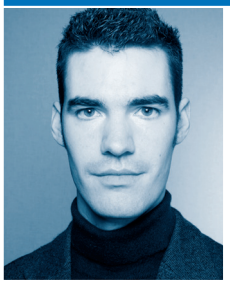
mehr zum thema:
**http://alistair.cockburn.us/**

# ALLES GUTE IST LEICHT: DIE CRYSTAL-METHODENFAMILIE

Viele agile Prozesse schränken die Arbeitsgruppen ein. Der Konflikt zwischen Dogma und Bedarf gefährdet den Projekterfolg. Crystal ist eine Familie von agilen Vorgehensmethoden, in der sich der Prozess verändern kann, um stets den Anforderungen des Moments zu genügen. Projekte jeder Größe finden dadurch eine Vorgehensweise, die sie stets optimal unterstützt. Dieser Artikel gibt einen Überblick die Besonderheiten von Crystal und vergleicht die Familie mit anderen agilen Prozessen.

der autor



Urs Reupke
(E-Mail: ursreupke@gmail.com)
ist Softwareentwickler bei der IDOS AE GmbH in Karlsruhe. Er begeistert sich für Spiele, Kampfkunst, die japanische Sprache und für agile Softwareentwicklung.

Agile Prozesse versprechen viel: Die Entwicklung sei schneller, die Ergebnisse besser und die Kunden zufriedener. Viele Bücher und Trainer bieten an, Unternehmen wieder in Fahrt zu bringen – neue Titel und Zertifikate inklusive. Der erste Erfolg schmeckt häufig süß, doch nach einigen Monaten folgt die Ernüchterung, denn längst nicht alles ist perfekt. Unsicher, was sie ändern dürfen, ändern viele gar nichts – und andere müssen sich bald belehren lassen, vom rechten Pfad abgekommen zu sein. Dieser Konflikt hat uns Worte wie „ScrumBut" beschert – und Lösungen wie Crystal.

## Crystal im Überblick

Crystal ist nicht nur ein Prozess, sondern eine ganze *Familie von agilen Entwicklungsprozessen*. Ihr Ziel ist es, für jedes Projekt den einfachsten Prozess zu finden, der das Projektziel erreicht. Im Mittelpunkt stehen dabei die Projektgröße und das Risiko, das wir mit dem Projekt eingehen. Der Vater von Crystal, Alistair Cockburn, beschreibt in [Coc07], dass diese beiden Faktoren entscheidenden Einfluss auf den Prozess nehmen. In einer Analogie zur Farbe und Härte von Mineralien leitet er daraus den Familiennamen ab (**Abbildung 1** illustriert die Analogie).

Projekte, an denen kleine Gruppen arbeiten, sind „klar". Die Farbe wird dunkler, umso mehr Mitarbeiter sich beteiligen. Je größer der Prozess wird, umso mehr Regeln braucht er, um den Austausch aufrechtzuerhalten. Rollen, Treffen und Artefakte sind fester bestimmt.

Crystal beurteilt das Risiko eines Projekts nach dem möglichen Verlust: Fehler in ungefährlichen Projekte führen lediglich zum Verlust von Komfort, mit steigendem Risiko folgen Verlust von verfügbaren Geldern, kritischen Geldern oder sogar Leben. Der Prozess muss „härter" werden, je mehr auf dem Spiel steht. So kommt ein Vokabeltrainer beispielsweise ohne regelmäßige Prüfungen und Code-Reviews aus. Wenn die gleiche Mannschaft sich dagegen damit beschäftigt, Unterseeboote zu belüften, werden Testläufe, Messungen und Zertifizierungen unumgänglich.

