DevOps

Wie IT-Projekte mit einem modernen Toolset und der richtigen Kultur gelingen

DAS VORWORT

Kapitel 1
Einleitung

»Nichts ist so beständig wie der Wandel.«

Dieses Zitat ist nun schon fast 2000 Jahre alt, aber relevanter denn je: Die Entwicklungszyklen von Produkten müssen heute schneller sein, um den sich ständig verändernden Anforderungen gerecht zu werden, auf die wir oft keinen direkten Einfluss haben. Ein Beispiel für diese Notwendigkeit war die COVID-19-Pandemie, die verdeutlichte, wie entscheidend Flexibilität ist.

Sie führte nicht nur dazu, dass man schnell vom Bürojob zum verteilten Arbeiten von überall aus wechselte, sondern betraf auch die Frage, wie die Arbeit überhaupt organisiert wurde. Plötzlich mussten gewohnte Arbeitsweisen und -prozesse von einem Tag auf den anderen angepasst werden. Diese Herausforderung betraf nicht nur reine IT-Unternehmen, sondern auch andere Organisationen, Behörden, Schulen und Vereine.

Unternehmen, die nicht nur ihre internen Arbeitsabläufe, sondern auch ihre Geschäftsmodelle schnell an veränderte Bedingungen anpassen konnten, erhielten einen erheblichen Wettbewerbsvorteil. Die Pandemie zeigte deutlich, wie sich Anforderungen rasch ändern können und wie wichtig es ist, sich agil anzupassen.

In dieser sich ständig wandelnden Umgebung spielt die DevOps-Kultur eine entscheidende Rolle. DevOps fördert die Zusammenarbeit zwischen *Entwicklung* (Dev) und *Betrieb* (Ops), um Prozesse zu beschleunigen, Qualität zu verbessern und gleichzeitig die Flexibilität zu erhöhen. Dieses Konzept ermöglicht es Unternehmen, sich effektiv an veränderte Anforderungen anzupassen und somit erfolgreich in einer dynamischen Welt zu agieren.

Sie suchen konkrete Beispiele? Eine bemerkenswerte gesetzliche Änderung, die wahrscheinlich alle Firmen in Deutschland traf, war der Beschluss vom 3. Juni 2020, die Mehrwertsteuer zum 1. Juli 2020 für ein halbes Jahr von 19 % auf 16 % zu senken. Zwischen diesem Entschluss und der Umstellung waren also nur knappe vier Wochen Zeit vorhanden, um nötige Änderungen zu implementieren, zu testen und überall dort auszurollen, wo es nötig war.

Auch zwei Jahre später, im Jahr 2022, gab es mit der Einführung des 9-Euro-Tickets für die Monate Juni, Juli und August wieder ein schönes Beispiel dafür, wie schnell doch einige Änderungen entschieden und umgesetzt werden können.

So hat die jüngste Entwicklung gezeigt, wie anpassungsfähig Unternehmen und Organisationen sein können, wenn die Notwendigkeit besteht. Anstatt nur in Ausnahmesituationen schnell Anpassungen vorzunehmen, wäre es jedoch weitaus effizienter, wenn dies zur normalen Vorgehensweise gehören würde. Das hätte nicht nur Vorteile für die Unternehmen selbst, sondern vor allem auch für die Nutzer, die von diesen Diensten profitieren.

Und das gilt nicht nur für hippe IT-Startups, die an Apps und Webseiten arbeiten. Vielmehr ist es so, dass viele Unternehmen – und sogar Behörden – bereits heute im Kern IT-Organisationen sind, auch wenn sie nicht unbedingt reine Softwareprodukte anbieten. Zum Beispiel basiert das Hauptgeschäft einer Bank weitgehend auf der Software, die zur Verwaltung von Bankkonten und -diensten benötigt wird. In gewisser Weise ist eine Bank also eher ein Unternehmen der Software-Entwicklung mit einer zusätzlichen Banklizenz. Es ist daher nicht überraschend, dass einige Banken wie die ING sich selbst genau so beschreiben.

