KAPITEL 2

Die Werte

Nachdem sie im vorigen Kapitel bereits kurz vorgestellt wurden, werden wir die Werte, auf denen Scrum basiert, hier nun ausführlich beschreiben. Erst durch das Zusammenspiel von Werten, Prinzipien und Praktiken entfaltet Scrum sein volles Potenzial. Deshalb haben wir den Werten ein eigenes Kapitel gewidmet. Wir definieren zunächst den Begriff »Wert«, bevor wir die fünf Scrum-Werte vorstellen und deren praktische Bedeutung anhand von Beispielen illustrieren. Abschließend zeigen wir, wie die Werte, Prinzipien und Praktiken ineinandergreifen, um in Scrum ihre Wirkung zu entfalten.

Was sind Werte?

Der Kern aller agilen Vorgehensmodelle ist ein Wertesystem. Diese Werte geben den Prinzipien und Praktiken von Scrum eine Bedeutung und der Mechanik von Scrum einen Sinn.

Wenn wir hier den Begriff »Werte« verwenden, meinen wir immaterielle, ethische (oder innere) Werte, die auf Werterfahrungen beruhen. Für den deutschen Soziologen und Sozialphilosophen Hans Joas sind Werte attraktiv-motivierend. Im Gegensatz dazu stehen die Normen, die restriktiv-obligatorisch sind. Während Normen bestimmte Handlungen ausschließen, erschließen Werte den Menschen neue Handlungsmöglichkeiten.

Agile Werte und die aus ihnen abgeleiteten Prinzipien scheinen oft abstrakt und wenig alltagsnah. Es ist deshalb schwierig, die Werte in der täglichen Arbeit zu leben. Das aber ist unerlässlich, wenn die Werte einen Nutzen stiften sollen. In [Wiechmann 2020] zeigen

Robert Wiechmann und Laura Paradiek Übungen und Techniken aus dem Improvisationstheater, die helfen, diese Werte greifbar und erfahrbar zu machen.

Die Werte in Scrum

Scrum gründet auf einem Fundament aus fünf Werten (siehe Scrum Guide):

- *Selbstverpflichtung (Commitment)*: »Das Scrum-Team committet sich, seine Ziele zu erreichen und sich gegenseitig zu unterstützen.« – Seien Sie willens, sich einem Ziel zu verpflichten. Scrum gibt Ihnen die nötigen Befugnisse, um diese Verpflichtung zu erfüllen.
- *Fokus (Focus)*: »[Der] primäre Fokus [des Scrum-Teams] liegt auf der Arbeit des Sprints, um den bestmöglichen Fortschritt in Richtung dieser Ziele zu bewirken.« – Erledigen Sie Ihre Arbeit. Verwenden Sie all Ihre Energie und Erfahrung auf die Aufgabe, zu der Sie sich verpflichtet haben. Machen Sie sich um alles andere keine Sorgen.
- *Offenheit (Openness)*: »Das Scrum-Team und dessen Stakeholder sind offen in Bezug auf die Arbeit und die Herausforderungen.« – Machen Sie alle Informationen für alle sichtbar.
- *Respekt (Respect)*: »Die Mitglieder des Scrum-Teams respektieren sich gegenseitig als fähige, unabhängige Personen und werden als solche auch von den Menschen, mit denen sie zusammenarbeiten, respektiert.« – Akzeptieren Sie die Prägung jedes Individuums durch seinen persönlichen Hintergrund und seine Erfahrungen. Respektieren Sie die verschiedenen Menschen, die sich zu einem Team zusammengeschlossen haben.
- *Mut (Courage)*: »Die Mitglieder des Scrum-Teams haben den Mut, das Richtige zu tun: an schwierigen Problemen zu arbeiten.« – Haben Sie den Mut, sich einem Ziel zu verpflichten, zu handeln, Offenheit zu zeigen und für gegenseitigen Respekt einzutreten.

