**IN DIESEM KAPITEL**

- Die wichtigsten Grundsätze von DevOps im Überblick
- Die Werte von DevOps verstehen
- Die Vorteile für Ihre Organisation erkennen

# Kapitel 1
# Einführung in DevOps

DevOps verändert die Zusammenarbeit von Entwicklungsteams bei der Erstellung und Bereitstellung von Software. Diese breit angelegte, umfassende Philosophie wirkt branchenübergreifend und in den verschiedensten Anwendungsgebieten inspirierend.

Ich definiere DevOps als Entwicklungskultur der Zusammenarbeit, Eigenverantwortung und des Lernens, alles mit dem Ziel, den Lebenszyklus der Softwareentwicklung von der Idee zum fertigen Produkt zu beschleunigen. DevOps kann zwischenmenschliche Reibungsverluste reduzieren, Engpässe beseitigen, die Zusammenarbeit optimieren, die Arbeitsmoral der Entwickler durch selbstbestimmte Tätigkeit erhöhen und die Produktivität der Teams steigern. DevOps ist kein Wundermittel, kann sich aber massiv auf Ihre Organisation und Ihre Produkte auswirken.

In diesem Kapitel betone ich die wichtige Rolle der Unternehmenskultur gegenüber Abläufen und Werkzeugen, diskutiere die Grundsätze und Werte von DevOps und führe aus, inwiefern Ihre Organisation von einem DevOps-Ansatz profitieren wird.

## Was ist DevOps?

Dieses Buch liefert Ihnen kein genaues DevOps-Rezept – denn es gibt keins. DevOps ist eine Philosophie, die Menschen über den Prozess stellt und Prozesse über Werkzeuge. DevOps schafft eine Kultur des Vertrauens, der Zusammenarbeit und kontinuierlichen Verbesserung. Der Entwicklungsprozess wird ganzheitlich betrachtet, dabei werden alle Beteiligten berücksichtigt: Entwickler, Tester, ITler, Sicherheitsleute und Infrastrukturentwickler. DevOps stellt keine dieser Gruppen über die andere, noch wertet es die Bedeutung ihrer Arbeit. Stattdessen ist für ein DevOps-Unternehmen das gesamte Entwicklerteam entscheidend, um dem Kunden die bestmögliche Erfahrung zu bieten. (In Kapitel 2 erfahren Sie mehr über Unternehmenskultur.)

## DevOps hat sich aus Agile entwickelt

Im Jahr 2001 trafen sich 17 Softwareentwickler und veröffentlichten das »Manifest für agile Softwareentwicklung«, in dem die 12 Grundsätze des agilen Projektmanagements dargelegt wurden. (Im Kasten »Die Ursprünge von Agile« in Kapitel 7 erfahren Sie weitere Details darüber.) Dieser neue Workflow war eine Reaktion auf die Frustration und Inflexibilität von Teams, die in einem (linearen) Wasserfallprozess arbeiten. Bei der Arbeit nach agilen Prinzipien müssen sich Entwickler nicht an die ursprünglichen Anforderungen halten oder einem linearen Entwicklungsworkflow folgen, bei dem jedes Team Arbeit an das nächste abgibt. Stattdessen können sie sich an die sich ständig ändernden Erfordernisse des Unternehmens oder des Markts und manchmal sogar an neue Technologien und Tools anpassen.

Obwohl Agile die Softwareentwicklung in vielerlei Hinsicht revolutionierte, konnte es den Konflikt zwischen Entwicklern und Ops-Team nicht lösen. Um technische Fähigkeiten und Spezialitäten bildeten sich immer noch Silos, und die Entwickler gaben den Code weiterhin an den IT-Betrieb weiter, der ihn bereitstellt und betreut.

Im Jahr 2008 machte Andrew Clay Shafer in einem Gespräch mit Patrick Debois seinen Ärger über den ständigen Konflikt zwischen Entwicklern und operativen Mitarbeitern deutlich. Gemeinsam organisierten sie die ersten DevOpsDays in Belgien, um eine bessere und flexiblere Vorgehensweise bei der Softwareentwicklung zu etablieren. Diese Weiterentwicklung von Agile konnte sich durchsetzen und DevOps ermöglicht es mittlerweile Unternehmen auf der ganzen Welt, schneller (und meist auch kostengünstiger) bessere Software zu produzieren. DevOps ist kein kurzlebiger Trend, sondern eine weithin akzeptierte Entwicklungsphilosophie.