Wenn Sie sich bereits ein wenig in der IT auskennen, wird Ihnen diese Forderung bekannt vorkommen: Das ist doch der Grundgedanke der agilen Software-Entwicklung. Soll es hier nicht um DevOps gehen?

Genau bei diesem Punkt stellt sich bereits die erste gute Frage: Was ist Agilität, was ist DevOps, wie hängen beide miteinander zusammen und wo vielleicht auch nicht?

Aber treten wir noch einen Schritt zurück, denn die vielleicht wichtigste Frage ist eigentlich: Warum das Ganze eigentlich? Viel zu oft läuft man Trends jeglicher Art hinterher und macht Dinge, weil es ja schließlich alle so machen. Viel sinnvoller ist jedoch, die Frage nach dem *Warum* zu stellen und darauf basierend dann Änderungen einzuführen.

Tatsächlich gibt es noch sehr viel mehr Fragen, die es zu beantworten gilt, nicht umsonst haben Sie sich wohl dieses Buch angeschafft.

Dieses Buch teilt sich im Grunde in zwei Bereiche auf: Kultur und Technik. Wir betrachten zunächst die *Kultur*: Es werden Konzepte, Ideen, Vor- und auch Nachteile behandelt, ohne zu tief auf einzelne technische Tools eingehen zu müssen. Ganz ausblenden können wir die IT-Tool-Landschaft aber nicht. Da sie im stetigen Wandel begriffen ist, finden Sie einen Überblick, wie der Stand der Technik ist, mit Fokus darauf, was es zu beachten gibt und welche Konzepte zugrunde liegen.

Damit das Buch möglichst wenig trocken und theoretisch wird, betrachten wir das ganze Thema anhand einer fiktionalen Firma, die in Kapitel 3 vorgestellt wird. Diese Firma setzte bislang nicht auf DevOps-Prinzipien. Kapitel für Kapitel werden immer mehr DevOps-Prinzipien in die Firma eingeführt, um zu zeigen, wie eine DevOps-Transformation aussehen kann. Denn die wenigsten werden die DevOps-Kultur auf der grünen Wiese starten können.

1.1 Kultur

Um eines gleich zu Beginn klarzustellen: Viel wichtiger als die letztendlichen Tools ist das Verständnis der DevOps-Kultur. Denn DevOps ist eine Kultur, also eine Art und Weise, wie im Team zusammengearbeitet wird. Viele haben ein falsches Verständnis von DevOps, weshalb es wichtig ist, dass Sie die DevOps-Kultur richtig verstehen, bevor wir uns in diesem Buch um technische Tools kümmern.

Der Begriff *DevOps* selbst ist ein Kofferwort aus »Development« und »Operations«. Es geht darum, die Wände zwischen dem Entwicklungsteam (»Development«) und dem Betriebsteam (»Operations«) einzureißen und zusammen am gemeinsamen Ziel zu arbeiten: an der kontinuierlichen Entwicklung und Bereitstellung der softwarebasierten Lösung.

In diesem Buch möchte ich beschreiben, was die DevOps-Kultur ausmacht. Ich beschreibe, was sie ist, und auch, was sie nicht ist oder wo sie sich mit anderen Arbeitsweisen überschneidet. Am wichtigsten ist allerdings: DevOps muss in der ganzen Organisation gelebt werden, und zwar sowohl ganz oben als auch unten in der Hierarchie.

Die größte Herausforderung für Sie und insbesondere für Ihre Organisation dürfte sein, dass der Schalter im Kopf umgelegt werden muss. Und das ist nicht einfach, denn es gibt bekanntlich viele Wege, die nach Rom führen. Dazu gehört auch, dass alle sich auf andere Aufgaben und Blickwinkel einlassen müssen: Entwickler müssen mal Bereitschaftsdienste leisten, Admins sollten wissen, was ein Pull Request ist, und für die Sicherheit und Zuverlässigkeit der Anwendung sind letztendlich alle verantwortlich.

1.2 Technik

Auch wenn DevOps im Wesentlichen eine Kultur ist und ihre Beschreibung allein schon das Buch relativ gut füllen würde, gehe ich trotzdem auf einige Tools aus der DevOps-Landschaft ein, damit Sie den heutigen Stand der Technik besser nachvollziehen können.