Auch eXtreme Programming [Beck 2004], eine leichtgewichtige und agile Softwareentwicklungsmethode, basiert auf fünf Werten. Der Wertekatalog ist bewusst klein gewählt, damit er von allen, die mit agilen Methoden arbeiten, gleichermaßen akzeptiert und gelebt wird. Wer einmal ernsthaft versucht hat, sein Handeln konsequent an den fünf Scrum-Werten auszurichten, wird festgestellt haben, wie anspruchsvoll diese wertorientierte Arbeitsweise ist. Wenn es dem Scrum-Team jedoch gelingt, diese fünf Werte zu verkörpern und zu leben, erweckt es die drei Säulen des Empirismus – Transparenz, Überprüfung (Inspection) und Anpassung (Adaptation) – zum Leben und schafft unter allen Beteiligten Vertrauen und Zuversicht. Betrachten wir deshalb die Scrum-Werte etwas genauer.

Selbstverpflichtung (Commitment)

»Das Scrum-Team committet sich, seine Ziele zu erreichen und sich gegenseitig zu unterstützen.«

Den ersten Scrum-Wert, »Commitment«, ins Deutsche zu übersetzen, ist nicht leicht, da dieser Begriff sehr vielschichtig und die Bedeutung im Scrum-Kontext sehr speziell ist. Hier bezeichnet »Commitment« insbesondere den gemeinsamen Willen innerhalb des Scrum-Teams, das miteinander erarbeitete Sprint-Ziel zu erreichen. Dazu geben die Developer im Sprint Planning eine Prognose (einen Forecast) ab. Eine nach bestem Wissen und Gewissen ermittelte Liste von Product Backlog Items geht dann als Sprint Backlog in den nächsten Sprint ein. Im Laufe des Sprints wird dieser Forecast kontinuierlich überprüft und angepasst. Wurde das Sprint-Ziel am Ende des Sprints dennoch nicht erreicht, kann das ganz unterschiedliche Gründe haben, die es im Rahmen einer Retrospektive zu untersuchen gilt.

Ein gutes Scrum-Team steht zu seinem Produkt. Es will ein Produkt mit größtmöglichem Nutzen schaffen, das den gestellten Anforderungen gerecht wird. Diese Anforderungen werden mit Blick auf das Produkt-Ziel und das Sprint-Ziel der Reihe nach umgesetzt – Sprint für Sprint. Und weil ein Scrum-Team weiß, wie wichtig es ist, dass am Ende das Sprint-Ziel erreicht ist (und wie gut das tut), wird es bei der Sprint-Planung sorgfältig zu Werke gehen.

Fokus (Focus)

»[Der] primäre Fokus [des Scrum-Teams] liegt auf der Arbeit des Sprints, um den bestmöglichen Fortschritt dieser Ziele zu bewirken.«

Der Sprint ist gestartet. Nun gilt es, die in das Sprint Backlog übernommenen Product Backlog Items umzusetzen. Das ist die Verantwortung der Developer – nicht mehr und nicht weniger. Deshalb sollen sich alle Developer auf diese Aufgabe konzentrieren. Das ist leichter gesagt als getan. Tagesgeschäft, Tätigkeiten für andere Teams und Projekte sowie »Kannste mal eben«-Aufgaben machen dem um Fokus bemühten Developer das Leben schwer.

Mehrere wissenschaftliche Untersuchungen (z.B. [Baethge 2010]) haben gezeigt, dass sowohl das gleichzeitige Bearbeiten mehrerer Aufgaben (Multitasking) als auch eine hohe Frequenz von Arbeitsunterbrechungen negative Auswirkungen haben. Die Qualität der Arbeitsleistung nimmt ab, unter anderem weil schneller gearbeitet wird, um die durch Unterbrechungen verlorene Zeit zu kompensieren. Trotzdem ist die Gesamtbearbeitungsdauer für jede Einzelaufgabe beim Multitasking höher als bei einer unterbrechungsfreien sequenziellen Abarbeitung der Aufgaben. Eine häufige Folge von Multitasking und Unterbrechungen ist Stress. Eine fokussierte Arbeitsweise sorgt dafür, dass die Zahl der Unterbrechungen abnimmt.

