iX DEVELOPER

Sommer 2021

14,90 €
Österreich 16,40 €
Schweiz 27.90 CHF
Luxemburg 17,10 €

www.ix.de

Besserer Code

Agil entwickelt, umfassend getestet, mit KI-Hilfe optimiert

Agile

Die passende agile Methode finden

Tools für erfolgreiche verteilte Teamarbeit

Programmiersprachen

Rust: Speichersicherheit ohne Performanceeinbußen

WebAssembly und Rust für Bytecode im Browser

C++20: Weniger Fehler dank klarerem Code

Qualität sichern

KI hilft beim Coden und Testen

Frameworks für Behavior-Driven Testing

Quellcodescanner im Vergleich

GitOps und Containerisierung

Infrastructure as Code aus dem Git-Repository

Robust und performant im Container

Softwarelebenszyklus: Mehr Automatisierung, weniger Fehler

Value Design bringt Software und Ethik in Einklang

B1 Consulting Managed Service & Support

individuell – umfassend – kundenorientiert

Wir unterstützen Sie bei Aufbau und Betrieb Ihrer **Linux-**, **Cloud-** oder **Containerlösung** – egal ob in der **Public Cloud** oder **on-premise**.

Ergänzen Sie mit einem individuellen Support- und Betriebsvertrag der B1 Systems Ihr Team um das Wissen und die Erfahrung unserer über 140 festangestellten Linux- und Open-Source-Experten.

Unser in Deutschland ansässiges Support- und Betriebsteam ist immer für Sie da – mit qualifizierten Reaktionszeiten ab 10 Minuten und Supportzeiten von 8x5 bis 24x7!

B1 SYSTEMS

B1 Systems GmbH - Ihr Linux-Partner

Linux/Open Source Consulting, Training, Managed Service & Support

ROCKOLDING · KÖLN · BERLIN · DRESDEN

www.b1-systems.de · info@b1-systems.de

Es geht immer besser

Das Bessere ist der Feind des Guten. Die bekannte, dem französischen Philosophen Voltaire zugeschriebene Weisheit beschreibt anschaulich das ständige Bestreben nach Fortschritt, wie es sich in allen großen Kulturen der Menschheitsgeschichte findet. Im Arbeitsalltag wie auch im Privaten gilt aber häufiger eher der Spruch „Gut genug ist der Feind des Besseren", denn wir hängen meist am Vertrauten und Gewohnten. Veränderung kostet Überwindung und ist bisweilen schmerzhaft.

Da macht die Softwareentwicklung keine Ausnahme. So lange der Code funktioniert, gibt es keinen akuten Handlungsbedarf, ihn weiter zu verbessern – getreu dem Motto „never touch a running system". Doch jede Anwendung ist nur ein Rädchen in einem größeren Gefüge und meist selbst aus verschiedenen Komponenten unterschiedlichster Herkunft zusammengestellt. Updates sind an der Tagesordnung – und jede Änderung birgt die Gefahr, alles aus dem reibungslosen Takt zu werfen.

Entwicklerinnen und Entwickler sind daher ständig gefordert, ihren Code up to date zu halten. Dabei müssen sie häufig an vielen Fronten gleichzeitig kämpfen, um die wachsende Codebasis lesbar und damit auch wartbar zu halten, Sicherheitslücken zu schließen, Build- und Delivery-Pipelines im Fluss zu halten und nicht zuletzt die Erwartungen der Anwenderinnen und Anwender an die Software immer wieder aufs Neue zufriedenzustellen.

Von Grund auf sauberer und leicht nachvollziehbarer Code, wie ihn die Clean-Code-Prinzipien postulieren, bildet zweifellos eine ideale Ausgangsbasis, die alle weiteren Verbesserungs- und Optimierungsmaßnahmen erleichtert. Kontinuierliches Testen entlang aller Prozessschritte vom Quellcode bis zur Anwendung in Produktion ist aber ebenso unerlässlich wie die individuelle Auswahl der geeignetsten Methoden, Werkzeuge, Programmiersprachen und Frameworks.

Um den Faktor Mensch beherrschbar zu machen und mit ihm verbundene Fehlerquellen zu vermeiden, ist die weitgehende Automatisierung, die auch im Zentrum des GitOps-Paradigmas steht, ein probates Mittel. Auch der gezielte Einsatz künstlicher Intelligenz verspricht Entwicklerinnen und Entwicklern automatisierte Hilfe, die sowohl beim Codeschreiben wertvolle Dienste leisten kann als auch beim anschließenden Testen.

Damit am Ende das Bessere endlich der Freund des Guten wird, möchten wir Ihnen mit diesem Sonderheft einen umfassenden Überblick verschaffen und vielfältige Anregungen liefern, welche Programmiersprachen, Tools, Methoden und bewährte Best Practices den Weg zu besserem Code ebnen. *iX* und *heise Developer* wünschen Ihnen viel Spaß beim Lesen – und beim Clean Coding!

MATTHIAS PARBEL

Agile

Agile Softwareentwicklung hat sich als Stand der Technik weitgehend durchgesetzt – kann aber auch schiefgehen. Eine dem jeweiligen Zweck angepasste Auswahl an Methoden, die richtigen Werkzeuge für verteilte Retrospektiven und solides Schätzen des Aufwands weisen den Weg zum Erfolg.

ab Seite 7

Qualitätssicherung

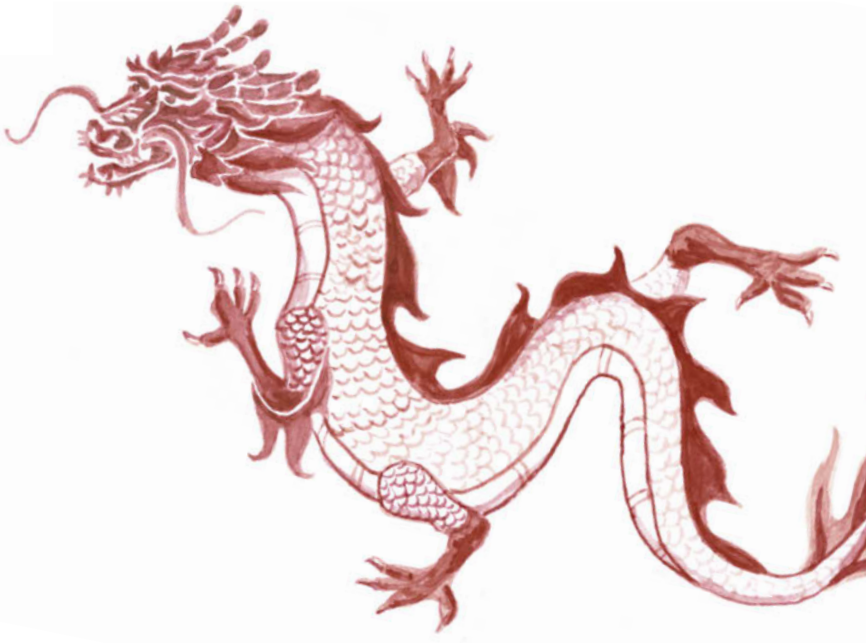
Künstliche Intelligenz unterstützt Entwicklerinnen und Entwickler in immer mehr Bereichen vom Schreiben des Codes bis zu dessen Analyse. Mit KI aufgerüstete Entwicklungsumgebungen und Unit-Testing-Tools erleichtern den Programmieralltag. Automatisierte Qualitätskontrolle gelingt aber auch ohne KI.

ab Seite 31

Programmiersprachen

Im schier unüberschaubaren Programmiersprachenwald tritt Rust an, einiges besser zu machen als etablierte Sprachen, beispielsweise durch sicherere Speicherverwaltung und Codeprüfung zur Kompilierzeit. Mit den Prinzipien des Clean Code lässt sich aber auch C++20 optimieren und auf das Wesentliche reduzieren, um den Code lesbarer zu gestalten und Fehler zu vermeiden.

ab Seite 85



GitOps und Containerisierung

Als neues Paradigma will GitOps die Ära des automatisierten IT-Betriebs einläuten. Von der Infrastructure as Code bis zum vollständigen Lebenszyklus einer Anwendung soll mehr Automatisierung helfen, typische Fehler zu vermeiden, und so zu höherer Qualität beitragen. Das macht ein Kubernetes-Praxisbeispiel aus der Versicherungsbranche ebenso deutlich wie ein Tutorial zur Optimierung containerisierter Java-Anwendungen.