## DevOps stellt Menschen in den Mittelpunkt

Wer sagt, dass es bei DevOps nur um Werkzeuge geht, will Ihnen etwas verkaufen. DevOps ist vor allem eine Philosophie, die sich auf Entwickler und die Verbesserung der Zusammenarbeit zwischen ihnen konzentriert – mit dem Ziel, großartige Software zu entwickeln. Sie könnten Millionen in alle DevOps-Tools der Welt investieren und wären trotzdem keinen Schritt weiter in Richtung DevOps-Nirvana. Konzentrieren Sie sich stattdessen auf Ihr wichtigstes Entwicklungsinstrument: die Entwickler. Glückliche Entwickler produzieren großartige Software. Wie bekommen Sie glückliche Entwickler? Nun, Sie schaffen eine kollaborative Arbeitsumgebung, in der gegenseitiger Respekt, gemeinsames Wissen und die Anerkennung harter Arbeit gedeihen können. In den Kapiteln 2 und 15 erfahren Sie mehr darüber, wie Sie Teams aus glücklichen, eigenverantwortlichen Entwicklern bilden können, die stolz auf ihre Arbeit sind und wachstumsorientiert denken.

## Unternehmenskultur ist die Grundlage von DevOps

Ihr Unternehmen verfügt über eine Kultur, selbst wenn sich diese durch Trägheit entwickeln musste. Diese Kultur hat mehr Einfluss auf die Arbeitszufriedenheit, Produktivität und Teamgeschwindigkeit, als Sie es wahrscheinlich für möglich halten.

Unternehmenskultur lässt sich am besten als die unausgesprochenen Erwartungen, das Verhalten und die Werte eines Unternehmens beschreiben. Sie signalisiert Ihren Mitarbeitern

auch, ob die Unternehmensführung offen für neue Ideen ist, und beeinflusst die Entscheidung des Einzelnen, ob er ein Problem ansprechen oder doch lieber unter den Teppich kehren soll.

Kultur möchte gestaltet und verfeinert werden und sollte nicht dem Zufall überlassen werden. Auch wenn die tatsächliche Definition von Unternehmen zu Unternehmen und von Person zu Person variiert, ist DevOps im Grunde eine kulturelle Herangehensweise an die Produktentwicklung.

Eine vergiftete Unternehmenskultur wird Ihre DevOps-Reise ruinieren, bevor sie überhaupt beginnt. Selbst wenn Ihr Entwicklungsteam sich eine DevOps-Mentalität zu eigen macht, werden die Haltungen und Schwierigkeiten der übergeordneten Organisation in Ihre Arbeitsumgebung einfließen.

Mit DevOps vermeiden Sie Schuldzuweisungen, Sie schaffen Vertrauen und konzentrieren sich auf den Kunden. Sie lassen Ihren Entwicklern freie Hand, das zu tun, was sie am besten können: Lösungen ausarbeiten. Wenn Sie mit der Umsetzung von DevOps beginnen, geben Sie Ihren Entwicklern Zeit und Raum, sich darauf einzustellen. Lassen Sie ihnen die Möglichkeit, sich gegenseitig besser kennenzulernen und Kontakte zu Leuten mit unterschiedlichen Fachkenntnissen aufzubauen. Zudem messen Sie den Fortschritt und belohnen Leistungen. Geben Sie niemals Einzelpersonen die Schuld an Misserfolgen. Stattdessen sollte sich das Team gemeinsam kontinuierlich verbessern, und Erfolge sollten gefeiert und belohnt werden.

## Sie lernen, indem Sie den Prozess überwachen und Daten sammeln

Die unvoreingenommene Beobachtung des Arbeitsablaufs ist eine leistungsstarke Technik, mit der Sie die Erfolge und Herausforderungen Ihres Workflows realistisch einschätzen können. Dies ist die einzige Möglichkeit, die richtige Lösung für Problembereiche zu finden, die Engpässe in Ihren Abläufen verursachen. Genau wie bei Software besteht die Lösung nicht immer darin, einfach Kubernetes (oder ein anderes neues Tool) einzusetzen. Sie müssen erkennen, wo die Probleme liegen, bevor Sie sie beheben. Währenddessen sammeln Sie Daten – nicht, um Erfolg oder Misserfolg zu messen, sondern um die Leistung des Teams zu verfolgen. Sie ermitteln, was funktioniert und was nicht und was Sie beim nächsten Mal versuchen sollten. In Kapitel 3 erfahren Sie, wie Sie Engpässe in Ihrem Entwicklungsprozess identifizieren können.