Scrum unterstützt eine fokussierte Arbeitsweise. Ein ideales Scrum-Team arbeitet für genau ein Produkt und ist personell stabil. Produkt- und Sprint-Ziel geben Orientierung, und die Aufgaben sind klar und ausreichend konkret benannt, sodass einer konzentrierten Bearbeitung nichts im Weg steht. Anhand der Akzeptanzkriterien und der *Definition of Done* wird entschieden, wann eine Aufgabe als erledigt betrachtet werden kann. Die Product Backlog Items werden mit Blick auf das Sprint-Ziel abgearbeitet. Idealerweise befindet sich zu jedem Zeitpunkt im Sprint genau ein Product Backlog Item in Bearbeitung. Ein regelmäßiger Austausch ist durch das Daily Scrum garantiert und fest in den Tagesablauf integriert, hat also nicht den Charakter einer (ungeplanten) Unterbrechung.

Nicht nur die Developer arbeiten fokussiert, auch der Product Owner hat das Produkt im Fokus. Er konzentriert sich auf die Beschreibung

der Ideen, Konzepte und Eigenschaften des Produkts. Grundsätzlich wird er sich möglichst viel Funktionalität wünschen. Demgegenüber muss der Scrum Master die Developer darin unterstützen, dass die Funktionen des Produkts in der geforderten Qualität und innerhalb der durch die Sprint-Länge vorgegebenen Zeit realisiert werden. Im Zweifelsfall wird er empfehlen, auf die Aufnahme eines weiteren Product Backlog Item in das Sprint Backlog zu verzichten, wenn dadurch die Erreichung des vom Scrum-Team formulierten Sprint-Ziels gefährdet wäre.

Der Fokus des Scrum Master liegt auf dem Scrum-Rahmenwerk und den Teamregeln. Er tut alles, um die Produktivität des Teams zu gewährleisten und zu verbessern. Dazu hilft er auch allen Personen außerhalb des Scrum-Teams, die Einfluss auf dessen Arbeit (oder Interesse an dessen Arbeitsweise) haben, ihren Beitrag und ihr Handeln so zu gestalten, dass sie das Scrum-Team, das Produkt und die Organisation optimal unterstützen.

Offenheit (Openness)

»Das Scrum-Team und dessen Stakeholder sind offen in Bezug auf die Arbeit und die Herausforderungen.«

Offenheit hat mehrere Facetten. Eine davon ist die Offenheit der Menschen gegenüber ihrer Umwelt. Sie brauchen z.B. einen offenen Blick auf andere beteiligte Personen und auf das, was im Umfeld geschieht. Nur so können sie Probleme, Risiken, Chancen und sonstige Veränderungen erkennen und danach gegebenenfalls ihr Handeln ausrichten.

Ein anderer, aktiverer Aspekt von Offenheit ist die Informationstransparenz. Der Forderung nach Offenheit im Umgang mit Informationen liegt die Erkenntnis zugrunde, dass ein Mensch nur dann in der Lage ist, eine fundierte Entscheidung zu treffen, wenn er Zugang zu allen relevanten Informationen hat. Je einfacher diese Informationen zugänglich sind und je offener sie präsentiert werden, desto intensiver werden sie genutzt. Ein Beispiel: Anstatt den Status in einem elektronischen Planungswerkzeug zu pflegen, findet man in vielen Scrum-Teamräumen ein Taskboard, an dem die Product