ab Seite 105

Extras

Mit Value Design lassen sich moderne Softwareentwicklung und Ethik in Einklang bringen. In Cloud-nativen Architekturen helfen die aus dem Reactive Manifesto abgeleiteten Prinzipien, die hohen Anforderungen an Elastizität und Resilienz zu erfüllen. Für das Arbeiten mit RESTful APIs existieren viele Ansätze. Wie es mit der Hypertext Application Language geht, zeigt ein Erfahrungsbericht.

ab Seite 129

Halten Sie Ihre Prozesse im Takt

Exklusiver Frühbucher-Rabatt!

heise Academy

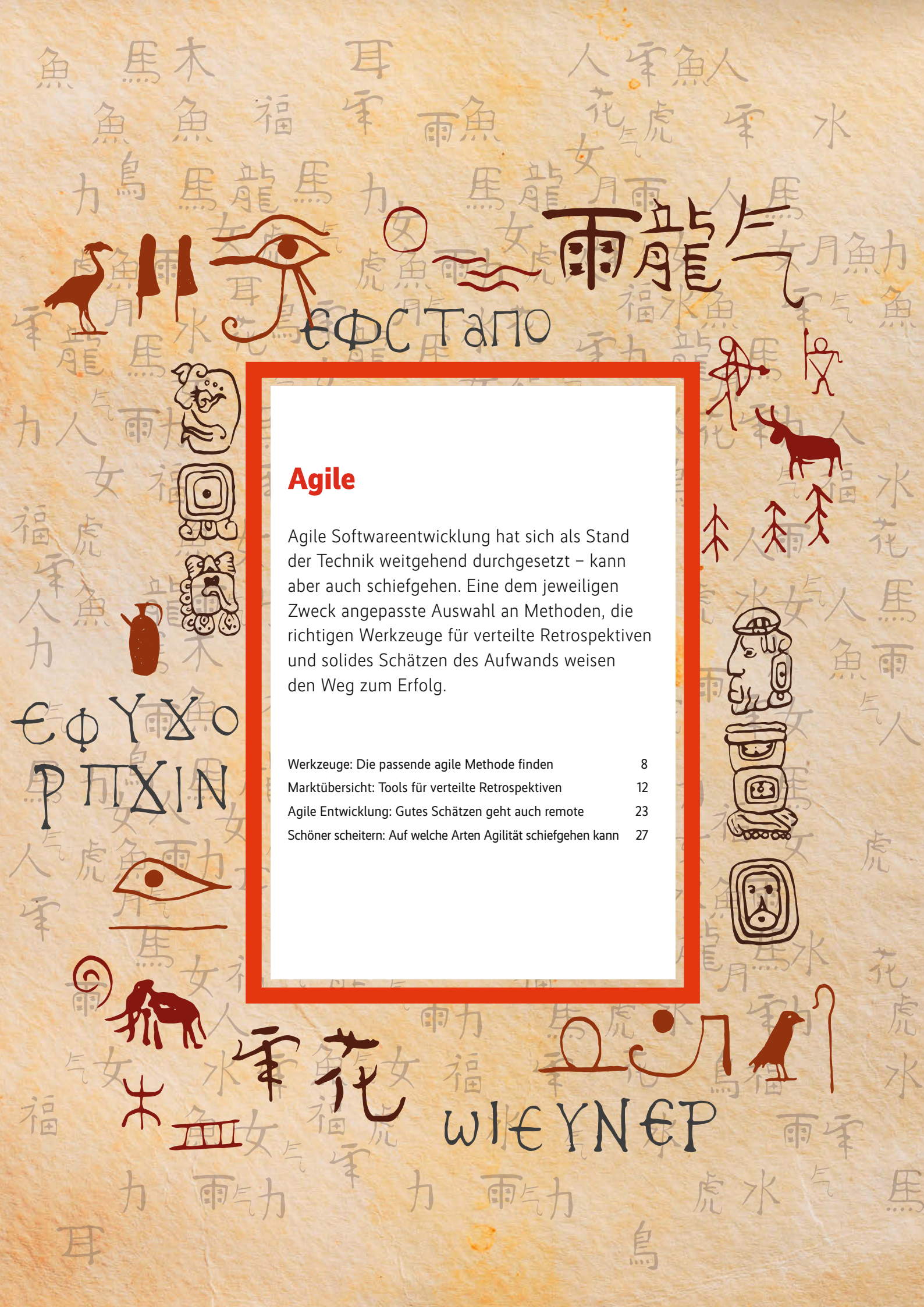
DevOps, Continuous Delivery und Container-Orchestrierung sind der Schlüssel für eine effizientere Softwareentwicklung. Werden Sie mit unseren Workshops und Webinaren zum Master Ihrer Softwareentwicklung.

Unser Angebot

Container-Orchestrierung mit Kubernetes 20. – 23. September WORKSHOP	**M365 für Administratoren** 29. September WEBINAR	**Elastic Stack Fundamentals** 5. – 7. Oktober WORKSHOP
Identiy- & Access-Management für M365 6. Oktober WEBINAR	**Sicheres Single-on mit Kerberos** 11. – 13. Oktober WORKSHOP	**Continous Integration** mit Jenkins 12. – 13. Oktober WORKSHOP

Mehr Informationen:

heise-academy.de/devops

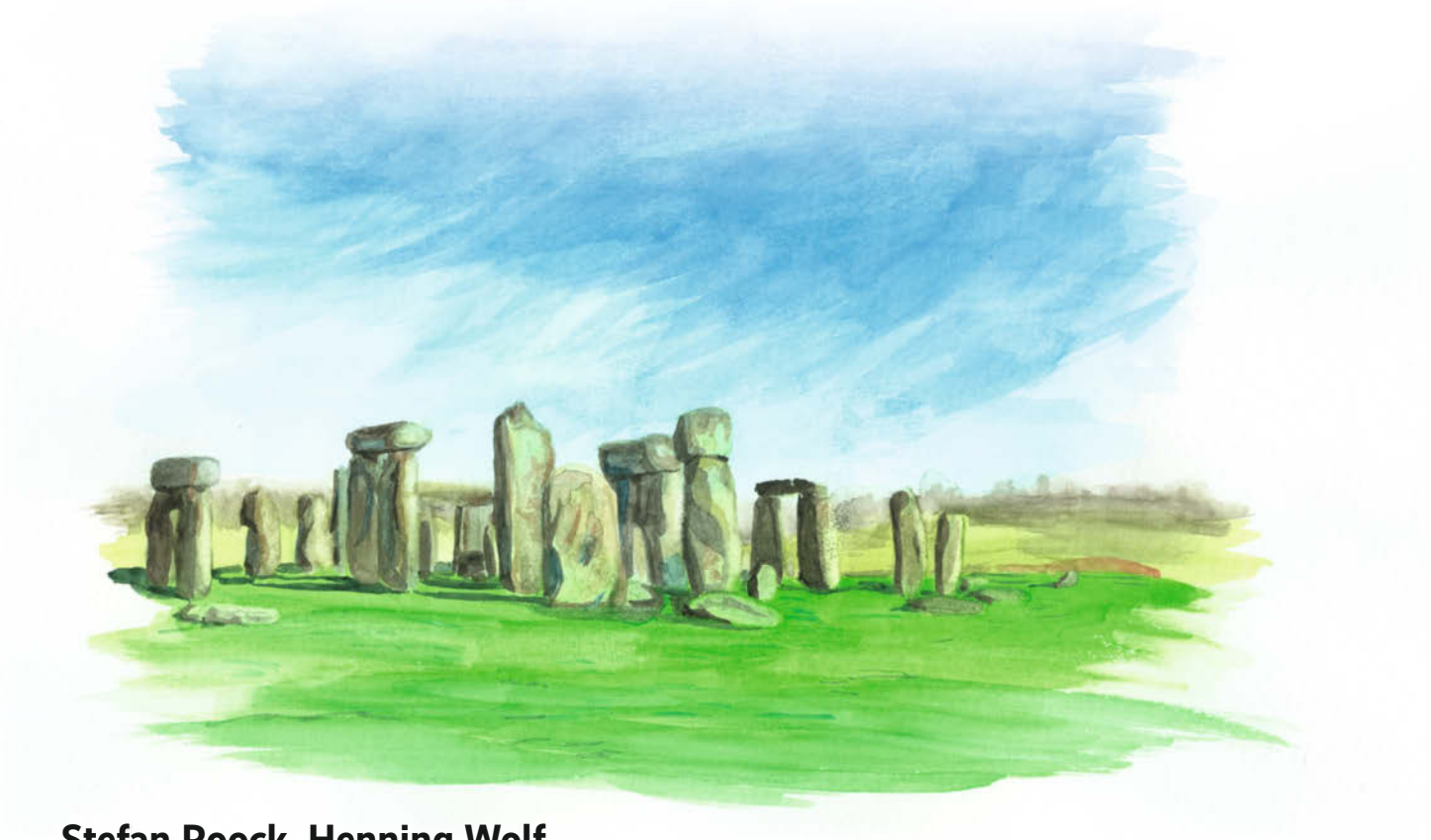


Agile

Agile Softwareentwicklung hat sich als Stand der Technik weitgehend durchgesetzt – kann aber auch schiefgehen. Eine dem jeweiligen Zweck angepasste Auswahl an Methoden, die richtigen Werkzeuge für verteilte Retrospektiven und solides Schätzen des Aufwands weisen den Weg zum Erfolg.

Die passende agile Methode finden

Agil, aber richtig



Stefan Roock, Henning Wolf

Der agile Methodenkasten stellt vielfältige Werkzeuge zur Verfügung. Aber welches passt am besten zur eigenen Situation?

Heute kommt kaum noch ein Unternehmen ohne agile Verfahren aus. Das gilt mindestens für die Softwareentwicklung, immer häufiger aber auch für Bereiche außerhalb der Entwicklung. Außer in Betrieb, Produktmanagement, Marketing und Vertrieb haben wir Agilität auch bei der Entwicklung von Operationstischen, im Callcenter, bei Herrenausstattern und in Arztpraxen gesehen. Genauso vielfältig wie die Einsatzbereiche von Agilität sind die methodischen Ansätze. Soll man auf Scrum oder Kanban setzen? Oder braucht man Extreme Programming? Und was ist eigentlich mit der Skalierung? Dieser Artikel gibt eine Orientierung, was für wen geeignet ist.

Der Kern von Agilität und die „Wunderwaffe“

Was agile Entwicklung ist, ist im agilen Manifest beschrieben (siehe ix.de/zv8z). Die wichtigsten Ideen lassen sich ganz einfach zusammenfassen (siehe Abbildung): Selbstorganisierte Teams überführen die Kundenprobleme in Lösungen. Dabei arbeiten sie in kurzen Iterationen (meist wenige Wochen), in denen sie die Lösung schrittweise verbessern und den Entwicklungsprozess optimieren.Letzteres passiert vor allem über Retrospektiven. In diesen untersucht und bewertet das Team seine Zusammenarbeit untereinander und mit anderen Akteuren und definiert Verbesserungsmaßnahmen. Das wäre auch der Minimaleinstieg in die agile Welt und die Universalantwort auf die Frage, welche agile Technik man auf eine einsame Insel mitnehmen würde, wenn man nur Platz für eine hätte. Schließlich kann man über Retrospektiven den Prozess schrittweise so anpassen, wie es notwendig ist.

Wenn also nichts der folgenden Beschreibungen geht, der Kontext aber nach Lernen, Verbessern, Flexibilität und Agilität schreit, dann können regelmäßige Retrospektiven einen Kristallisationspunkt für Prozessverbesserungen darstellen, mit deren Hilfe ein eigenes agiles Vorgehen über die Zeit entsteht.

Muss es denn immer agil sein?

Bevor man sich der Frage stellt, welche agile Methode die für die aktuelle Situation geeignete wäre, muss man sich darüber klar werden, inwiefern der Kontext nach Agilität schreit. Agilität hilft in Situationen großer Unsicherheit: dort, wo das notwendige Endergebnis nicht von vornherein klar ist, wo mit ganz

neuen Technologien gearbeitet wird und wo die optimale Arbeitsweise nicht vorab bekannt ist. In Kontexten, in denen kaum Unsicherheiten existieren, sind klassische sequenzielle Ansätze besser geeignet. Es ist wichtig, diese unterschiedlichen Kontexte zu verstehen und differenziert zu betrachten. Nur dann wird das „Warum" hinter dem Wunsch nach Agilität klar vermittelbar und es besteht die Chance, dass die Beteiligten sich auf das Abenteuer einlassen. Man schafft mit Agilität hervorragende Voraussetzungen für erfolgreiche Vorhaben, aber auch bei Agilität wird der Anfang steinig sein und ein Umlernen, insbesondere hinsichtlich der Planungsillusion bisheriger Vorgehensweisen, ist für alle Beteiligten nötig.

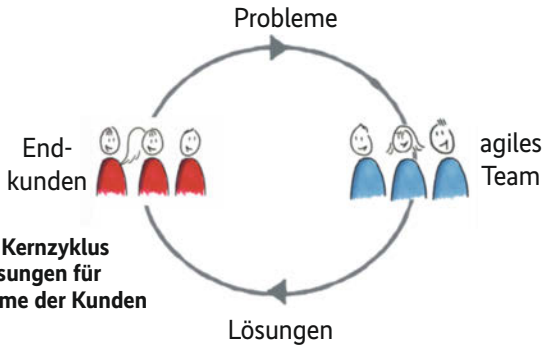
Entwicklung mit Scrum

Die derzeit mit Abstand populärste agile Methode ist Scrum (siehe ix.de/zv8z und [1]), auch wenn Scrum genau genommen nur einen Rahmen (Framework) für die Entwicklung bildet. Aber eben einen Rahmen, der mit seinen Rollen beziehungsweise Zuständigkeiten Product Owner, Scrum Master und Developer, einem Satz an Artefakten und Events und festen Iterationen – in Scrum Sprints genannt – hervorragend geeignet ist für die agile Produktentwicklung.

Scrum funktioniert am besten mit möglichst stabilen Teams, die crossfunktional besetzt sind. Im Optimalfall sind im Scrum-Team alle Qualifikationen vorhanden, die von der Idee bis zum fertigen Produkt benötigt werden. Bei Scrum wird am Ende jedes Sprints, also typischerweise alle zwei bis drei Wochen, im Sprint Review gelernt, welche Aspekte des bereits erstellten Produktinkrements für die Stakeholder nützlich sind und was man gemeinsam daraus für die weitere Produktentwicklung lernen kann. Auch wenn Scrum stets von Produkt spricht, ist es ebenfalls für Projekte einsetzbar.

Universalwerkzeug Kanban

Bei Kanban für Wissensarbeit handelt es sich genau genommen nicht um eine agile Entwicklungsmethode, sondern um



Der agiler Kernzyklus schafft Lösungen für die Probleme der Kunden (Abb. 1).

eine schlanke Change-Methode [2]. Kanban umfasst eine Reihe von Prinzipien und Praktiken, am bekanntesten und sichtbarsten ist die Visualisierung der Arbeit auf einem Kanban-Board.

Kanban kennt keine festen Iterationslängen, begrenzt mit sogenannten WiP-Limits (Work in Progress) die Menge der gleichzeitigen Arbeit und befördert den kontinuierlichen Verbesserungsprozess von und zwischen Teams. Dabei spielt es keine Rolle, ob Admin-Teams, Verwaltungsteams oder Entwicklungsteams mit Kanban arbeiten. Im Vergleich zu Scrum ist der Einstieg leichter, was auch daran liegt, dass eines der Kanban-Prinzipien „Starte da, wo du gerade bist" lautet.

Mehr Softwarequalität mit Extreme Programming

Extreme Programming (XP) ist in seinem Ablauf sehr ähnlich zu Scrum [3]: Ein selbstorganisiertes Team entwickelt in Iterationen nützliche Software für Kunden und passt seinen Prozess über Retrospektiven an.

Darüber hinaus adressiert XP einen Aspekt, der in vielen Scrum-Implementierungen leider vernachlässigt wird: technische Exzellenz. Agile Entwicklung kann auf Dauer nur erfolgreich sein, wenn passende Entwicklungspraktiken verwendet werden. Schließlich erfordert das iterative Vorgehen immer wieder Änderungen am Programmcode. Daher

müssen solche Änderungen langfristig mit wenig Aufwand und geringem Risiko möglich bleiben. XP schlägt dazu Entwicklungspraktiken wie Test-driven Development (TDD), Pair Programming, Continuous Integration (CI) und weitere vor.