Allerdings geht es mir nicht darum, jedes einzelne Tool bis ins letzte Detail zu besprechen. Ganz im Gegenteil: Sinn und Zweck ist vielmehr, dass Sie die Konzepte der Tools verstehen, damit Ihnen klar wird, warum gerade diese Werkzeuge in der aktuellen DevOps-Tool-Landschaft genutzt werden. Zudem sollte Ihnen anschließend klar sein, warum einige Tools seltener verwendet werden oder auch wo und in welcher Kombination sie gerade bei ordentlicher Umsetzung von DevOps nützlich sein können.

Wichtig ist zu verstehen, dass Technik lediglich ein Hilfsmittel zur Implementierung der Kultur ist. Die besten Tools helfen Ihnen nicht, wenn die Prozesse und die Kultur nicht gelebt werden. Gute Tools unterstützen so die Kultur, allerdings helfen gute Tools nicht bei schlechter Arbeitskultur.

1.3 Mein Weg zu DevOps und zu diesem Buch

Viele Freunde, Bekannte und Arbeitskollegen fragten mich, warum ich denn nun ein DevOps-Buch schreibe, schließlich, so der Tenor, ist das Thema doch schon ein alter Hut und ja bereits »fast durch«. Inzwischen behaupten ja sowohl kleinere als auch größere Firmen, dass sie DevOps machen. In der Realität sieht das alles allerdings ein wenig anders aus.

Meine Motivation für dieses Projekt hat ihre Wurzeln zu Beginn meiner IT-Karriere. Im Jahr 2011 startete ich ein duales IT-Studium bei einem kleinen mittelständischen Unternehmen und wollte Software-Entwickler werden. Zum Programmieren bin ich damals allerdings nie so wirklich gekommen, denn es gab viel mehr zu tun, um die Entwicklungsproduktivität zu verbessern.

In diesem Unternehmen gab es fast gar keine Automatisierungen, vieles wurde händisch zusammengefrickelt. Daran ging viel Arbeitszeit verloren, sodass weder die verschiedenen Teams gut zusammengearbeitet haben noch die Tools für das Bauen und Testen effizient benutzt wurden. Und hier schreibe ich nur von dem Entwickler-Workflow: Was dann mit der Software passierte, wenn sie ausgerollt wurde, erreichte das Entwicklungsteam kaum, denn das fertige Paket wurde quasi »über die Mauer« zu einem anderen Team eines Dienstleisters geworfen, das dann den Betrieb übernahm. Was dort geschah? Keine Ahnung, denn Feedback gab es nie.

Hinzu kam, dass es keine Umgebungen gab, in denen man die Software produktionsnah testen konnte. Ob eine Änderung funktionierte, war immer eine große Frage, die niemand so recht beantworten konnte. Als Notlösung zogen die Mitarbeitenden viele kleine Testumgebungen von Hand hoch, und es ging viel Zeit verloren, da diese Testsysteme immer wieder manuell aufgeräumt und neu installiert werden mussten. Richtig sauber waren die Testsysteme nie.

Für kleine oder mittelständische Unternehmen sind wahrscheinlich auch heute noch viele dieser Unzulänglichkeiten typisch – vor allem dann, wenn die Firmen schon etwas älter sind. Die Aussage »Aber das haben wir schon immer so gemacht!« hört man oft genug, und durch den fehlenden Willen, stetig besser zu werden und dazuzulernen, gibt es in solchen Firmen praktisch keine Fortbildungsmaßnahmen. Nach einigen Jahrzehnten Stillstand merkt man dann leider, dass das Leben der Entwicklerinnen und Entwickler unnötig schwer gemacht wurde, wodurch die Produktivität stark gelitten hat.

Im Laufe der Zeit führte ich damals Automatisierungen ein, vereinfachte das Test-Setup für das Dev-Team, sorgte für automatisierte Test-Umgebungen und machte so den Entwicklungsalltag effizienter und angenehmer.