## Ein kooperatives Spiel?

Alle Prozesse der Crystal-Familie teilen den Gedanken, das Softwareentwicklung ein kooperatives Wirtschaftsspiel ist. Cockburn versteht den Begriff „Spiel" im Sinne der Spieltheorie als Modell, das festen Regeln genügt, nicht als unterhaltende Freizeitbeschäftigung. Kooperativ ist es, weil die Teilnehmer – Softwareentwickler und Manager – zusammen arbeiten müssen. Sie gewinnen oder verlieren gemeinsam und nicht allein. Cockburn erörtert die Idee in [Coc] ausführlich.

Das Modell sieht jedes Projekt als neue Runde eines Spiels, in der die Mitspieler zwei Ziele erreichen müssen:

- Einerseits gilt es, das gegenwärtige Projekt erfolgreich zu beenden. Dieses Ziel ist erreicht, wenn die Beteiligten die Software ausliefern und der Kunde zufrieden ist.
- Andererseits ist es unabdingbar, die Vorbereitung für die nächste Runde abzuschließen: Die Firma muss überleben, die Mitarbeiter brauchen Kraft und Wissen für ein weiteres Projekt und – falls ein Bezug zum letzten Projekt

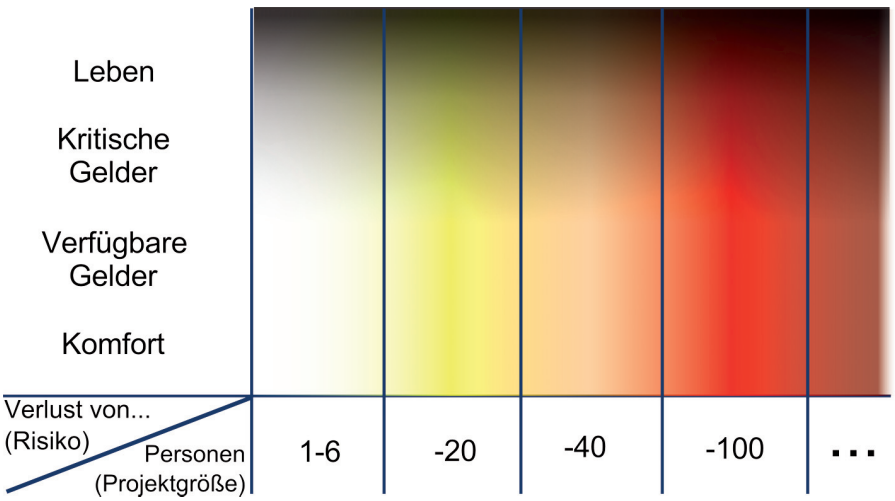



*Abb. 1: Cockburn setzt Größe und Risiko eines Projekts Farbe und Härte von Kristallen gleich.*

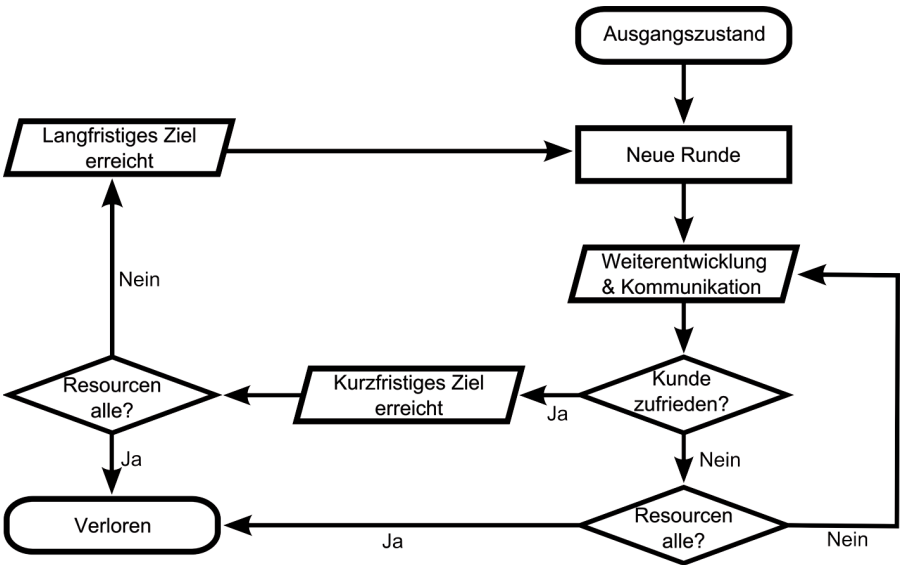



*Abb. 2: Das kooperative Wirtschaftsspiel geht weiter, solange die Teilnehmer erfolgreich sein.*