Weil die Qualität des entwickelten Programmcodes eine so wichtige Voraussetzung für erfolgreiches agiles Arbeiten ist, gehören unserer Meinung nach die Entwicklungspraktiken aus XP in den Werkzeugkoffer jedes (agilen) Entwicklers, unabhängig davon, mit welcher konkreten Methode gearbeitet wird. XP-Praktiken gehören zu den Hard Skills agiler Entwickler.

Die Verlockung ist in manchen Firmen groß: Da fängt man relativ spät mit der Einführung agiler Methoden an, dafür will man aber gleich die ganz großen Vorhaben mit 100 Leuten agil durchführen. Das kann man machen, es gibt auch agile Skalierungsframeworks dafür, aber unser Tipp lautet: Erst wenn du etwas im Kleinen beherrschst, solltest du darüber nachdenken, wie du es möglichst organisch größer machen kannst. Wenn die Beteiligten Agilität nicht schon erlebt und verstanden haben, gibt es sonst einfach nur sehr großes Chaos.

Quelle: Shutterstock.com/LanKS

Die Methode Scrum eigent sich optimal für die agile Produktentwicklung (Abbildung 2).

Angepasst oder „by the book“?

In der Praxis stellt sich immer wieder die Frage, ob man die Methode der Wahl „by the book“ verwenden, an den eigenen Kontext anpassen oder gar ein komplettes Eigengewächs züchten sollte. Bei dieser Frage hilft der agile Kernzyklus vom Anfang dieses Artikels (s. Abb. 1). Wir begreifen die verschiedenen methodischen Ansätze wie Scrum, Kanban und XP als Hilfsmittel, um den agilen Kernzyklus immer besser umzusetzen: kontinuierliche Lieferung von Wert an Kunden. Der agile Kernzyklus steht im Vordergrund und die konkrete Methodik im Hintergrund.

Ein großes Problem in der Praxis ist, dass man das eigentliche Ziel aus den Augen verliert und die Methodik in den Vordergrund gerät. Der Erfolg einer unternehmensweiten Scrum-Einführung wird dann schon mal an Methodenkonformität gemessen, statt den Fokus auf Wertsteigerung zu legen. Wenn man aber über eine Anpassung an zum Beispiel Scrum nachdenkt, ohne davon überzeugt zu sein, dass dadurch besserer Wert an Kunden geliefert wird, drückt man sich vermutlich gerade vor etwas, was schwierig umzusetzen ist, obwohl hier möglicherweise großer Nutzen zu ziehen wäre.

Nach unserer Erfahrung herrschen in der Praxis die schmerzvermeidenden gegenüber den wertsteigernden Anpassungen vor. Die meisten Unternehmen wären besser beraten, die gewählte Methodik zunächst „by the book“ einzusetzen, um zu erleben, wie es ist, wenn man es so macht, wie es gedacht ist. Einer unserer Kunden brachte es so auf den Punkt: „Man kann Scrum nicht verstehen, ohne es praktiziert zu haben.“ Auf Basis dieses Erlebens kann man dann viel informierter über Anpassungen entscheiden.

Evolution oder Revolution?

In gewisser Weise stellt sich für die meisten Firmen die Frage, wie dringend und massiv der Umstieg sein soll und kann. Wenn man bereit ist für die Revolution mit ganz neuen Rollen und völlig anderen Meetings als bisher, dann sind Scrum und XP gute Kandidaten. Braucht man eher eine bedächtigere Vorgehensweise bei der Einführung, die die eigenen Besonderheiten von vornherein berücksichtigt und den Mitarbeitern nicht zu viel Veränderung auf einmal abverlangt, wird Kanban der nützlichere Kandidat sein.

Sowohl im revolutionären als auch im evolutionären Fall der Einführung agiler Methoden gilt: Entscheidend ist das Dranbleiben am Verbessern, die weitere Evolution nach den ersten Quick Wins. Nach unserer Erfahrung sind viele Unternehmen schon mit den Quick Wins zufrieden und nutzen die wirklich großen Vorteile nicht, die sich erst ergeben, wenn man am Ball bleibt, nach mehr Verbesserung sucht und mutig Neues ausprobiert. Wir glauben, dass am Ende die Firmen erfolgreich werden und bleiben, die am schnellsten lernen und dauerhaft weiterlernen. (ane@ix.de)

Quellen

[1] Stefan Roock, Henning Wolf; Scrum – verstehen und erfolgreich einsetzen; 2. Auflage, dpunkt.verlag 2018
[2] David J. Anderson; Kanban: Evolutionäres Change Management für IT-Organisationen; dpunkt.verlag 2011
[3] Kent Beck, Cynthia Andres; Extreme Programming Explained: Embrace Change; Addison Wesley 2004
[4] Agiles Manifest, Scrum Guide: ix.de/zv8z

Stefan Roock, Henning Wolf

sind seit Ende der 90er-Jahre mit agilen Methoden unterwegs. Es begann mit Extreme Programming, dann kamen Scrum und Kanban dazu. 2005 gründeten sie gemeinsam mit Kollegen die it-agile GmbH und arbeiten dort als Scrum-, Kanban- und Agile-Leadership-Trainer und Managementberater.

VON DER VISION ZUR REALITÄT

ENTWICKELN SIE ZUKUNFT

Mit modernen Technologien zur IT-Lösung.

Als international agierende Unternehmensgruppe mit weltweit mehr als 8.500 Mitarbeitenden bieten wir ausgezeichnete Karrierechancen in der Softwareentwicklung und IT-Beratung. Wir unterstützen Sie kontinuierlich beim Ausbau Ihrer Qualifikationen. Denn unser gemeinsamer Erfolg ist die Basis Ihres persönlichen Fortschritts. Überzeugen Sie sich selbst. Steigen Sie ein bei msg und zeigen Sie uns, was Sie können!

Beste Arbeitgeber™ Deutschland Great Place To Work. 2020

.denken .gestalten .wachsen

.msg

Tools für verteilte Retrospektiven

Verteilter Rückblick

Stefan Mintert

An verschiedenen Standorten arbeitende Teams gibt es nicht erst seit der Coronapandemie. Zahlreiche Tools unterstützen Scrum Master, agile Coaches oder Teamleiter bei der verteilten Teamentwicklung. Hier liegt der Fokus auf der Retrospektive.

Die Retrospektive ist eine zentrale Scrum-Veranstaltung. Auch außerhalb der agilen Arbeitswelt findet sich der regelmäßige Rückblick. Beispielsweise gibt es bei der US Army den Begriff After Action Review (AAR). Hier wie dort geht es darum, zurückliegende Teammaßnahmen zu analysieren und Verbesserungen für zukünftige Handlungen zu identifizieren. In der Wirtschaft kennt man kontinuierliche Verbesserungsmaßnahmen spätestens seit den 1950er-Jahren, als Begriffe wie Shewhart Cycle, Deming Cycle oder PDCA (Plan, Do, Check, Act) an Bedeutung gewannen.

Im Bereich der agilen Entwicklung schreiben verschiedene Quellen Scrum den größten Anteil zu und sehen einen Marktanteil von 72 Prozent [1] (siehe ix.de/zvr6). Aus diesem Grund orientieren sich die folgenden Ausführungen an Scrum, seiner Terminologie und seinen Veranstaltungen, ohne Scrum als maßgeblich anzusehen.

Für die Durchführung von Retrospektiven gibt es eine vergleichsweise große Zahl an Onlinetools. Im Gegensatz dazu scheinen die anderen Scrum-Events in einem verteilten Szenario einfach durchführbar zu sein; andernfalls könnte man ein größeres Toolangebot auch für andere Events erwarten. Ein kurzer Blick auf die Veranstaltungen soll das unterstreichen.

In einem Daily Scrum oder Stand-up informieren sich die Teammitglieder über die aktuelle Arbeit, etwaige Hindernisse und über Unterstützungsbedarf. Sie sprechen auch die Arbeit der nächsten 24 Stunden ab und prüfen, ob sie im Hinblick auf das Sprint-Ziel auf dem richtigen Kurs sind.