Backlog Items und Tasks des laufenden Sprints als Pappkarten oder Haftnotizzettel in einer der drei Spalten *To Do*, *In Progress* und *Done* hängen. So lässt sich auch für einen Außenstehenden der Status (bezogen auf den laufenden Sprint) auf einen Blick erkennen. Ein anderes Beispiel: Hindernisse, die das Scrum-Team in seiner Arbeit beeinträchtigen, können in einer offen geführten Liste, einem Impediment Backlog, sichtbar gemacht werden. Es ist Aufgabe des Scrum Master, für die Beseitigung dieser Hindernisse zu sorgen. Dass die Liste für alle einsehbar ist, eröffnet auch anderen Personen die Möglichkeit, ein Hindernis aus dem Weg zu räumen. Man kann schließlich nur die Probleme lösen, von deren Existenz man weiß.

Die Offenheit im Umgang mit Informationen hat eine Kehrseite, die wir jedoch für eine Stärke halten: Transparenz wirft Fragen auf. Wenn der Fortschritt für alle sichtbar in einem Burndown Chart entlang einer Ideallinie dokumentiert wird, sind Abweichungen – nach oben wie auch nach unten – sofort erkennbar. In einem Unternehmen, das nur gut laufende Projekte kennt (weil es die weniger gut laufenden entsprechend »frisiert«), kann ein Scrum-Team unangenehm auffallen. Es wird sich von traditionellen Projektleitern die Frage gefallen lassen müssen, warum es im Vergleich zu den traditionellen Projekten so schlecht läuft. Und das Management könnte sich darüber beschweren, dass das Scrum-Team das erste Team in der Geschichte des Unternehmens sei, das nicht ständig im grünen Bereich ist. Dabei wissen die Projektleiter, dass auch in ihren Projekten nicht alles glänzt. Sie haben aber immer bis zur nächsten Lenkungsausschusssitzung (oder einem vergleichbaren Projektstatusmeeting) Zeit, zwischenzeitliche Abweichungen wieder einzufangen, weil bis dahin niemand den aktuellen Projektstatus erfragt. Anders das Scrum-Team: Es ist tagesgenau auskunftsfähig – nicht etwa, um dem Management zu gefallen, sondern in erster Linie, weil das Scrum-Team selbst daran interessiert ist, zu wissen, wie es um den eigenen Fortschritt steht. Je früher es eine Fehlentwicklung erkennt, desto effektiver kann das Scrum-Team Gegenmaßnahmen ergreifen. Das funktioniert am besten in einer Unternehmenskultur, die Änderungen willkommen heißt und Fehler als Chance zum Lernen begreift. Wer nur positive Statusberichte bekommen möchte, muss akzeptieren, dass er betrogen wird. Und

wer nicht am tatsächlichen Zustand seiner Projekte interessiert ist, der wird sich wundern, warum trotz vermeintlich erfolgreicher Projekte der Erfolg auf der Strecke bleibt. Transparenz wirft Fragen auf – aber Transparenz macht vor allem Probleme frühzeitig sichtbar und somit behebbar. Das ist die wahre Stärke von Scrum und anderen agilen Methoden.

Respekt (Respect)

»Die Mitglieder von Scrum-Teams respektieren sich gegenseitig als fähige, unabhängige Personen und werden als solche auch von den Menschen, mit denen sie zusammenarbeiten, respektiert.«

Ein respektvoller Umgang im privaten wie im beruflichen Leben sollte eigentlich selbstverständlich sein. Die Wirklichkeit sieht oft anders aus. Da stehen Programmierhelden im Rampenlicht, die den Rest des Teams im Schatten stehen lassen. Wir erleben paarweises Programmieren, bei dem ein Entwickler seine überlegenen Programmierkünste zum Besten gibt, anstatt seinem Partner zu helfen, besser zu werden. Und natürlich gibt es die vielen kleinen und größeren versteckten und offenen »Nickeligkeiten« – genau wie im Fußball, der diesen Begriff populär gemacht hat.