besteht – muss das Endprodukt dieser vergangenen Runde eine wartbare, verständliche Form haben. Abbildung 2 verdeutlicht den Kreislauf.

Die beiden Ziele konkurrieren miteinander, weil die Spieler sie mit den gleichen Ressourcen verfolgen: der Zeit und dem Wissen des Entwicklerteams. Damit gleichzeitig der Erfolg gewährleistet ist, muss der Prozess die drei großen Parteien – Kunden, Leitung und Entwicklung – auf verschiedene Weise zufriedenstellen. Cockburn hat dafür der Familie Crystal drei Prioritäten gesetzt, die einen Weg zeigen, um beide Ziele zu erreichen.

## Drei Prioritäten

Die drei Werte „Sicherheit", „Effizienz" und „Bewohnbarkeit" (*Safety, Efficiency, Habitability*) genießen in allen Prozessen der Crystal-Familie Vorrang: Erfüllt ein Prozess diese Forderungen, bietet er die besten Voraussetzungen, um das Spiel über viele Runden (Projekte) erfolgreich zu spielen.

- *Sicherheit* gibt dem Kunden Zuversicht, dass das Projekt ein gutes Ergebnis liefert. Es ist die Sicherheit, dass die verfügbaren Ressourcen auf die Projektziele ausgerichtet werden.
- *Effizienz* verspricht der Geschäftsführung, dass das Team die knappen Ressourcen optimal nutzt, damit sich das Projekt lohnt.
- *Bewohnbarkeit* bedeutet für die Entwickler, dass sie sich im Prozess wohl fühlen können und dass dieser sie zu nichts zwingt, was sie in ihrer Arbeit behindert. Ein guter Prozess berücksichtigt selbstverständlich auch Tester, Fachexperten und alle anderen, die für den Erfolg sorgen. Diese Eigenschaft sorgt dafür, dass der Prozess wirklich benutzt wird, anstatt im Archiv zu verstauben.

Die Prioritäten können einander widersprechen: Der effizienteste Prozess muss nicht der am besten bewohnbare sein, und ein Prozess, der Bewohnbarkeit über alles stellt, füllt die Beteiligten möglicherweise nicht mit der Sicherheit, die Ziele auch zu erreichen. Entscheidend ist, dass der Prozess über den gesamten Verlauf des Projekts alle drei Prioritäten angemessen erfüllt. Dadurch wird das Projektziel erreicht, während die Mitarbeiter und die Organisation gleichzeitig für das folgende Projekt bereit sind.

## Sieben Eigenschaften

Um diese Prioritäten stets zu erreichen, schlägt Crystal sieben Eigenschaften vor, die die Projektgruppe in ihrem Prozess anstreben sollte. Auch bei diesen Eigenschaften handelt es sich um Ziele, nicht um Wege. Andere Prozesse schreiben Arbeitsweisen vor. Crystal fordert nur, dass der Prozess diese Eigenschaften mitbringt.

Drei der Eigenschaften sind kritisch für den Prozess – ohne sie ist er kein Crystal-Prozess. Die übrigen sind Ergänzungen, die helfen, die Ziele des kooperativen Spiels zu erreichen.

An erster Stelle stehen *häufige Auslieferungen*. Um ausliefern zu können, müssen die einzelnen Bauteile des Systems zusammenpassen und auf den Rechnern der Kunden funktionieren. Durch häufige Auslieferungen bekommen Sponsoren und Entwickler schnell Rückmeldung von den Benutzern. Diese Eigenschaft hilft, das Projekt auf Kurs zu halten. Damit sinkt die Angst vor dem großen Tag. Jede Auslieferung prüft das System auf Herz und Nieren, bringt das Team näher an den Projekterfolg und zeigt dem Kunden, dass die erwartete Leistung auch geliefert wird.