Ob dabei neben einer Software für Videokonferenzen ein weiteres Tool erforderlich ist, hängt vom individuellen Bedarf des Teams ab. Teams, die an verteilten Standorten arbeiten, verwalten ihre Arbeitspakete meist in einem Ticketsystem wie Jira, Redmine oder Trello. Diese Tools haben auch ohne verteilte Standorte ihre Berechtigung, sodass man sie nicht als spezielle Software für verteiltes Arbeiten bezeichnen kann. Da es sich um Onlinetools handelt, ist ein Blick auf Tickets und gegebenenfalls ein Board wie Kanban von jedem Ort der Welt problemlos möglich. Es funktioniert genauso einfach wie bei der Zusammenarbeit an einem Ort. Wenn also ein Stand-up vor Ort ohne besondere Software auskommt, stellt sich die Frage, was ein verteiltes Team mehr braucht als Video-Calls und Ticketsystem.

Das Gleiche gilt für die Sprint-Planung. Wesentlich ist hier der Umgang mit den Tickets. Software für Video-Calls, Screensharing und ein Ticketsystem dürften den meisten Teams gute Dienste leisten und nichts vermissen lassen.

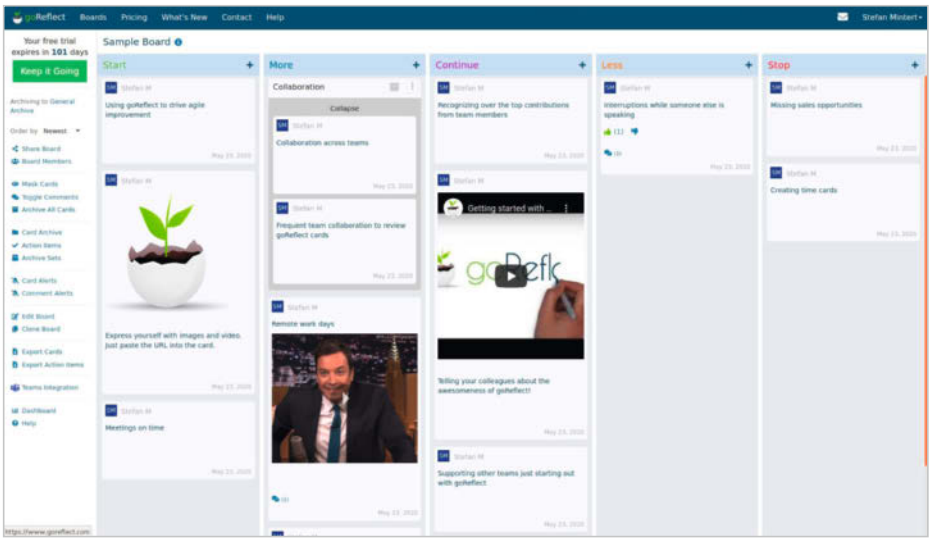
Im Sprint-Review stehen das Produkt und der Fortschritt im Mittelpunkt. Einerseits ist dabei kaum eine besondere Toolunterstützung notwendig, andererseits ist es nicht leicht, ein allgemein nützliches Werkzeug zu entwerfen, weil die agil entwickelten Produkte zu verschieden sind.

Best Practices für die Retrospektive?

Für Scrum Master finden sich einige Medien, die Hilfestellung bei der Vorbereitung einer guten Retrospektive leisten. Neben Büchern [2–4] steht mit retromat.org auch ein Onlinetool für die Auswahl von Retrospektivenbausteinen zur Verfügung.

Viele Elemente setzen darauf, dass sich das Team zur Retrospektive in einem Raum zusammenfindet. Für manche Übungen ist die physische Nähe zwingend erforderlich, zum Beispiel: Die Teammitglieder sollen sich im Raum mit so viel Abstand verteilen, wie es der Menge an Zusammenarbeit entspricht. Haben zwei Personen viel zusammengearbeitet, stehen sie nah zusammen, haben sie wenig zusammengearbeitet, stehen die beiden Personen weiter voneinander entfernt. Man kann die Übung aber für ein verteiltes Szenario abwandeln, indem man die Teammitglieder ihre Avatare auf einem Onlineboard platzieren lässt.

Die meisten Übungen gehen davon aus, dass Flipcharts, Stifte, Moderationskarten und Ähnliches zum Einsatz kommen. Die persönliche Interaktion und die Verwendung von physischen Hilfsmitteln kann durchaus eine Rolle spielen; beides steht im verteilten Setting nicht zur Verfügung. Für die genannten Materialien stellt sich die Frage: Welches Tool tritt an die Stelle des üblichen Moderationsmaterials? Bereits einfache Tools können gute Hilfe leisten. So lässt sich etwa mit



Ein Board von goReflect besteht aus den fünf Spalten Start, Stop, More, Less und Continue und einigen Karten (Abb. 1).

Google Presentations ein Whiteboard simulieren, auf dem jedes Teammitglied schreiben und zeichnen kann. Wenn das zu umständlich, zu aufwendig oder aus anderen Gründen ungeeignet ist, kommen Onlinetools für verteilte Retrospektiven ins Spiel.

Die im Folgenden genannten Tools lassen sich in zwei Gruppen zusammenfassen. In beiden Gruppen sammeln die Teams zu Beginn der Retrospektive ihre Themen und stellen sie auf einem Board dar. Alle Mitglieder schreiben ihre Themen auf „Moderationskarten". Manchmal sind diese Karten visuell als Karten dargestellt, manchmal als Klebezettel, manchmal als einfache Rechtecke.

Die erste Gruppe der Tools ordnet die Karten in Spalten an. Das Board ähnelt dem, was landläufig als Kanban-Board bezeichnet wird, auch wenn es hier natürlich nicht um Kanban als Methode geht. Die Boarddarstellung in dieser Gruppe kennt man aus Ticketsystemen wie Jira oder Trello. Im Folgenden heißt dieser Typ „Spaltenboard".

Die zweite Gruppe verwendet ein Board, das eher mit einem Whiteboard oder Flipchart zu vergleichen ist. Die Teilnehmer können ihre Beiträge auf Klebezetteln an jede beliebige Stelle auf dem Board kleben und frei verschieben. Im Folgenden heißt dieser Typ „Whiteboard".

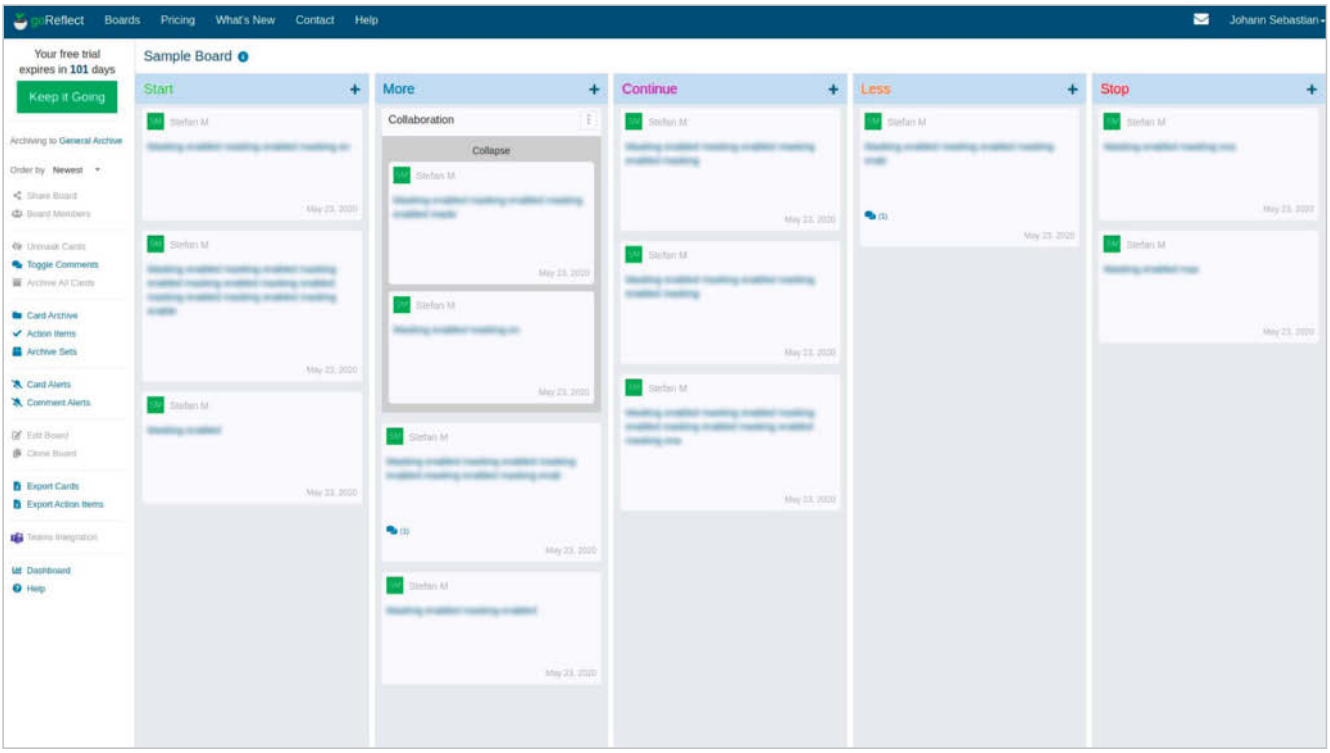
Zu den Vertretern der Spaltenboards gehören die Tools goReflect, FunRetro, ScatterSpoke, TeamRetro, Retrium, Parabol, Retrospected, Scrumlr, RemoteRetro und Team O'clock. Whiteboards benutzen Stormboard und Metro Retro. Echometer geht einen eigenen Weg und sammelt Retrobeiträge in einer Formularumfrage ein.