Ich stellte allerdings schnell fest: Dass es eine Automatisierung gibt, ist eine Sache. Die Nutzung und ihre Akzeptanz von weiter oben in der Hierarchie ist aber eine ganz andere. Denn auch wenn die meisten Automatisierungen mit wenig Aufwand hätten genutzt werden können, war es doch unumgänglich, dass sich die Leute damit auseinandersetzen. Das klappt nur, wenn Zeit dafür da ist und Prozessverbesserungen auch erwünscht sind.

Auch wenn ich DevOps als Begriff und Kultur damals nicht kannte – beides steckte schließlich selbst noch in den Kinderschuhen –, war mir schnell klar: Diese Kultur muss in dem Unternehmen fest verankert sein. Es muss der Wille herrschen, sich stets zu verbessern, komplexe Prozesse zu hinterfragen und Unnützes wegzuschneiden. Auch die Barrieren zwischen den Teams und den Aufgabengebieten müssen von der ganzen Organisation aktiv abgebaut werden, damit effizient gearbeitet werden kann.

Zur gleichen Zeit besuchte ich nacheinander zwei verschiedene Hochschulen: erst eine Fachhochschule und dann eine Universität, an der ich meinen Master absolviert habe. Dort gab es das eine oder andere Gruppenprojekt, in dem etwas gemeinsam in einem Team entwickelt und deployt werden musste. Dabei habe ich festgestellt, dass es bei den verschiedenen Teammitgliedern zwar ordentliche Programmierkenntnisse gab, allerdings hatte noch kaum jemand in einem »echten« Software-Projekt gearbeitet.

Das hieß also, dass zwar grundlegende Entwicklungskenntnisse in der Theorie vorhanden waren, allerdings kaum jemand Ahnung und praktische Erfahrung darin hatte, wie moderne Software-Entwicklung heutzutage aussehen sollte. Das fing mit grundlegenden Werkzeugen wie Git und CI-Tooling an und erstreckte sich weiter auf die richtige Verwendung von Reviews, Tests und Deployments.

Viele dieser Themen wurden zwar an einigen Stellen im Studium erwähnt, allerdings fehlte die praktische Erfahrung, um die vielen Einzelpunkte zusammenzuführen.

Eigentlich wäre ein solches Uni-Projekt eine gute Ausgangslage, um DevOps-Techniken auszuprobieren, denn man hat mit ihm eine leere Spielwiese, auf der man neue Prozesse und Tools ausprobieren kann. In der Realität ist man in Firmen eher mit Projekten konfrontiert, die älter sind als die meisten Studierenden selbst. Ein Sprung zu neuen DevOps-Techniken ist dann wesentlich schwieriger, denn ausgetretene Pfade kann man nur schwer verlassen. Wenn Sie sich gerade in der Ausbildung oder einem Studium befinden, möchte ich Ihnen daher mit diesem Buch etwas an die Hand geben, mit dem Sie schon frühzeitig die Grundlagen der modernen Software-Entwicklung nach DevOps-Prinzipien verstehen können.

Nach meinem Studium hatte ich als Consultant die Möglichkeit, in weitere Unternehmen hineinzuschauen. Dort erkannte ich oft ein ähnliches Muster: festgefahrene, starre Strukturen und Prozesse, in denen Aufgaben und Probleme zwischen verschiedenen Teams hin und her geworfen werden, anstatt kollaborativ daran zusammenzuarbeiten.

Stets hatte dies mit einer Unternehmenskultur zu tun, die eine gute Zusammenarbeit gar nicht ermöglichte, sondern teilweise sogar aktiv verhinderte. Das ging fast immer zulasten der Endnutzer: Nützliche Funktionen und dringende Fehlerkorrekturen erreichten sie erst verspätet. Die Nutzer waren verständlicherweise mit den Produkten unzufrieden, und alle waren genervt. In Privatgesprächen unter Freunden und Bekannten in der IT-Szene hörte ich häufiger ähnliche Storys.

Seit Frühjahr 2020 bin ich als *Solutions Architect* bei GitLab tätig. Dort berate ich Konzerne aus Deutschland, Österreich und der Schweiz und helfe ihnen, die DevOps-Idee mit GitLab umzusetzen. Mit diesem Job wurde mir klar, dass DevOps zwar *irgendwie* fast überall angekommen ist, also auch bei den großen Firmen mit Tausenden von Mitarbeitenden, allerdings wird noch immer fast nur auf die Tool-Landschaft und damit auf die Technik geschaut. Die Kultur wird nur eher schwergängig angepasst, wenn überhaupt.