Die zweite notwendige Eigenschaft ist die ständige *Verbesserung durch Reflexion*. Crystal schreibt weder die Häufigkeit noch die Art dieses Rückblicks vor. Die Einsichten, die die Reflexion gewährt, richten den Prozess immer wieder neu auf die Prioritäten aus: Während der Reflexion deckt die Mannschaft Probleme auf und nimmt sich die Zeit, nach Verbesserungsmöglichkeiten zu suchen. In der nächsten Iteration setzt sie diese Ideen um.

Das letzte Muss, das der Prozess braucht, um Crystal zu genügen, ist eine *enge Kommunikation*. Damit jede Information, die das Projekt betrifft, die richtigen Mitarbeiter erreicht, ist es unabdingbar, dass diese fließen kann. Barrieren behindern diesen Fluss – ganz gleich ob sie in der Umgebung sind oder in den Köpfen. Um dieses Ziel zu erfüllen, müssen die Beteiligten sich leicht erreichen können. Die gemeinsame Arbeit im gleichen Raum übertrumpft dabei die in benachbarten Büros. Letztere wiederum ist mächtiger als jene in verschiedenen Stockwerken.

Technik kann zu einem gewissen Grad dabei helfen, physische Barrieren zu überwinden, verringert aber die Informationsfülle. Nur ein direktes Gespräch bietet eine verzögerungsfreie Übertragung, die Wahrnehmung der Körpersprache und die Gelegenheit, jederzeit einen Gedanken zu ergänzen. Je größer ein Projekt ist, umso größer ist auch sein Platzbedarf. Während der Platzbedarf steigt, sinkt die Leichtigkeit, mit der sich Wissen verbreitet. Hier ist Kreativität gefragt. Wissensstrahler (*Information Radiators*) und elektronische Hilfsmittel verringern die Gefahr, Informationen zu verpassen. Barrierefreie Köpfe bedeuten ebenso viel wie ein barrierefreier

Raum: Wann eine Information für wen wichtig wird, ist ungewiss. Wer zufällig etwas sagt, mit dem er eine Entscheidung in die richtige Richtung lenkt, lässt sich nicht voraussehen. Geheimnisse und Kommunikation im Stillen senken die Chance, dass das richtige Wort in das richtige Ohr gelangt.

Darüber hinaus gefährden Geheimnisse die vierte Eigenschaft, die das Projekt näher an sein Ziele bringt: *persönliche Sicherheit*. Diese erlaubt es, Probleme ohne Angst vor Repressalien anzusprechen: Es ist die Sicherheit, Kritik am eigenen Chef zu üben, die Sicherheit, unangenehme Wahrheiten auszusprechen, oder die Sicherheit, auch Persönliches auf den Tisch zu bringen. Persönliche Sicherheit ermöglicht es dem Team, die eigenen Schwächen und verborgene Probleme zu entdecken und zu bewältigen. Ohne diese Probleme kann das Team enger zusammenarbeiten und hat den ersten Schritt unternommen, um Vertrauen aufzubauen.

Vertrauen verringert die Ablenkung durch soziale Probleme und hilft damit, die fünfte Eigenschaft erfolgreicher Projekte sicherzustellen: den *Fokus*. Dieser braucht mehr als eine Umgebung ohne Ablenkung. Häufige Richtungswechsel verhindern Fokus genauso wie Sonderaufgaben und Unterbrechungen. Die Entwickler benötigen ein festes Ziel, auf das sie ihren Blick richten können. Sie brauchen Zeit, Ungestörtheit und Konzentration, um dieses auch zu erreichen. **Kasten 1** stellt den Wunsch nach Ungestörtheit dem nach Kommunikation gegenüber.