Spaltenboards

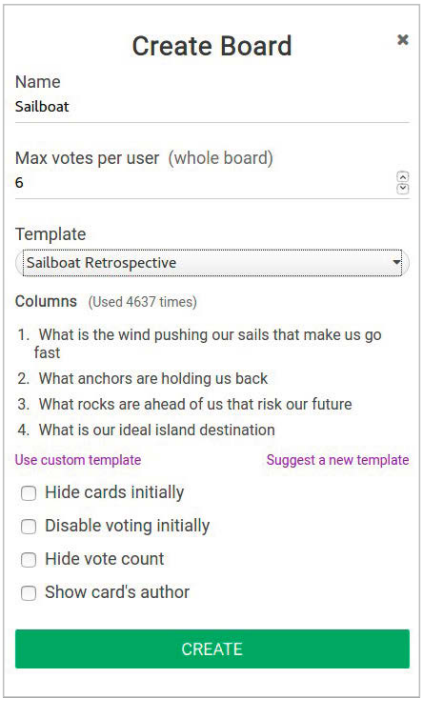
Der erste Vertreter eines Spaltenboards ist goReflect. Ohne das Tool besonders hervorheben zu wollen, gehen die folgenden Ausführungen mehr ins Detail als bei späteren Produkten. Damit ist keine Wertung verbunden. Vielmehr geht es darum, den typischen Ablauf einmal vorzustellen.

goReflect

Die Teammitglieder schreiben ihre Beiträge auf Karten. Die Karten sind in Spalten unter frei wählbaren Überschriften angeordnet. Abbildung 1 zeigt das Beispielboard von goReflect.



Ist das goReflect-Board maskiert, sind die Karteninhalte für die Teilnehmer nicht lesbar (Abb. 2).



Wählt man ein Boardtemplate in FunRetro aus, zeigt die Software die Spaltenbezeichnungen vorab an (Abb. 3).

Es verwendet die Überschriften Start, Stop, More, Less und Continue. Damit folgt es einem bekannten Schema für Retrospektiven. Die anderen von goReflect angebotenen Boardtypen unterscheiden sich in der Zahl und in der Bezeichnung der Überschriften.

Karten enthalten Text und können per URL angegebene multimediale Inhalte darstellen. Die meisten anderen Tools erlauben ausschließlich textuelle Inhalte. Der Moderator kann Karten „maskieren". Dabei schaltet das Tool einen Filter über die Karten, sodass die Inhalte nicht mehr lesbar sind. Auf diese Weise können die Teilnehmer ihre Meinungen zunächst unbeeinflusst voneinander abgeben, bevor Moderatoren alle Beiträge gleichzeitig lesbar schalten (s. Abb. 2). Auch diese Funktion ist bei den anderen Produkten so oder ähnlich vorhanden.

Sind die Inhalte lesbar, kann jeder Teilnehmer die Beiträge kommentieren und mit Likes/Dislikes versehen. Karten lassen sich gruppieren, um zusammenhängende Themen darzustellen. Moderatoren können auf den Karten eine Stimmabgabe erlauben. Die Zahl der verfügbaren Stimmen je Teilnehmer ist für das Board konfigurierbar. Dadurch unterscheiden sich Stimmen von Likes. Letztere kann man auf beliebig vielen Karten abgeben, je Karte jedoch nur maximal einmal. Bei einer Stimmabgabe kann man auch mehrere Stimmen auf eine Karte geben, um Themen höher zu gewichten.

Stimmen gibt man für eine Karte ab, nicht für die zugehörige Gruppe. Für die spätere Sortierung addiert goReflect die Stimmen aller Karten einer Gruppe und sortiert die Gruppe entsprechend. Die Details der Stimmabgabe weichen bei anderen Produkten ab. Dazu ein Beispiel: Die Teilnehmer schreiben die Karten „Zusammenarbeit verbessern", „Bessere Kollaboration" und „Mehr Teamwork". Im Gespräch stellt sich heraus, dass die drei Karten ein und dasselbe Thema bezeichnen. Der Moderator gruppiert die drei Karten und bezeichnet die Gruppe mit „Zusammenarbeit". Daneben gibt es eine weitere Karte mit dem Text „Mehr Zeit für Meetings". Bei der anschließenden Stimmabgabe erhalten die ersten drei Karten je zwei Stimmen, die letzte Karte bekommt vier Stimmen. Sortiert man die Karten nach erhaltenen Stimmen, steht die Gruppe „Zusammenarbeit" mit sechs Stimmen oben, auch wenn jede Karte der Gruppe nur zwei Stimmen bekommen hat.

Bei der Usability lässt goReflect an einigen Stellen zu wünschen übrig. So nimmt das Tool die Sortierung nur in dem Moment vor, in dem man die Funktion auswählt. Das Board aktualisiert sich danach nicht selbst.

Wenn das Team in der Diskussion Verbesserungsmaßnahmen gefunden hat, lassen sie sich als Maßnahmen („Action Item") aufnehmen. Zu den Action Items gelangt man über das Menü auf der linken Seite. Ein Action Item besteht aus einem Titel und einer optionalen Beschreibung. Man kann jedes Item einem Teammitglied zuweisen. Ein Workflowstatus (Not started, In progress, Completed, Blocked) zeigt den Fortschritt an, und ein Datum gibt vor, wann die Aufgabe erledigt sein muss.

Action Items und Karten lassen sich als Excel-Datei exportieren. Darüber hinaus besitzt goReflect eine Integration mit Microsoft Teams. Neue Karten und Kommentare lösen eine Benachrichtigung in Teams aus. Auf Wunsch postet goReflect auch den Inhalt in den gewählten Teams-Kanal. Die Funktion dürfte für eine am Sprintende durchgeführte Retrospektive nicht sehr interessant sein. Sammelt man Beiträge jedoch kontinuierlich, um sie zeitnah zu besprechen, kann eine Benachrichtung im Teams-Chat nützlich sein.

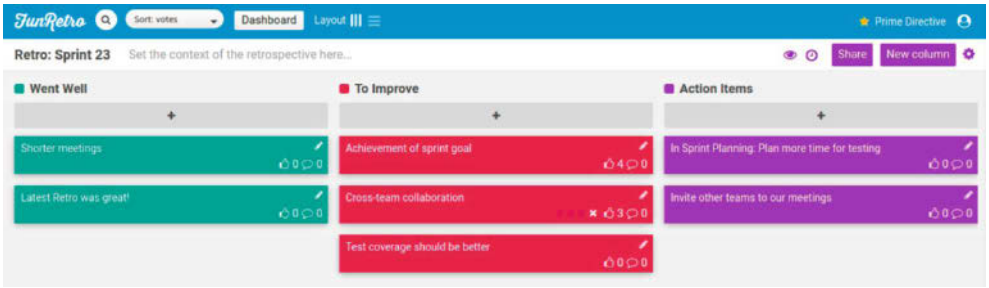
FunRetro

Die Eigenschaften eines FunRetro-Boards legen Moderatoren bei der Erzeugung eines Boards fest. Dazu gehören die Anzahl der Stimmen, ob eine Stimmabgabe überhaupt möglich ist, die Auswahl, ob Karten zunächst unlesbar sind, und ob Autoren einer Karte sichtbar sind. FunRetro besitzt ein Freemium-Preismodell. In der kostenlosen Stufe kann man Teilnehmer über einen Link einladen. In diesem Fall sind die Teilnehmer automatisch anonym.