In Scrum-Teams wird sehr eng und intensiv miteinander gearbeitet. Das funktioniert nur, wenn sich die Teammitglieder gegenseitig respektieren, ihre Verschiedenheit akzeptieren, die Stärken kennen und nutzen, die Schwächen verzeihen oder gemeinsam daran arbeiten und auf ein gemeinsames Ziel hinarbeiten. Daher wird ein Scrum-Team ohne gegenseitigen Respekt keine Erfolge erzielen können.

Viele Developer äußern den Wunsch nach einem homogenen Team, in dem alle ähnlich »ticken«. Dabei steckt oft in der Verschiedenheit der Teammitglieder der Schlüssel zum Erfolg. Scrum-Teams sind interdisziplinär: Sie vereinen alle Kompetenzen, die erforderlich sind, um die anstehenden Aufgaben ohne Unterstützung von außerhalb zu erledigen. In Scrum-Teams treffen leidenschaftliche Programmiererinnen auf begeisterte Tester, kreative Webdesigner auf pfiffige Systemadministratorinnen, großartige Grafikerinnen auf visionäre

Product Owner. Nur gemeinsam können sie ihr Ziel erreichen und ein Produkt entwickeln, das der Kundschaft gefällt.

Idealerweise bringen die Teammitglieder neben ihrem Expertenwissen ein Interesse an den Spezialgebieten der anderen mit. Sie wollen voneinander lernen und ihren eigenen Horizont erweitern – nicht nur aus Neugierde, sondern auch, um die anderen Teammitglieder unterstützen zu können. Solche Persönlichkeiten werden T-förmig (T-shaped) genannt, weil sie über ein breites Basiswissen (waagerechter T-Strich) und zugleich über ausgezeichnete Expertise auf (mindestens) einem Gebiet verfügen (senkrechter T-Strich). Zunehmend finden sich auch Pi-förmige (Pi-shaped) bis Kamm-förmige (Comb-shaped) Persönlichkeiten, die sich durch mehr als nur ein Spezialgebiet bei dennoch breitem Basiswissen auszeichnen.

Ein schönes Beispiel dafür, wie man respektvoll miteinander umgeht, liefert Norman L. Kerth, wenn er Scrum-Teams seine »Retrospective Prime Directive« als Leitlinie für Teamretrospektiven empfiehlt ([Koschek 2014], nach [Kerth 2001]):

»Ganz egal, was wir entdecken werden: Wir verstehen und glauben zutiefst, dass jede(r) nach besten Kräften gearbeitet hat, wenn man den aktuellen Wissensstand, die Fähigkeiten und Fertigkeiten, die verfügbaren Ressourcen und die derzeitige Situation zugrunde legt.«

Teams, deren Mitglieder diese Grundeinstellung teilen, sind in der Lage, ihr Bestes zu geben und dabei gut miteinander auszukommen. Ein gutes Team ist deshalb mehr als die Summe seiner Mitglieder.

Mut (Courage)

»Die Mitglieder des Scrum-Teams haben den Mut, das Richtige zu tun: an schwierigen Problemen zu arbeiten.«

Mut im beruflichen Kontext? Was auf den ersten Blick unpassend erscheinen mag, entpuppt sich bei näherer Betrachtung als wichtiger Wert in einem kollaborativen Umfeld. Je intensiver Menschen zusammenarbeiten, desto größer ist der Einfluss zwischenmenschlicher Befindlichkeiten. Persönliche oder fachliche Differenzen, un-

genaue Kommunikation, übertriebener oder fehlender Ehrgeiz – es gibt viele Gründe dafür, dass die Stimmung in einem Team kippen kann. Solche Situationen aufzulösen und zu neuer Gemeinsamkeit zu finden, erfordert nicht nur Erfahrung und Fingerspitzengefühl, sondern vor allem den Mut, die Probleme offen anzusprechen.