## Überzeugungskraft ist der Schlüssel zur Umsetzung von DevOps

Es ist nicht einfach, DevOps an Ihre Führungskräfte, Kollegen und Mitarbeiter zu verkaufen. Auch für Entwickler ist der Ablauf nicht immer intuitiv. Sollte sich eine gute Idee nicht von selbst verkaufen? Wenn es doch nur so einfach wäre! Als Schlüsselkonzept bei der Umsetzung von DevOps sollten Sie jedoch immer daran denken, dass es die Menschen in den Vordergrund stellt. Die sogenannten »Soft Skills«, also Kommunikation und Zusammenarbeit,

sind für die Einführung von DevOps von zentraler Bedeutung. Andere Leute in Ihrem Team und in Ihrem Unternehmen vom Einsatz von DevOps zu überzeugen, erfordert gute Kommunikationsfähigkeiten. Frühe Gespräche mit Kollegen über DevOps können Wegbereiter für den späteren Erfolg sein – besonders wenn Sie auf unerwartete Hindernisse stoßen.

## Kleine, inkrementelle Änderungen sind unbezahlbar

Der Aspekt von DevOps, der darauf abzielt, immer nur kleine, graduelle Änderungen durchzuführen, hat seine Wurzeln in der schlanken Fertigung, die auf schnelles Feedback, kontinuierliche Verbesserung und kürzere Time-to-Market setzt. Wenn ich über die Einführung von DevOps spreche, verwende ich als Metapher gerne das Wasser. Wasser ist eines der mächtigsten Elemente der Welt. Wenn es nicht gerade eine Hochwasserkatastrophe gibt, halten wir es für relativ harmlos. Der Colorado River hat den Grand Canyon geformt. Langsam, über Millionen von Jahren, schnitt sich das Wasser durch den Stein, um fast zwei Milliarden Jahre alte Bodenschichten und Felsen freizulegen.

Sie können wie Wasser sein, der langsame, unerbittliche Wandel in Ihrem Unternehmen. Hier zu Ihrer Inspiration das berühmte Zitat aus einem Bruce-Lee-Interview (`https://www.youtube.com/watch?v=cJMwBwFj5nQ`):

> Werde formlos, gestaltlos – wie Wasser. Wenn man Wasser in eine Tasse gießt, wird es die Tasse. Gießt man Wasser in eine Teekanne, wird es die Teekanne. Wasser kann fließen, kriechen, tropfen, stürzen und schmettern. Sei Wasser, mein Freund.

Inkrementelle Änderungen anzuwenden, bedeutet beispielsweise, dass Sie ein Problem finden und dieses beheben. Dann beheben Sie das nächste. Sie gehen nicht zu schnell zu viel an und Sie stellen sich auch nicht unmittelbar jedem Kampf. Sie begreifen, dass einige Kämpfe nicht die Energie oder das soziale Kapital wert sind, die sie Sie kosten können.

# Von DevOps profitieren

Das gesamte Buch befasst sich damit, wie Sie und Ihr Team vom Einsatz von DevOps in Ihrem Unternehmen profitieren können. Neben der menschlichen Komponente, die schnellere Auslieferung, verbesserte Funktionalität und furchtlose Innovation ermöglicht, bietet DevOps auch technische Vorteile.

Kontinuierliche Integration und kontinuierliche Bereitstellung (CI/CD) sind eng mit DevOps verknüpft. Durch die kontinuierliche Softwarebereitstellung werden viele der Engpässe beseitigt, die häufig in Teams auftreten, die nur selten liefern. Wenn Sie automatisierte Pipelines erstellen, um neuen Code durch eine robuste Testsuite zu leiten, können Sie sich mit Ihren Implementierungen sicherer fühlen. (Mehr zu CI/CD erfahren Sie Kapitel 11.)