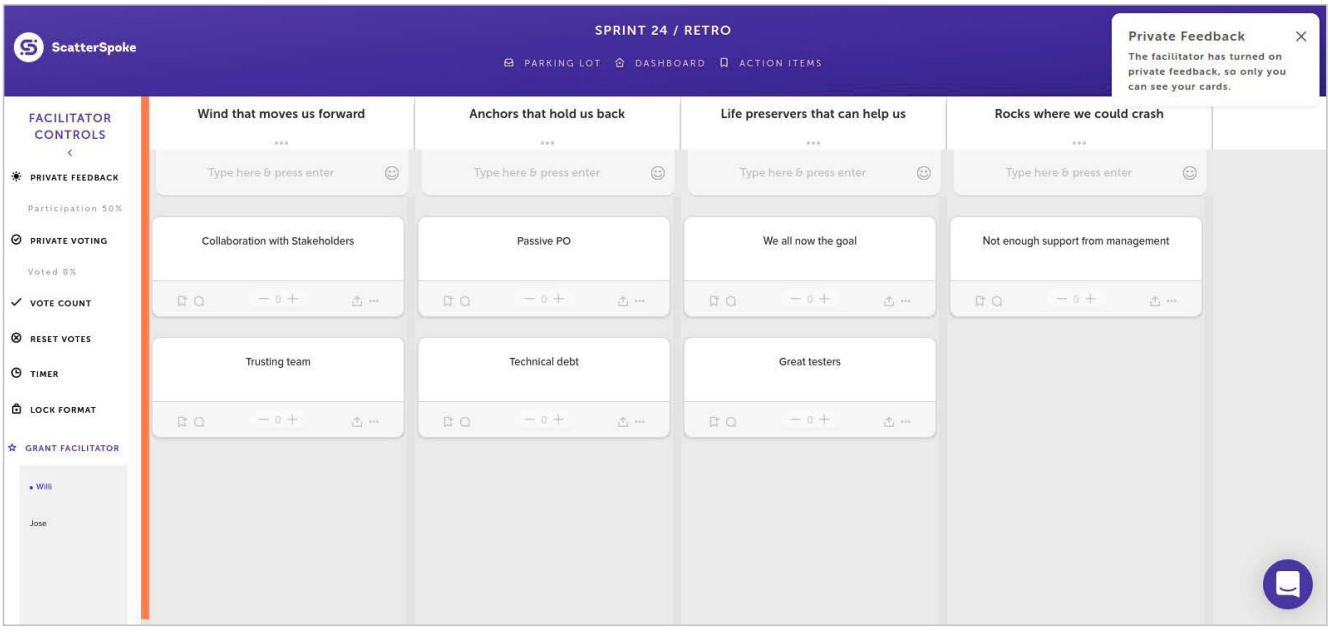
Beim Anlegen eines Boards ist die große Auswahl vordefinierter Templates auffällig. Neben 16 Boardtypen, die der FunRetro-Entwickler anbietet, stammen mehr als 100 Templates aus der Benutzercommunity.

Die Boardtypen unterscheiden sich in Anzahl und Benennung der Spalten. Wählt man im Dialog ein Template aus, sieht man die Spaltenbezeichnungen, bevor man das Board anlegt (s. Abb. 3). So kann man sich bei der Auswahl orientieren. Nebenbei erfährt man noch, wie oft ein Template bereits verwendet wurde. Ein neues Boardlayout kann man den Entwicklern an gleicher Stelle vorschlagen.

Die Arbeit mit FunRetro verläuft weitgehend analog zur Verwendung von goReflect. Action Items sind hier allerdings



Action Items bilden in FunRetro keine eigenen Objekte, sondern einen Kartentyp wie hier im dreispaltigen Board (Abb. 4).



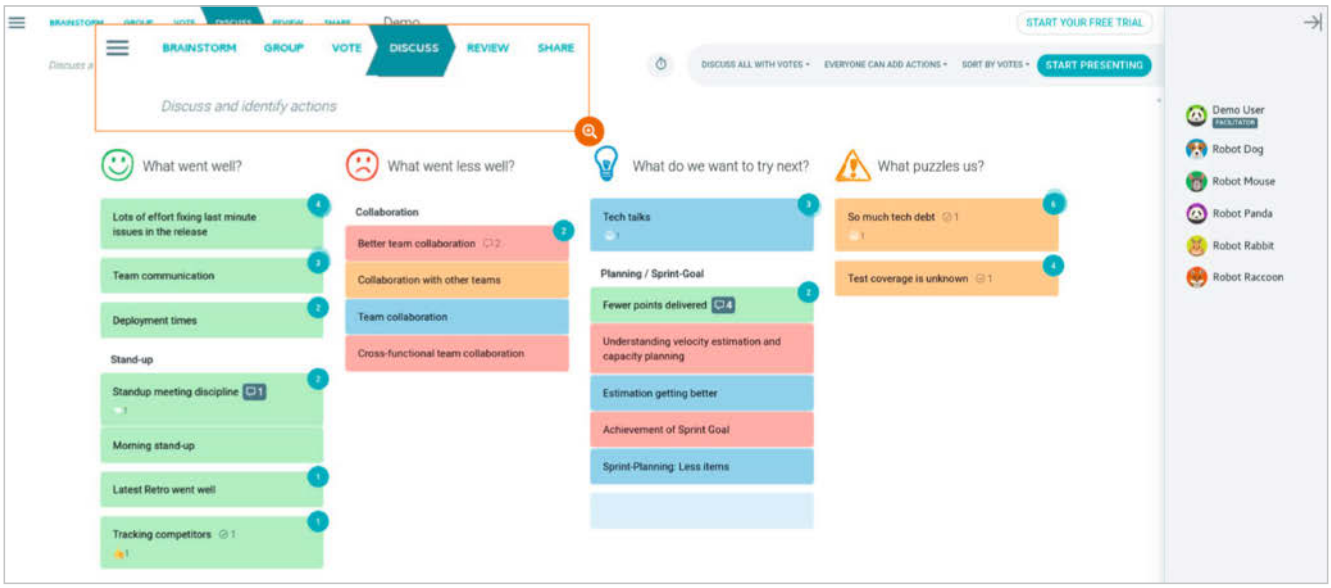
Treten in einer Retro mit ScatterSpoke Themen auf, die außerhalb des Teams zu besprechen sind, kann man sie gleich auf den Themenparkplatz der teamübergreifenden Retro verschieben (Abb. 5).

keine eigenen Objekte, sondern müssen als Spalten und damit als Kartentyp abgebildet werden (s. Abb. 4). Damit steht FunRetro allein da. Für den Export eines Boards stehen die Formate PDF, CSV und PNG zur Verfügung. Die Funktion ist zahlenden Nutzern vorbehalten. Man kann Karten aus einer CSV-Datei importieren.