Eine weitere Eigenschaft, die Projekte erfolgreich macht, sind leicht erreichbare und erfahrene Benutzer, also eine Gruppe von Benutzern, die das Problem kennt und täglich damit arbeitet. Sie zeigen dem Team besser als jeder andere, welche Teile des Projekts am wichtigsten sind und wo Stolpersteine die Arbeit behindern.

Diese Gruppe ist ideal dafür, die häufigen Auslieferungen in der Praxis auf Benutzbarkeit und Fehlerfreiheit zu prüfen. Durch den täglichen Umgang können die erfahrenen Benutzer nach ihren Tests leicht neue Anforderungen formulieren. Sie sind die ersten, die Änderungen im Fachbereich zu spüren bekommen – daher können sie schon früh Wünsche aussprechen, die das Produkt erfüllen soll.

Das Team benötigt viel Energie, um sich auf die vorangehenden Eigenschaften zu konzentrieren. Alltägliche Arbeiten behindern es dabei. Crystal empfiehlt drei Maßnahmen, um sie aus dem Weg zu räumen: Anstatt wiederkehrende *Testserien* von Hand durchzuführen, *automatisieren* erfolgreiche Teams sie. Um Änderungen zu untersuchen und auszutauschen, verwenden sie eine *Versionsverwaltung*. Um Fehler so früh wie möglich zu erkennen, *integrieren* sie ihre Änderungen *häufig* und *automatisch*.

Gemeinsam führen diese drei Maßnahmen zu Programmen mit weniger Fehlern. Die verbleibenden Probleme zeigen sich früher und wiegen weniger schwer – sie lassen dem Team Zeit und Kraft, sich auf das Wesentliche zu konzentrieren.

Auf den ersten Blick steht die Forderung nach „Fokus" im Widerspruch zu der nach „enger Kommunikation", setzt letztere doch eine räumliche Nähe voraus, die der Ruhe abträglich ist. Allerdings sind Wortwechsel im Entwicklungsteam durch das gemeinsame Ziel auf wenige Themen beschränkt. Eine enge Zusammenarbeit, wie z. B. beim Programmieren in Paaren, hilft vielen Entwicklern sogar, sich besser zu konzentrieren, statt dem Fokus zu schaden. Der Prozess muss die Balance zwischen allen Eigenschaften halten. Jede Eigenschaft kann anderen im Weg stehen, wenn sie übertrieben wird.

***Kasten 1:*** *Ruhe und Gesprächigkeit.*

Crystal Clear – der Grad für kleine Projekte bis zu sechs Mann – fordert osmotische Kommunikation, statt es, wie die größeren Geschwister, bei enger Kommunikation zu belassen. Osmotische Kommunikation entsteht, wenn diejenigen im gleichen Raum arbeiten, die viel miteinander sprechen (müssen) und deren Themen sich damit zwangsläufig überschneiden. Durch die unmittelbare Nähe entsteht die Möglichkeit, Gespräche mitzuhören und daraus sowohl wichtige Informationen im Vorübergehen aufzuschnappen, als auch Beiträge dazu zu leisten, die andernfalls verloren gingen.

***Kasten 2:*** *Osmotische Kommunikation.*

Die Ausprägung der sieben Eigenschaften hängt von der Einordnung des Prozesses in der Crystal-Familie ab. **Kasten 2** zeigt solch einen Unterschied am Beispiel der Kommunikation in Crystal Clear.

Unabhängig von der Ausprägung steigen die Chancen eines Projekts, das Spiel zu gewinnen, je mehr Eigenschaften der Prozess erfüllt. Crystal unterbreitet lediglich Vorschläge und überlässt die Umsetzung der Projektmannschaft.

## Crystal und die anderen

Die Grenzen eines Crystal-Prozesses sind also bewusst weit gesteckt. Verschiedene agile Ansätze finden sich leicht darin wieder.