Mut sollte nicht mit Übermut verwechselt werden. Mut ist kalkuliertes Risiko, basierend auf Erfahrung und Urteilsvermögen. Mutige Handlungen haben oft positive Auswirkungen. Übermut hingegen tut, wie schon der Volksmund sagt, selten gut.

Mut bedarf es aber nicht nur zur Auflösung von Konflikten. Wenn ein Teammitglied seine eigene Meinung z.B. zur Softwarearchitektur oder einer Prozessverbesserung konsequent vertritt, auch gegen den Widerstand anderer Teammitglieder, dann ist das mutig. Ein Scrum Master muss oft allen Mut zusammennehmen, um sein Team vor störenden Einflüssen von außen zu schützen. Und ein Product Owner beweist Mut, wenn er beispielsweise eine fachliche Anforderung eines hochrangigen Stakeholders ablehnt, weil diese mit dem Produkt-Ziel oder der Produktvision nicht vereinbar ist.

Nicht alle Menschen sind von Natur aus mutig, und die meisten können oder wollen es auch gar nicht sein. Unter gewissen Rahmenbedingungen können aber fast alle Menschen über ihren Schatten springen und Wahrheiten offen aussprechen. Zu diesen Rahmenbedingungen zählt der oben beschriebene Respekt als Grundwert genauso wie ein offener Umgang mit Fehlern. Es gibt Organisationen, die es sich zum Ziel gesetzt haben, aus den eigenen Fehlern zu lernen und diese Erkenntnisse allen Mitgliedern der Organisation zur Verfügung zu stellen. Das setzt voraus, dass alle bereit sind, eigene Fehler zuzugeben und gemeinsam mit den Kolleginnen und Kollegen an der Fehlerbehebung zu arbeiten. Wer eine solche Lernkultur leben darf, für den ist es selbstverständlich, mutig über Fehler zu sprechen. In einem solchen Umfeld fällt es leichter, auch andere Missstände anzusprechen, die keine Fehler sind, aber beispielsweise Verschwendung bewirken oder ein großes Risiko für die Organisation darstellen. Mit dieser Grundeinstellung wird sich eine Organisation positiv weiterentwickeln – die Welt gehört den Mutigen!

Das Zusammenspiel von Werten, Prinzipien und Praktiken

Aus den Scrum-Werten können Prinzipien abgeleitet werden, in denen wiederum die Scrum-Praktiken ihren Ursprung haben. Werte, Prinzipien und Praktiken spannen somit den Bogen von den Handlungsmöglichkeiten bis hin zu konkreten Handlungsanweisungen.

In der Scrum-Literatur besteht keine Einigkeit darüber, welche agilen Prinzipien das Scrum-Framework ausmachen. Eine große Rolle spielen mit Sicherheit die zwölf Prinzipien des Agilen Manifests, die wir im Abschnitt »Das Agile Manifest« auf Seite 22 kurz vorgestellt haben. Tobias Mayer beschreibt in seinem Essay »The Soul of Scrum« (in [Mayer 2018]) die aus seiner Sicht wichtigsten inhärenten Prinzipien von Scrum:

- *Fokus (Focus)*: Dieser agile Wert hilft Teams, sich zu jeder Zeit auf die Erledigung genau einer Aufgabe zu konzentrieren. Ein gut zusammengestelltes interdisziplinäres Team vorausgesetzt, steht alles zur Verfügung, was dazu erforderlich ist. Es gibt keinen Konkurrenzkampf um Zeit, Ressourcen und Personal – alle Energie wird zielgerichtet und nutzbringend auf die Erledigung der Aufgabe verwendet.
- *Ausrichtung (Alignment)*: Der regelmäßige persönliche Austausch der Developer mit den Stakeholdern und Anwendern sorgt dafür, dass Missverständnisse vermieden und Fragen schnell geklärt werden, sodass am Ende das Richtige entsteht.
- *Kunstvolles Schaffen (Artful Making)*: In dem gleichnamigen Buch [Austin 2003] ziehen Rob Austin und Lee Devin Parallelen zwischen dem künstlerischen Bereich und der aus ihrer Sicht kreativen Softwareentwicklung. Sie haben vier Qualitäten identifiziert, die ein Softwareentwicklungsteam aufweisen sollte: Loslassen (*Release*) innerhalb eines klar definierten Rahmens, eine Zusammenarbeit (*Collaboration*), die zu einem großen Ganzen (*Ensemble*) führt, das größer ist als die Summe seiner Teile, und eine Haltung wie im (Schau-)Spiel (*Play*), wo die qualitativ hochwertige Darbietung wichtiger ist als die Leistung des Einzelnen.

