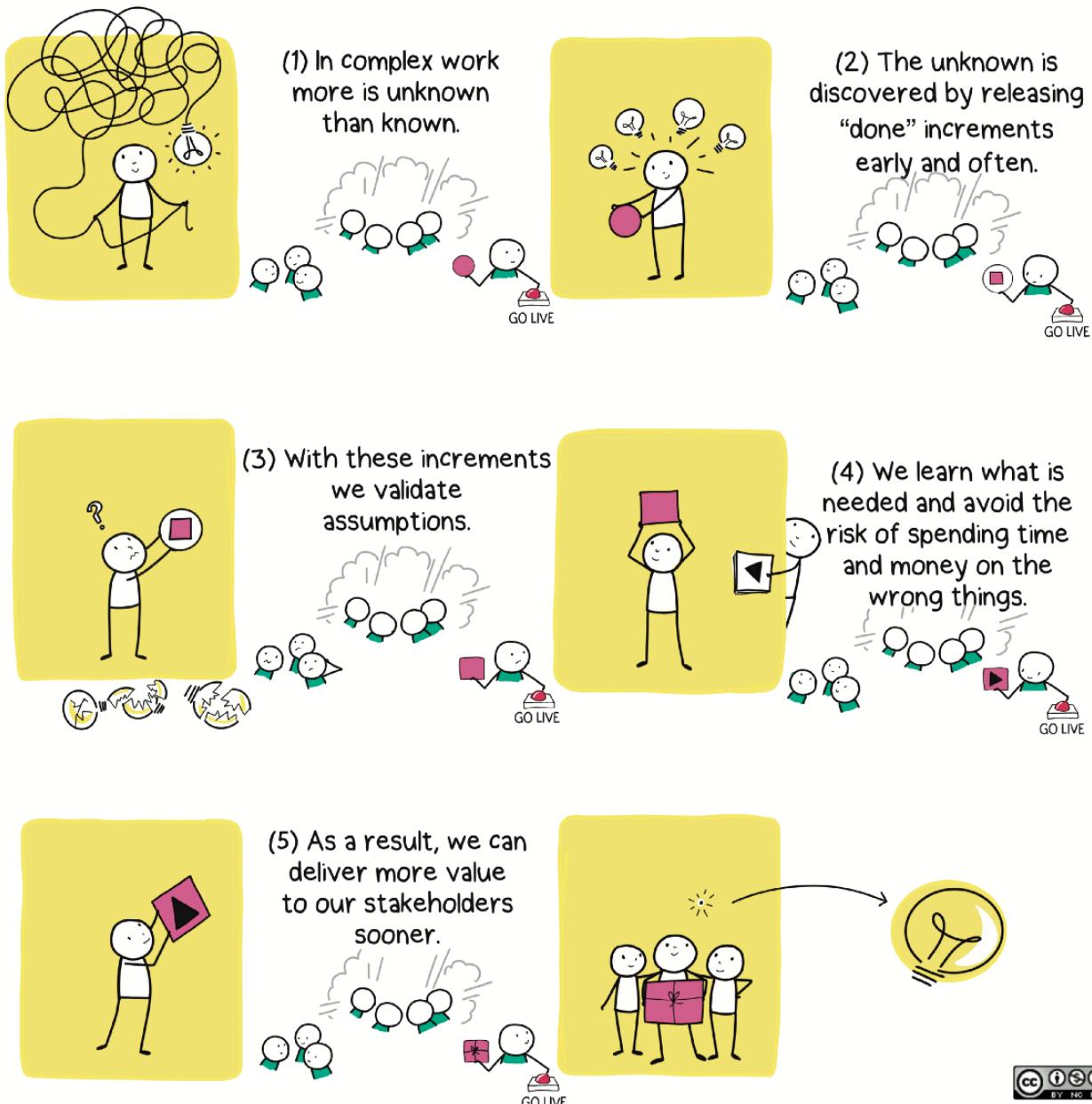
- **Offenheit:** Offen sein im Hinblick darauf, wie alles steht. Was läuft gut? Was läuft nicht gut? Wo liegen die Herausforderungen und Chancen?
- **Mut:** Seien Sie mutig, das Richtige zu tun. Sagen Sie "Nein" zu Dingen, die den empirischen Prozess behindern. Zeigen Sie Mut, indem Sie gemeinsam an schwierigen Herausforderungen arbeiten. Fragen Sie nach und geben Sie Feedback zu Dingen, bei denen Sie sich nicht sicher sind. Stellen Sie Fragen und geben Sie zu, wenn Sie etwas nicht wissen oder sich über etwas unsicher sind;
- **Fokus:** Konzentrieren Sie sich auf das Sprintziel und die Ziele des Scrum Teams. Schaffen Sie einen Raum, in dem die Leute den Fokus behalten und aufrechterhalten können;
- **Respekt:** Respektieren Sie die Fähigkeiten, Expertise und Intelligenz der Mitglieder des Scrum Teams. Vertrauen Sie auf ihre Fähigkeit, sich bei komplexen Problemen selbst zu organisieren. Und respektieren Sie die Unsicherheit, die mit komplexer Arbeit verbunden ist;
- **Verpflichtung / Engagement:** Schaffen Sie ein Umfeld, in dem sich die Menschen persönlich dazu verpflichten können, gemeinsam als Team auf das Sprintziel hinzuarbeiten;