### Scrum

Mit seinen Werten – Mut, Offenheit, Selbstverpflichtung, Fokus und Respekt – erfüllt Scrum Crystals Forderung nach Sicherheit, Effizienz und Bewohnbarkeit (zu den Scrum-Werten vgl. auch [Sch02]).

- *Bewohnbarkeit* erreicht Scrum durch Respekt voreinander und durch die Selbstbestimmung aller Parteien, die aus dem Wert der Selbstverpflichtung hervorgeht.
- *Sicherheit* gewährt es durch die klare, iterative Struktur. Kunden, Mannschaft und Manager können den Projektstatus anhand der verschiedenen Artefakte leicht überblicken. Auslieferbare Produktteile, die der Prozess fest vorsieht, gewähren dem Kunden optimalen Einblick.
- *Effizienz* entsteht in Scrum durch die Selbstorganisation des Teams. Das Team verteilt die Aufgaben so, dass es die vorhandenen Fähigkeiten optimal nutzt und damit die größtmögliche Wirtschaftlichkeit erreicht.

Zwei Dinge fehlen Scrum, die Crystal fordert: In Scrum führt jede User-Story das Produkt in einen Zustand, den der PO ausliefern kann. Es bleibt jedoch ihm überlassen, ob er die Auslieferung durchführt. Crystal dagegen besteht darauf, dass häufige Auslieferungen tatsächlich stattfinden, damit der Fluss der Rückmeldungen nicht abreißt. Außerdem müssen die Beteiligten mehr Wert auf Kommunikation und freien Informationsfluss legen. Eine regelmäßige Retrospektive genügt nicht – insbesondere

bei langen Sprints bedarf es mehr, um die Forderung nach enger Kommunikation zu erfüllen.

Scrum hat vorgegebene Praktiken und feste Rollen. Dadurch stößt die Verbesserung durch Reflexion an eine vom Prozess vorgegebene Grenze. Das in Verruf geratene „Scrum, but...“ zeigt, dass Scrum für viele Organisationen nicht flexibel genug ist. Das allgemeiner gehaltene Crystal kann diesen Organisationen helfen, einen neuen Rahmen zu finden.

### XP

*eXtreme Programming (XP)* wirkt mit seinen Werten „Einfachheit“, „Kommunikation“, „Feedback“, „Respekt“ und „Mut“ schon im ersten Moment wie ein entfernter Verwandter von Crystal (vgl. [Wik]). Ein Blick auf die Prinzipien und Praktiken bestätigt, dass XP die gleichen Ziele erreichen möchte, die Crystal mit seinen Prioritäten anstrebt.

Durch häufiges Feedback und den Fokus auf Qualität gewinnt der Kunde Sicherheit. Einfachheit, Feedback und kleine Änderungen helfen der Mannschaft, effizient zu arbeiten – denn sie weiß immer, worauf sie ihr Augenmerk richten muss, um die nächsten Ziele zu erfüllen. Der Wert „Respekt“ und das Prinzip der Menschlichkeit gewährleisten die Bewohnbarkeit des Prozesses.

Häufige Auslieferungen und enge Kommunikation bedeuten XP ebenso viel wie Crystal. XP sieht auch ständige Verbesserung als unabdingbar für den Entwicklungsprozess und setzt wie Crystal regelmäßige Retrospektiven voraus.

XP ist also von Haus aus eine vollwertige Umsetzung von Crystal. Durch die vorgeschriebenen Praktiken und eine eingeschränkte Teamgröße bietet es jedoch weniger Flexibilität, als pures Crystal es verspricht.