DevOps ermöglicht auch eine schnellere Regeneration nach Zwischenfällen. Sie werden irgendwann unweigerlich eine Ihre Kunden beeinträchtigende Unterbrechung Ihres Dienstes erleben, egal wie gut Ihr Code getestet ist. Aber Teams, die nach der DevOps-Methode arbeiten, finden durch bessere Koordination und Zugänglichkeit, gemeinsames Lernen und bessere Leistungsüberwachung schnellere Lösungen.

Nicht nur die Entwicklungsabteilung Ihrer Organisation profitiert von DevOps. Die geschäftliche Seite wird weniger Kundenreklamationen, schnellere Bereitstellung neuer Funktionen und eine höhere Zuverlässigkeit der bestehenden Dienste verzeichnen.

Mit Hilfe von DevOps können Sie mit den bereits vorhandenen Ressourcen mehr erreichen. Die Methode akzeptiert die realen Einschränkungen und zeigt Ihnen, wie Sie in Ihrer speziellen Umgebung erfolgreich sein können.

## Das CALMS-Modell

Während Sie sich mit DevOps vertraut machen, stoßen Sie wahrscheinlich auch auf das CALMS-Modell. Es steht für Kultur, Automatisierung, Schlankheit, Messung und Austausch und bietet einen hilfreichen Rahmen, um die DevOps-Prinzipien zu verstehen und Ihren DevOps-Erfolg bei der Anwendung dieser Prinzipien in Ihrem Unternehmen zu bewerten.

### Kultur (Culture)

Ihre Kultur muss auf Zusammenarbeit und Kundenorientierung ausgerichtet sein. Ihre Entwickler sollten begreifen, dass die Technologie dafür da ist, den Kunden das Leben zu erleichtern. Wenn die Kunden im Produkt keinen Wert erkennen, wird es versagen. Die Technologie ist diesem Ziel untergeordnet. Die besten DevOps-Kulturen sind extrem kollaborativ und funktionsübergreifend; Menschen aus verschiedenen Teams und mit unterschiedlichen Fähigkeiten arbeiten zusammen an einem besseren Produkt. Zuhören ist ein wichtiger Aspekt der Kommunikation, und wenn Sie sich Gespräche anhören, bekommen Sie einen einfachen Lackmustest der Kultur. Fallen sich die Leute ständig gegenseitig ins Wort? Wenn ja, dann können Sie die Kultur mit Sicherheit noch wesentlich verbessern.

### Automatisierung (Automation)

Routinetätigkeiten sind der schlimmste Albtraum eines Entwicklers, denn sie sind nicht nur, na ja, langweilig, sondern auch ineffizient. Entwickler sprechen Computersprache, damit sie Computern beibringen können, die Aufgaben zu erledigen, die die Menschen nicht übernehmen wollen. Normalerweise sind Code Builds, Testroutinen, Implementierungen und Infrastrukturbereitstellungen am einfachsten zu automatisieren. In Kapitel 3 werde ich noch genauer darauf eingehen, welche Änderungen sich besonders niederschwellig umsetzen lassen.

### Schlank (Lean)

Schlank bezieht sich nicht nur auf die schlanke Fertigung. Das Prinzip lässt sich auch auf die Natur von DevOps-Teams übertragen, die agil und schlagkräftig sind. Lean-Teams vermeiden Aktivitäten mit geringen Auswirkungen, da sie für den Kunden keinen Mehrwert bieten. Ein weiterer Aspekt von Schlankheit ist die konsequente kontinuierliche Verbesserung. Jeder arbeitet wachstumsorientiert und will sich ernsthaft verbessern.

### Die Geschichte einer Entwicklerin: Was mich zu DevOps geführt hat

Ich will Ihnen ein kleines Geheimnis verraten. Ich bin aus Versehen auf DevOps gestoßen. Ja! Absolut zufällig. Aber ich denke, meine Geschichte spricht Bände über die Bedeutung der DevOps-Bewegung und ihrer Community.

Ich war Java-Backend-Entwicklerin in einem kleinen Betrieb mit einem traditionellen Entwicklerteam. Das Team bestand aus einem Dutzend Entwicklern und zwei Administratoren. (Klingt nach dem üblichen Verhältnis, oder?)

