Mit freundlicher Unterstützung von IBM

2. limitierte Auflage von IBM

DevOps FÜR DUMMIES®

In diesem Buch:

- Unternehmerische Notwendigkeit und unternehmerischer Mehrwert von DevOps
- Was DevOps kann und wie DevOps eingeführt wird
- Wie die Cloud DevOps voranbringt
- Zehn Mythen über DevOps

Sanjeev Sharma
Bernie Coyne

DevOps

FÜR

DUMMIES®

2. limitierte Auflage von IBM

von Sanjeev Sharma und Bernie Coyne

WILEY

DevOps für Dummies®, 2. limitierte Auflage von IBM
Herausgeber:
John Wiley & Sons, Inc.
111 River St.
Hoboken, NJ 07030-5774
www.wiley.com

ISBN: 978-1-119-17745-6 (Taschenbuch); ISBN: 978-1-119-17746-3 (E-Book)

Hergestellt in den Vereinigten Staaten von Amerika

10 9 8 7 6 5 4 3 2 1

Anmerkungen des Herausgebers

Zu den Personen, die maßgeblich an der Herstellung dieses Buches mitgewirkt haben, gehören:

Projekteditorin: Carrie A. Johnson
Autorenbetreuerin: Katie Mohr
Redaktionsmanagerin: Rev Mengle
Business Development Representative: Sue Blessing
Produktionskoordinatorin : Melissa Cossell

Inhaltsverzeichnis

Einleitung

DevOps (Wortschöpfung aus Development und Operations, zu Deutsch Entwicklung und Betrieb) ist für viele Menschen nur ein Modewort, wie es bei neuen Ideen häufig der Fall ist. Jeder spricht darüber, aber viele wissen gar nicht, was es eigentlich ist. Grob umrissen handelt es sich bei DevOps um einen Ansatz, der auf Lean-Prinzipien und agilen Methoden basiert und bei dem Unternehmensleitung, Entwicklung, Operations-Teams und Qualitätssicherung gemeinsam an einer kontinuierlichen Softwareentwicklung arbeiten. Somit kann ein Unternehmen Marktchancen schnell nutzen und Kundenfeedback zeitnah einbinden. Unternehmensanwendungen sind in der Praxis sehr unterschiedlich und meist ein Mix aus mehreren Technologien, Datenbanken, Endgeräten usw., dass tatsächlich nur der DevOps-Ansatz dieser Komplexität überhaupt noch gerecht werden kann. Doch bei der Frage, wie diese Idee umgesetzt werden sollte, gehen die Meinungen weit auseinander.

Manche sind der Meinung, dass DevOps ausschließlich für Praktiker geeignet ist - andere meinen, dass es alles umfasst. IBM vertritt hier eine weitreichende und ganzheitliche Auffassung und betrachtet DevOps als einen Ansatz für eine unternehmensorientierte Softwareentwicklung. Mit DevOps wird ein neuer oder erweiterter Prozess von der Idee bis hin zur eigentlichen Produktion begleitet, wobei die größere Effizienz dem Kunden einen tatsächlichen Mehrwert bietet und gleichzeitig das Einholen von Feedback ermöglicht, da der Kunde unmittelbar am Prozess beteiligt ist. Doch dafür ist die Beteiligung von Akteuren jenseits der eigentlichen Entwicklungsabteilung und dem Operations-Team erforderlich. Zu einem holistischen DevOps-Ansatz gehört die Beteiligung aller Geschäftssparten, Fachkräfte, Führungskräfte, Partner, Zulieferer usw.

Über dieses Buch

Dieses Buch beschreibt einen unternehmensorientierten DevOps-Ansatz. Um in unserer sich immer schneller drehenden Welt mithalten zu können, ist DevOps mittlerweile unererlässlich geworden. Gerade Unternehmen, die agil und schlank genug sein müssen, um unverzüglich auf Veränderungen bei der Kundennachfrage, auf eine neue Wettbewerbssituation oder Anforderungen des Gesetzgebers reagieren zu können, kommen an diesem Ansatz nicht vorbei.

Wenn Sie dieses Buch in den Händen halten, gehen wir davon aus, dass Sie bereits von DevOps gehört haben. Doch wir möchten, dass Sie sich näher mit dieser Idee vertraut machen und verstehen, wie Ihr Unternehmen von DevOps profitieren kann. Dieses Buch richtet sich an Führungskräfte, Entscheidungsträger und Fachkräfte, denen das Thema DevOps neu ist, die sich genauer mit diesem Konzept auseinander setzen möchten oder jenseits des Hypes um DevOps tiefer in die Materie einsteigen wollen, um zum Kern der Idee durchzudringen.