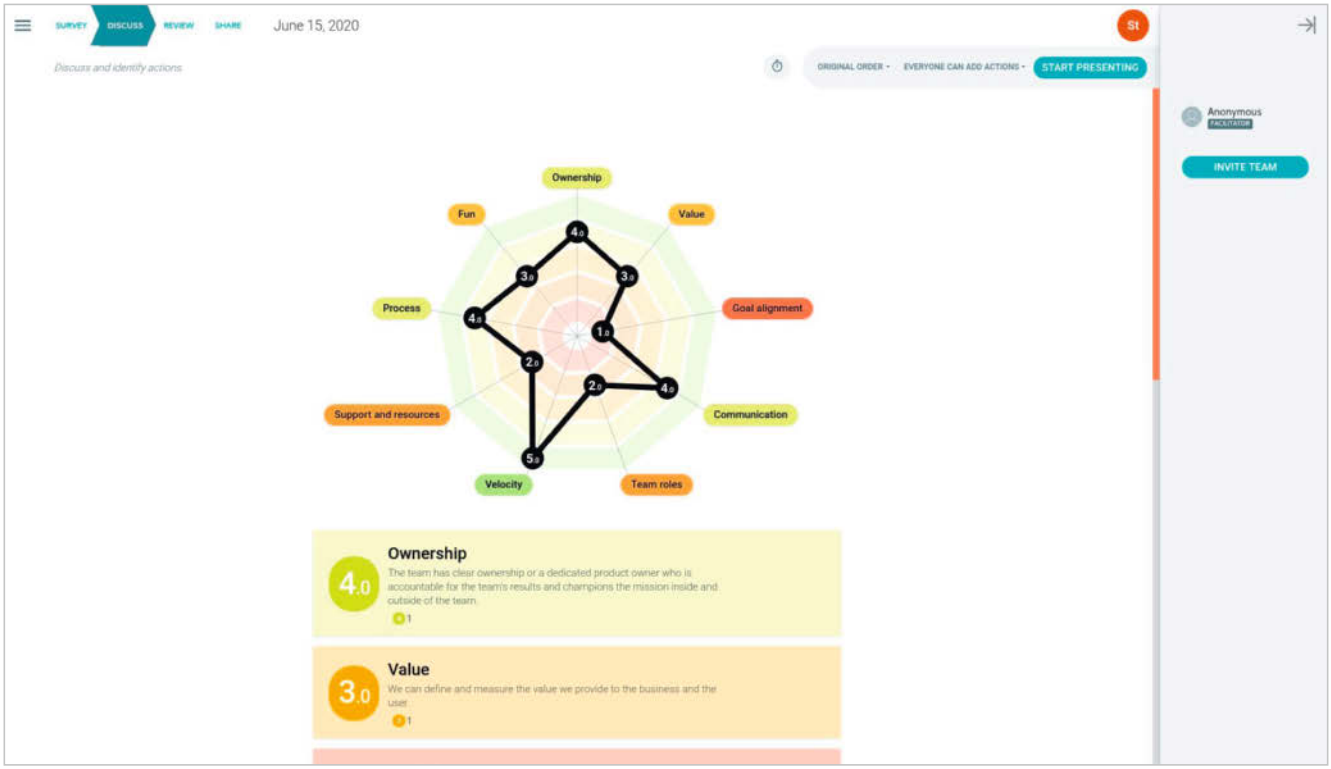
ScatterSpoke

Auch ScatterSpoke arbeitet mit Spaltenboard, Karten, Gruppierung und Stimmabgabe (s. Abb. 5). Für Beiträge, die man für einen späteren Zeitpunkt zurücklegen möchte, gibt es einen Bereich namens Parkplatz. Das ist zunächst vielleicht nicht bemerkenswert. Mit den anderen Tools könnte man sich einen Parkplatz als eigene Spalte definieren, Karten und Board nach einer Retro archivieren und bei der nächsten Retro aus dem Archiv holen. Eine besondere Bedeutung kommt dem Themenparkplatz bei einem einzigartigen Feature von ScatterSpoke zu.

ScatterSpoke kennt das Konzept von skalierten Retros. Dahinter steht eine Gruppe von mehreren Teams, die ScatterSpoke als „Program“ bezeichnet. Tauchen in einer Teamretro Hindernisse oder sonstige Beiträge auf, die nicht unmittelbar auf der Teamebene zu klären sind, können sie aus der laufenden ScatterSpoke-Sitzung heraus auf den Parkplatz der Program-Retro gelegt werden. Führt man später eine Program-Retro mit allen beteiligten Teams durch, finden sich die Karten dort wieder, und man kann sie auf der teamübergreifenden Ebene besprechen.



TeamRetro mit gruppierten, kommentierten Karten nach der Abstimmung. Oben links zeigt das Tool an, in welchem Ablaufschritt sich die Retro gerade befindet (Abb. 6).



Wie ist es um die Teamgesundheit bestellt? Das Teamradar stellt die Meinungen der Mitglieder dar (Abb. 7).

Die Jira-Cloud-Integration von ScatterSpoke erlaubt es, Karten ins Backlog zu exportieren. Das Gleiche ist auch mit Trello möglich.

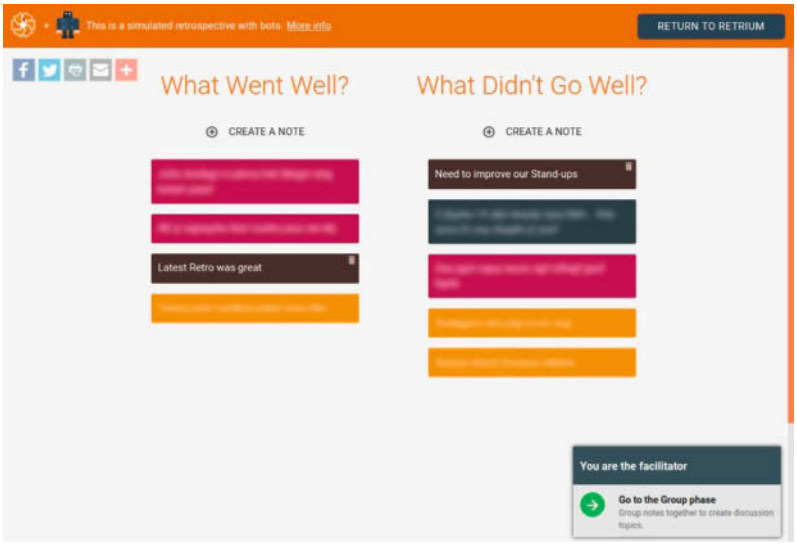
TeamRetro

Der nächste Vertreter des bekannten Spaltenschemas ist TeamRetro. Auf einem Board lassen sich Beiträge aller Retroteilnehmer sammeln, gruppieren und abstimmen. Kommentare hängen an den Karten.

Ein auffälliger Unterschied zu den bisherigen Produkten besteht darin, dass TeamRetro, wie auch die meisten der folgenden Tools, einen Workflow vorgibt. Er macht den Einstieg in die Arbeit mit dem Werkzeug sehr leicht. Die Schritte des Workflows lauten: Brainstorm, Group, Vote, Discuss, Review, Share (s. Abb. 6). Dahinter stehen folgende Aufgaben: Im Schritt „Brainstorm" sammelt das Team die Beiträge in den durch das Boardtemplate vorgegebenen Kategorien. Die Karten schreiben die Teilnehmer verdeckt. Erst wenn alle fertig sind, werden die Beiträge sichtbar. Es folgt das Gruppieren. Hier versucht TeamRetro über einen automatischen Matchingmechanismus verwandte Beiträge zu erkennen und zu gruppieren. Man kann diese Gruppierungen akzeptieren, verwerfen oder nachträglich verändern. Es folgt die Abstimmung. In der Diskussionsrunde zeigt TeamRetro die Themen sortiert nach abgegebenen Stimmen. Hier kann das Team Kommentare an jede Gruppe von Karten hängen oder Action Items benennen. Jedes Action Item kann man einem Teilnehmer zuweisen und mit einem Zieldatum versehen. Abschließend sieht sich das Team die Ergebnisse der Retro an und der Moderator kann die Daten als CSV exportieren.

Wer den Ablauf einmal sehen möchte, ohne sich zuvor anzumelden, kann auf der Homepage eine Livedemo mit einem Bot-Team durchführen. Dabei nimmt man selbst die Rolle des Moderators ein. Die Livedemo ist hervorragend geeignet, den Workflow kennenzulernen. Auch hier zeigt sich, dass die TeamRetro-Macher viel Wert auf einen einfachen Einstieg legen. Einem Moderator gelingt es mit wenig Übung und ohne Schulung der Teammitglieder, Retros mit Teams durchzuführen, die in der Bedienung des Tools keine Vorkenntnisse haben.

Neben der Retrospektive besitzt TeamRetro eine Funktion namens „Health Check". Dabei vergeben die Teammit-



Retrium in der Phase „Sammeln von Beiträgen"; es schließen sich „Gruppieren", „Abstimmen", „Diskutieren" und „Handlungen festlegen" an (Abb. 8).