Der Code hatte einen Fehler. Ich hatte den Code daraufhin aktualisiert, der Vorschaubilder in der Anwendung ausgewählt hat. Die Änderungen wurden auf der Homepage aber nicht sichtbar, und die Administratoren gaben mir die Schuld. Ich untersuchte den Fehler und kam zu dem Schluss, dass es sich um ein Problem mit dem Content Delivery Network (CDN) handelte. Aufgrund der Zugangsbeschränkungen des Entwicklerteams konnte ich das Problem nicht selbst beheben. Ich brauchte das Team aus dem operativen Geschäft.

Der Ops-Experte war der Überzeugung, dass es sich um ein Code-Problem handelte, und weigerte sich, mir zu helfen. Das ging dreimal hin und her, bevor ich mich in einen Raum einschloss und verärgert eine Zusammenfassung tippte. »Humpty Dumpty: A Story of DevOps Gone Wrong« war mein erster Tech-Talk, und er war inspiriert von meinen persönlichen Erfahrungen und Frustrationen mit Entwicklern, die sich gegen Ops-Leute behaupten.

In diesem Unternehmen und so vielen anderen war das Operations-Team ein Engpass. Sie hinderten mich an meiner Arbeit. Es war aber nicht ihre Schuld. Die beteiligten Personen machten das Problem nur sichtbar, aber eigentlich handelte es sich um ein Ablaufproblem.

Meine Erfahrung in diesem Job hatte mein Interesse an DevOps geweckt. Während ich über DevOps dazulernte, wurde mir klar, dass es bei den Problemen, die ich erlebt hatte, nicht nur um mich ging! Das war eine Riesenerleichterung. Ich war keine schlechte Entwicklerin. Ich war nur ein Mensch und andere Entwickler waren genauso frustriert von ihrer Arbeit. Es ist mein größter Wunsch, dass Ihnen dieses Buch einerseits die Sicherheit gibt, dass Ihre Erfahrungen berechtigt und normal sind, und es Ihnen andererseits einige Ansätze zeigt, die Ihren Job noch ein wenig toller machen können.

## Messung (Measurement)

Daten sind für DevOps von entscheidender Bedeutung. Messdaten zum Fortschritt beeinflussen nahezu jeden Aspekt der Umgestaltung Ihres Unternehmens. Denken Sie aber daran, dass Fortschritte niemals an individuelle Leistung gebunden sein sollten. Betrachten Sie sie eher als die Beobachtung Ihrer Fortschritte bei einem endlosen Marathon denn als eine Möglichkeit zu wissen, wann Sie »fertig« sind. Sie sind niemals fertig. Niemand ist das jemals.

Statt aus den gesammelten Daten herauszulesen, wie schlecht Sie es machen, sehen Sie darin lieber eine Messung Ihrer Fortschritte. Feiern Sie die Siege. Dieser Ansatz stärkt das gesamte Team und Ihre Entwickler bleiben glücklich, motiviert und produktiv. Ich garantiere Ihnen, dass Sie einiges gut machen, und es ist wichtig, das Gute herauszustellen. In Kapitel 5 spreche ich darüber, was Sie messen können.

### Austausch (Sharing)

DevOps wurde ins Leben gerufen, weil Geschäftstätigkeit und Entwicklung in einem Konflikt standen. Es fehlte an Gemeinsamkeiten und beide Bereiche erhielten Anreize auf unterschiedlichen Grundlagen. Das operative Geschäft wird typischerweise an der Zuverlässigkeit und Verfügbarkeit einer Anwendung gemessen, während Entwickler meist dazu motiviert werden, neue Funktionen für die Anwendung zu entwickeln. (Ich spreche im nächsten Abschnitt ausführlicher darüber, wie betriebliche Abläufe und die Entwicklung gemessen werden.) Wissen Sie, was die größte Bedrohung für die Systemverfügbarkeit ist? Deployments. Entwickler initiieren Deployments mit neuen Code-Versionen. Demzufolge hassen die Leute aus dem Geschäftsbetrieb die Entwickler. Ganz so schlimm ist es meistens zwar nicht, aber es steckt ein Fünkchen Wahrheit drin. Diese Reibung macht die Problemlösung fast unmöglich und am Ende stehen nur noch Schuldzuweisungen. DevOps möchte diese Atmosphäre grundlegend verändern und eine Umgebung schaffen, in der sich beide Teams gegenseitig unterrichten und sich bestärkt fühlen – und so letztlich ein einziges Team bilden, zu dem alle etwas beitragen.