Im Buch verwendete Symbole

Folgende Symbole finden Sie an den Rändern des Buchs und das bedeuten diese Symbole:

Das Tipp-Symbol verweist auf hilfreiche Informationen zu den unterschiedlichsten Aspekten rund um DevOps.

Alle Bereiche mit einem Wichtig-Symbol sollten Sie sich besonders gut ansehen und merken.

Das Hinweis-Symbol weist auf besonders wichtige Informationen hin.

Hier finden Sie Informationen, die über das Grundlagenwissen zu DevOps hinausgehen. Diese müssen Sie nicht zwingend lesen.

Über das Buch hinaus

Weitere Informationen über DevOps und den DevOps-Ansatz von IBM sowie verfügbare Services finden Sie auf folgenden Webseiten:

- **IBM DevOps Solution:** `ibm.com/devops`
- **DevOps — the IBM approach (Whitepaper):** `ibm.biz/BdEnBz`
- **The Software Edge (Studie):** `ibm.co/156KdoO`
- **Adopting the IBM DevOps Approach (Artikel):** `ibm.biz/adoptingdevops`
- **DevOps Services for Bluemix (Service):** `bluemix.net`

Kapitel 1

Was ist DevOps?

In diesem Kapitel

- Die unternehmerische Notwendigkeit von DevOps verstehen
- Den unternehmerischen Mehrwert von DevOps erkennen
- Die Prinzipien von DevOps erfassen

Alte Gewohnheiten über Bord zu werfen, ist immer mühselig und bedeutet in der Regel, dass man etwas investieren muss. Wenn ein Unternehmen eine neue Technologie, eine neue Methode oder einen neuen Ansatz einführt, muss diese Veränderung unternehmerisch notwendig sein. Um ein Szenario für die Einführung von DevOps zu entwickeln, müssen Sie zunächst die wirtschaftliche Notwendigkeit hinter dieser Entscheidung verstehen und natürlich auch die Herausforderungen, die damit einhergehen. In diesem Kapitel erhalten Sie die Grundlagen, damit Sie ein solches Szenario entwickeln können.

Die unternehmerische Notwendigkeit von DevOps verstehen

Unternehmen entwickeln innovative Anwendungen oder Dienstleistungen, um die Probleme Ihrer Kunden zu lösen. Dabei geht es genauso um interne Unternehmensprobleme (wie bspw. die Entwicklung eines besseren Customer-Relationship-Management-Systems) als auch um die Probleme von Kunden oder Endnutzern (wie bspw. das Bereitstellen einer neuen mobilen App).

Viele Unternehmen scheitern jedoch an Softwareprojekten und meist hängt dieses Scheitern mit den Herausforderungen in der Softwareentwicklung und -bereitstellung zusammen. Obwohl die meisten Unternehmen wissen, dass Softwareentwicklung und -bereitstellung kritisch sind, zeigt eine kürzlich durchgeführte Untersuchung der Branche durch IBM, dass nur 25 Prozent der Unternehmen davon überzeugt sind, über effektive Teams zu verfügen. Diese Lücke in der Umsetzung führt dazu, dass Chancen vertan werden.

Dieses Problem wird noch verstärkt durch die weitreichenden Verschiebungen hinsichtlich der Art der Anwendungen, die Unternehmen heute liefern müssen - vom System of Records hin zum System of Engagement.

- System of Records: Herkömmliche Softwareanwendungen sind große Systeme, die als ein so genanntes System of Records funktionieren, sprich enorme Datenmengen und/oder Datentransfers bedeuten, und für hohe Zuverlässigkeit und Stabilität konzipiert wurden. Da diese Anwendungen nur selten verändert werden müssen, reicht es aus, wenn Unternehmen Ihren Kunden oder für eigene Geschäftsanforderungen ein oder zwei große Releases pro Jahr bereitstellen.
- System of Engagement: Mit dem Aufkommen mobiler Kommunikation und der fortschreitenden Entwicklung von Webanwendungen wird das System of Records durch das so genannte System of Engagement ergänzt, auf das die Kunden direkt zugreifen und worüber sie direkt mit dem Unternehmen interagieren können. Derartige Anwendungen müssen einfach anzuwenden und äußerst leistungsfähig sein, sowie schnell an neue Kundenanforderungen und sich verändernde Marktbedingungen angepasst werden können.