### Software-Kanban

Kanban ist ein Werkzeug, um bestehende Prozesse in der Softwareentwicklung zu beschleunigen und fortlaufend zu verbessern. Daher kommt es ohne Wertekanon aus. Dennoch teilt das System viele Gedanken mit Crystal:

- Effektivität und Sicherheit sind zwei der Ziele, die Kanban erreichen möchte. Es betont daher kurze Durchlaufzeiten und bemüht sich, Ausschuss zu vermeiden.
- Häufige Auslieferungen und ständige Verbesserungen sind ebenso Teil des Kanban-Rezepts.
- Kanban deckt vormals unsichtbare Flaschenhälse auf und fordert die Organisation auf, sich damit auseinanderzusetzen. Dabei muss die Mannschaft Ursachen und Veränderungsmöglichkeiten identifizieren. Dieser Rückblick geschieht bei Bedarf, nicht regelmäßig.
- Darüber hinaus sollten alle Beteiligten am Kanban-Prozess regelmäßig zum Operations-Review zusammenkommen, um sich aus der Vogelperspektive über den Stand des Projekts zu informieren und Denkanstöße für Verbesserungsmöglichkeiten zu bekommen.

Der soziale Aspekt der Softwareentwicklung steht bei Kanban im Hintergrund. Die Bewohnbarkeit eines Prozesses, der mit Kanban verbessert wird, ist nicht gewährleistet. Anderson berichtet aber in [And10], dass er nach längerer Arbeit mit Kanban eine Verbesserung der Arbeitsatmosphäre spürte. Um sicherzustellen, dass diese Priorität erfüllt wird, können Teile von Kanban mit Elementen anderer Prozesse kombiniert werden. Crystal kann hierzu Ideen beisteuern, gleiches gilt für Scrum und XP.

Kanban hat nicht den Anspruch, einen vollen Prozess abzubilden oder diesem einen Rahmen zu stecken, und kann daher keine Umsetzung von Crystal sein. Durch die überlappende Zielsetzung können sich die beiden Ansätze jedoch gut ergänzen.

## Fazit

Die Crystal-Familie bietet einen flexiblen Rahmen, mit dem Softwareprojekte jeder Größe behandelt werden können. Durch die weit gefassten Prioritäten und Eigenschaften findet jedes Projekt den Prozess, den es benötigt.

Da Crystal nur Vorschläge unterbreitet, aber keine Praktiken vorschreibt, kann Crystal sich an Veränderungen im Projekt anpassen. Diese Freiheit bedeutet gleichzeitig eine größere Investition: Um Crystal einzusetzen, muss die Mannschaft zunächst einen Prozess gestalten. Cockburns Vorschläge in [Coc05] helfen kleinen Projekten, Crystal (Clear) zu verwenden. Sie finden mit Crystal einen leichtgewichtigen Einstieg in die agile Arbeit. Größere Projekte können in [Coc05] und [Coc07] Ideen gewinnen, wie Crystal ihnen helfen kann. Sie finden mit Crystal Wege, um ihre bestehenden Prozesse an einen größeren Rahmen anzupassen.

## Neugierig geworden?

Im Literaturkasten finden Sie unter [Coc04] den Verweis auf eine Vorabversion von „Crystal Clear“, die kostenlos zur Verfügung steht. Bitte beachten Sie, dass die Druckfassung des Buchs von dieser Ausgabe abweichen kann. ■

## Literatur & Links


**[And10]** D.J. Anderson, Kanban, Blue Hole Press 2010

**[Coc04]** A. Cockburn, Crystal Clear, siehe:
http://st-www.cs.illinois.edu/users/johnson/427/2004/crystalclearV5d.pdf

**[Coc05]** A. Cockburn, Crystal Clear, Addison-Wesley 2005

**[Coc07]** A. Cockburn, Agile Software Development (2nd Ed.), Addison-Wesley 2007

**[Coc]** A. Cockburn, The end of software engineering and the start of economic-cooperative gaming, siehe:
http://alistair.cockburn.us/The+end+of+software+engineering+and+the+start+of+economic-cooperative+gaming

**[Sch02]** K. Schwaber, M. Beedle, Agile Software Development with Scrum, Prentice Hall 2002

**[Wik]** Wikipedia, Extreme Programming, siehe: http://de.wikipedia.org/wiki/Extreme_Programming