## Das Problem der Interessenskonflikte lösen

In traditionellen Teams stehen sich Entwickler (die den Code schreiben) und IT-Verantwortliche (die die Systeme einsetzen und die Infrastruktur warten) als Gegner eines endlosen Krieges gegenüber. Okay, das stimmt nicht ganz. Aber sie kommen nicht miteinander klar, und das liegt daran, dass sie nach unterschiedlichen Kriterien beurteilt werden.

Entwickler werden typischerweise an der Anzahl der Features gemessen, die sie veröffentlichen oder an der Anzahl der von ihnen behobenen Fehler. (Die Bewertung von Entwicklern nach geschriebenen Codezeilen ist eine schreckliche Idee. Die besten Entwickler löschen häufig mehr Codezeilen, als sie hinzufügen.)

Leider werden Qualität und Zuverlässigkeit des Codes meist nicht gemessen. Daher priorisieren Entwickler natürlich die Arbeit, die sie produktiv wirken lässt. Sie verbringen keine Zeit mit der Überarbeitung des Codes, um ihn lesbarer zu machen, oder der Tilgung der technischen Schuld, die durch die letzte große Produktoffensive entstanden ist.

Im Gegensatz zur Bewertung von Entwicklern werden Ops-Teams typischerweise an der Zuverlässigkeit und Verfügbarkeit der Website gemessen. Wahrscheinlich kennen Sie die fünf Neuner: 99,999 Prozent Verfügbarkeit. Das läuft darauf hinaus, dass Ihre Website nur fünf Minuten pro Jahr offline sein darf. Fünf Minuten … pro Jahr. Das ist ziemlich viel verlangt. Es verursacht auch enorme Betriebskosten aufgrund der großen Speicher- und Computerressourcen, die Ihnen zur Verfügung stehen müssen, ganz zu schweigen vom Personalaufwand, um die Verfügbarkeit auf diesem Niveau zu halten. Den Verantwortlichen werden

oft heroische Anstrengungen abverlangt, um auf Probleme zu reagieren, unabhängig von Tag, Zeit, vorhandener Arbeitsbelastung oder persönlichen Verpflichtungen.

Um den Konflikt zu verdeutlichen: In traditionellen Softwareunternehmen müssen Entwickler neuen Code ausrollen, um neue Funktionen freizugeben. Aber Deployments sind die häufigste Ursache für Dienstunterbrechungen und Website-Ausfälle.

Aus dieser Situation ergeben sich zwei Probleme:

- ✔ **Die Verantwortung ist siloisiert:** Entwickler wissen nicht, wie sie ihren Code veröffentlichen oder unterstützen sollen, und es fehlt ihnen an Systemkenntnissen, die sie die Anforderungen an die Infrastruktur verstehen lassen. Die meisten Entwickler wissen nicht (oder interessieren sich nicht dafür), wie ihr Code tatsächlich läuft. Sie haben ihren Job erledigt.
- ✔ **Ziele und Anreize stehen zueinander im Widerspruch:** Entwickler überschütten das Operations-Team mit Code und erwarten, dass dieses den Code bereitstellt und für seine einwandfreie Funktion sorgt. Die operativen Mitarbeiter werden durch Uptime, Verfügbarkeit und Zuverlässigkeit motiviert. Sie gehen oft davon aus, dass der Code schlecht geschrieben ist und werden wegen eines nicht von ihnen verschuldeten Vorfalls angeschrien (oder gefeuert).

Verstehen Sie, warum Sie hörbare Seufzer vernehmen, wenn Entwickler und Betriebspersonal zusammenarbeiten? Mit DevOps versucht man, die Herausforderungen, die durch siloisierte Verantwortung und widersprüchliche Ziele entstehen, zu eliminieren. Durch Anpassung von Anreizen, Wissensaustausch, Beseitigung von Barrieren und Berücksichtigung verschiedener Rollen kann DevOps die zwischenmenschliche Kommunikation und Zusammenarbeit in Ihrem Team erheblich verbessern.