- *Selbstorganisation (Self-Organization)*: Kleine, interdisziplinäre und selbstorganisierende Teams kümmern sich eigenverantwortlich darum, wie sie die vom Product Owner beschriebenen Anforderungen umsetzen. Die Entscheidungsfreiheit hinsichtlich der Umsetzung beschränkt sich nicht nur auf die Wahl der geeigneten Technologien und Architekturen, sondern umfasst auch die Ausgestaltung des Entwicklungsprozesses. Warum sollte diese Aufgabe jemand anderes übernehmen als jene Menschen, die Experten auf diesen Gebieten sind, weil sie täglich damit arbeiten? In vergangenen Versionen des Scrum Guide wurde diese Arbeitsweise der Developer als »selbstorganisierend« (*self-organizing*) bezeichnet. Mit der stärkeren Fokussierung auf das gesamte Scrum-Team, das den Product Owner und den Scrum Master einschließt, wird nun der Begriff »self-managing« verwendet. Wir empfinden die deutsche Übersetzung »selbstmanagend« als sperrig und nur bedingt zutreffend, weil es weniger um die Verwaltung als vielmehr um die Steuerung der Teamaufgaben geht. Deshalb verwenden wir in diesem Buch – abweichend von der offiziellen deutschen Übersetzung des Scrum Guide – die Begriffe »Selbststeuerung« und »selbststeuernd«.
- *Rhythmus (Rhythm)*: Je größer eine Aufgabe ist, desto schwieriger lässt sich deren Inhalt und Umfang abschätzen. Deshalb verwendet Scrum kleine Zeiteinheiten (Timeboxen) und fügt diese in immer wiederkehrender Folge zusammen. So entsteht ein Rhythmus, der es Unternehmen erlaubt, schnell voranzuschreiten und schnell zu lernen.

Scrum selbst vertraut laut Mayer auf diese fünf Prinzipien: Ausrichtung und Rhythmus sind die Basis des Empirismus (siehe Abschnitt »Scrum ist angewandter Empirismus« auf Seite 18). Rhythmus entsteht durch feste Zeiteinheiten und regelmäßige Synchronisation. Fokus hilft, ein Backlog zu verwalten, und sorgt gemeinsam mit einer klaren Ausrichtung für Nutzen stiftende Sprint-Ziele. Kunstvolle Herstellung und Selbstorganisation schaffen den idealen Rahmen für Teams, um fachliche Anforderungen exzellent zu lösen.

Das Prinzip des Empirismus findet beispielsweise in der Sprint-Retrospektive seine Anwendung. Diese Scrum-Praktik dient dazu, Erkenntnisse aus dem vergangenen Entwicklungszyklus zu gewinnen und in Verbesserungsmaßnahmen für den folgenden Zyklus umzuwandeln. Dazu bedient sich das Team unter anderem historischer Daten aus den vergangenen Sprints. Solche Daten müssen verfügbar sein, erfordern also eine Informationstransparenz, die dem Wert der Offenheit entspricht. Offenheit (Wert) ist also eine Voraussetzung für ein empirisches Vorgehen (Prinzip), beispielsweise in der Sprint-Retrospektive (Praktik) – so funktioniert das Zusammenspiel von Werten, Prinzipien und Praktiken.