Alle diese Erfahrungen haben mir deutlich gemacht, dass in der praktischen Umsetzung von DevOps noch ein langer Weg zu gehen ist. In zahlreichen Kundengesprächen wird häufig ein falsches Bild von DevOps gemalt oder es viel zu sehr vereinfacht. Mit diesem Buch möchte ich den Ein- und Umstieg in die DevOps-Welt vereinfachen und so zu Verbesserungen an der Arbeitskultur, der Mitarbeiterzufriedenheit und der generellen Produktivität beitragen.

1.4 Zielgruppe

Das Buch richtet sich grundsätzlich an alle, die eine Rolle im Software-Delivery-Lifecycle innehaben. Dazu gehört explizit auch der Betrieb der Anwendungen, und mit ihm sind nicht nur die Techniker angesprochen, sondern auch die Business-Teams und Führungskräfte, die die Richtung vorgeben. Das Verständnis von DevOps ist für alle relevant, und alle müssen sich gemeinsam an den DevOps-Prinzipien ausrichten, nur dann kann die Umsetzung auch erfolgreich sein.

Das Buch richtet sich also an zwei überschneidende Gruppen. Zum einen soll zunächst erst ein generelles, klares Verständnis vermittelt werden, was DevOps ist und wie eine DevOps-Transformation gelingen kann. Dieser Teil des Buches ist für alle wichtig, die sich näher mit DevOps beschäftigen wollen.

Das sind nicht aber nur die Personen aus einem cross-funktionalen Team mit Fokus auf Entwicklung und Betrieb, sondern auch Personen im Management mit Entschei-

dungsbefugnissen. Denn ohne Unterstützung von oben lässt sich DevOps nicht realistisch umsetzen. Und dafür wird ein tiefes Verständnis von den Prozessen und Dynamiken im kompletten Software-Delivery-Lifecycle benötigt, von der Planung über das Programmieren und die Auslieferung.

Die Ausführungen zu Tools und Techniken sind hauptsächlich für diejenigen interessant, die täglich in einem DevOps-Team arbeiten. Aber auch Entscheider können aus diesem Teil etwas an Erkenntnissen ziehen, da sie dann einige Tools besser einordnen können. Immer wieder habe ich gesehen, wie Firmen und deren Mitarbeiter nach Kubernetes rufen, ohne sich mit den dazugehörigen Fragen beschäftigt zu haben. Bevor Sie sich also aus dem Bauch heraus für einzelne Tools entscheiden, sollten Sie die Konzepte der einzelnen Tools und Techniken verstanden haben.

1.5 Die Struktur des Buches

Zu Beginn erfolgt im nächsten Kapitel eine Einordnung dessen, was DevOps eigentlich ist. Damit das Ganze auch anschaulich ist, wird in Kapitel 3 die *Schick Gekleidet GmbH* vorgestellt. Anhand dieser Firma möchte ich Ihnen zeigen, welche Tücken es in der »alten« Welt gibt und wie Sie es in der »neuen« DevOps-Welt besser machen.

In Kapitel 3 bis Kapitel 11 werden die einzelnen Stages des DevOps-Lifecycles behandelt, dabei wird immer wieder die Schick Gekleidet GmbH als Beispiel genommen. Die einzelnen Kapitel beinhalten sowohl einen kulturellen sowie einen technischen Teil. Mir ist es wichtig, dass die DevOps-Prinzipien anhand von anschaulichen Beispielen verstanden werden, damit Sie anschließend auch Tools und Techniken kennenlernen, die für die jeweilige Stage jeweils relevant ist.

Am Ende jedes Abschnitts folgt eine Zusammenfassung zur »Reflexion«. So möchte ich Ihnen die wichtigsten Aspekte in Erinnerung rufen, ein paar Einblicke aus der Praxis mit Ihnen teilen und idealerweise Ideen liefern, mit denen Sie über Ihre eigene Organisation und Technik nachdenken. Hoffentlich finden Sie direkt ein paar Ansatzpunkte, die Sie anpassen können.