Diese Werte im alltäglichen Verhalten zu vertreten lernen ist ein lebenslanger Weg. Je fähiger die Teams werden, desto effektiver werden häufige Transparenz, Überprüfung und Anpassung sein. Die gute Nachricht ist, dass das Scrum-Rahmenwerk hervorragend funktionierende Grenzen bietet, innerhalb derer Teams bei der Umsetzung dieser Werte lernen und wachsen können.



Gemeinsam: Ein einfaches Rahmenwerk

Scrum ist ein einfaches Rahmenwerk als Navigationshilfe bei komplexen, anpassungsfähigen Problemen. Es schreibt nur vor, was Teams tun sollten, um empirisch zu arbeiten, aber nicht, wie sie es tun sollten. Da jedes Team, jedes Produkt und jede Organisation anders ist, müssen Teams ihren eigenen Weg finden, um dies für sich zu nutzen. Wenn sie das tun, können sie das in komplexer Arbeit innewohnende Risiko verringern, früher damit beginnen, ihren Interessenvertretern einen Mehrwert zu bieten, und schneller reagieren. Diese Reise wird für einige Scrum Teams leichter sein als für andere. Aber der Wandel wird folgen, wenn Sie nicht lockerlassen.



Created by Thea Schukken for the Zombie Scrum Survival Guide
by Christiaan Verwijs, Johannes Schartau & Barry Overeem
zombiescrum.org

Unleashing Teams All Over The World

We are The Liberators – Barry Overeem and Christiaan Verwijs. Our mission is to create data-driven products to unleash the superpowers of teams all over the world. We do this together with a growing community of patrons.

Awesome content for awesome teams

We unleash teams with our [blogposts](#), our [podcast](#), our [newsletter](#), our [videos](#), and our frequent [meetups](#). While we offer most of this for free, we also have plenty of premium content in our [webshop](#) for you to explore.

Supported by the community

We are super proud that The Liberators is almost entirely funded by the community. If you appreciate our work too, you can already support us for 12 dollars/year by becoming a [patron](#). In return, you gain free access to premium content, we share our work-in-progress and involve you in creating more awesome content.



Thank you for respecting our work!

We work hard to create high-quality content that puts you in a position to unleash your team. We're sure you appreciate that a lot of our time and money goes into this. At the same time, content like this is our main source of income. It's what pays our bills :)

If you purchased this content, we ask only that you treat our work with respect and don't share it with people outside your team. If you stumbled on it elsewhere without paying for it, and it offers you value, would you consider [supporting us too](#)? You can also check out our [other offerings](#).