1.6 Feedback

In diesem Buch geht es nicht nur um die Technik. Ganz im Gegenteil: Es geht mir um die Arbeitskultur. Und das ist meine größte Herausforderung. Während es bei technischen Problemen meist eine klare Lösung gibt, ist das bei diesem Thema nicht der Fall.

Wenn es um Kultur geht, gibt es sehr viele Wahrheiten, und bei Änderungen der Arbeitskultur können Sie vieles richtig, aber noch viel mehr falsch machen. Es gibt mannigfache Fallstricke und zahlreiche Fehler, die für Frust sorgen können. Und leider gibt es fast nie eine allgemeingültige Lösung, die für alle Firmen, alle Teams, alle

Menschen passt. Letztlich hängen viele Faktoren in diesem Buch von diversen anderen Faktoren ab: von der Branche, der Firmengröße und der bisherigen Firmenkultur.

Und genau deshalb bin ich an Ihrem Feedback interessiert. Wenn Sie Punkte haben, von denen Sie denken, dass sie hier fehlen oder tiefer behandelt werden sollten, dann schreiben Sie mir doch einfach eine E-Mail an *mail@svij.org*, alternativ kurz gerne auch über Mastodon unter *@svij@ruhr.social* oder in LinkedIn unter *https://www.linkedin.com/in/svijee/*. Ich bin sehr daran interessiert, was bei Ihnen gut oder auch schlecht funktioniert hat und was Sie daraus gelernt haben.

1.7 Danke!

Ein solches Buch lässt sich ohne tatkräftige Unterstützung auf verschiedenen Ebenen – fachlich und nicht fachlich – nicht umsetzen. Entsprechend muss ich mich bei etlichen Leuten bedanken und dabei hoffen, dass ich niemanden vergessen habe.

In erster Linie bedanke ich mich bei Dirk Deimeke als Mentor und Ratschlaggeber, der mir immer ausgezeichnetes Feedback gegeben hat, damit das Buch so wird, wie es ist. Und das ging Hand in Hand mit der Betreuung meines Lektors Christoph Meister, der sehr viel Input geliefert hat, um das Buch immer besser zu machen.

Ebenso gilt mein Dank meinen Eltern, meiner Schwester Shrani, meinen Schwager Max und meiner Nichte Mira, die zuletzt leider viel zu wenig von mir hörten, während ich mit dem Schreiben beschäftigt war.

Mein Dank gilt weiterhin aktuellen und ehemaligen Arbeitskollegen und -kolleginnen von GitLab, die nicht nur wesentlich zu meiner DevOps-Karriere beigetragen haben, sondern von denen ich auch sehr viel Feedback zu einzelnen kleineren und größeren Punkten für dieses Buch erhalten habe.

Mein Dank gilt daher an Ralf Gronkowski, Vlad Budica, Kristof Goossens, Alexander Dess, Timo Schuit, Manuel Kraft, Martin Brümmer, Jörn Schneeweisz, Jörg Heilig, Sarina Kraft, Andrea Obermeier, Michael Friedrich, Sander Brienen, Ted Gieschen, Julia Gätjens, Christoph Parschau, Christoph Caspar, Simon Mansfield, Idir Ouhab, Christopher Allenfort, Dominique Top, Stefania Chaplin, James Moverley und Stephen Walters.

Viel gelernt habe ich in den vergangenen 15 Jahren in der deutschen Linux- und Open-Source-Community, maßgeblich von den Kolleginnen und Kollegen von ubuntuusers.de: Torsten Franz, Stefan Betz, Christian Klotz, Jörg Kastning, Sarah Blume, Philipp Schmidt, Christoph Volkert und Dominik Wagenführ.

Last but not least auch noch ein Dank an ehemalige Arbeitskollegen meiner ersten Arbeitsstelle, der otris software AG, namentlich Dr. Veit Jahns und Thomas Schmidt.

Ach ja, und natürlich den Hörerinnen und Hörern meines Podcasts TILpod, die jeden Monat hören und mitfühlen konnten, wie es um das Buch steht.