Da Systems of Engagement unmittelbar von Kunden verwendet werden, muss der Fokus ganz besonders auf das Nutzererlebnis, einer schnellen Bereitstellung sowie Agilität liegen. Mit anderen Worten: Ein DevOps-Ansatz ist notwendig.

Natürlich ist ein System of Engagement keine einsame Insel, sondern ist meist an ein System of Records angebunden. Schnelle Veränderungen in dem einen System führen unmittelbar zu Veränderungen in dem anderen System. In der Tat benötigt jedes System, das zügig Innovationen erfordert, schlussendlich DevOps. Derartige Veränderungen und Innovationen werden in erster Linie durch neue Trends wie Cloud Computing, mobile Anwendungen, Big Data und Social Media erforderlich, die sich auf jede Art von Systemen auswirken. Mit diesen neuen Technologien beschäftigen wir uns aus DevOps-Sicht in den Kapiteln 4 und 5.

Den unternehmerischen Mehrwert von DevOps erkennen

DevOps wendet in der gesamten Wertschöpfungskette Lean-Prinzipien und agile Grundsätze an. Auf diese Weise kann ein Produkt oder ein Service so schnell wie möglich bereitgestellt werden, von der

ersten Idee bis hin zu Produktrelease und Kundenfeedback sowie auf diesem Feedback basierenden Nachbesserungen.

Da Sie Ihren Kunden, Zulieferern und Partnern mit DevOps einen tatsächlichen Mehrwert bieten, ist DevOps nicht nur eine IT-Ressource, sondern viel mehr ein essenzieller Geschäftsprozess.

DevOps bietet Ihnen einen signifikanten Return on Investment in den folgenden drei Bereichen:

- Verbesserung der Kundenerfahrung
- Mehr Innovationskapazitäten
- Schnellere Wertschöpfung

Im Folgenden sehen wir uns diese drei Bereiche näher an.

Verbesserung der Kundenerfahrung

Durch eine verbesserte (sprich eine differenzierte und gewinnende) Kundenerfahrung steigt die Bindung Ihrer Kunden an Ihr Unternehmen und damit Ihr Marktanteil. Um das gewährleisten zu können, muss ein Unternehmen kontinuierlich Kundenfeedback einholen und darauf schnell reagieren. Dafür benötigen Sie Mechanismen, über die alle an der Entwicklung einer Anwendung Beteiligten schnelles Feedback geben können, also Kunden, Unternehmenssparten, Anwender, Zulieferer, Partner usw.

In unserer Welt der Systems of Engagement (siehe „Die unternehmerische Notwendigkeit von DevOps verstehen“ in diesem Kapitel) führt diese Fähigkeit, auf agile Art und Weise reagieren und adaptieren zu können, zu einer verbesserten Kundenerfahrung und damit zu einer besseren Kundenbindung.

Mehr Innovationskapazitäten

Moderne Unternehmen setzen auf das Lean-Konzept, um innovativer sein zu können. Ziel dieser Prinzipien ist es, überflüssige Aktivitäten und Nacharbeit zu minimieren und frei werdende Ressourcen stattdessen für Aktivitäten mit höherer Wertschöpfung einzusetzen.

Eine weit verbreitete Idee des Lean-Ansatzes ist beispielsweise das *A/B-Testen*, bei dem Unternehmen einige wenige Nutzer zwei oder mehr Softwareversionen testen und bewerten lassen, die Unterschiedliches leisten können. Der Testsieger wird dann unter allen Nutzern eingeführt, während die durchgefallene Version verworfen wird. Ein solches A/B-Testverfahren macht aber nur mithilfe effizienter und automatisierter Verfahren Sinn, wie sie DevOps bietet.

Schnellere Wertschöpfung

Für eine schnellere Wertschöpfung muss eine entsprechende Kultur aufgebaut, sowie Praktiken und ein Automatisierungsprozess entwickelt werden, damit die Softwareentwicklung bis hin zur Produktion schnell, effizient und zuverlässig ablaufen kann. Als Schnittstelle zwischen Anwendungsentwicklung und Betrieb bietet DevOps die Prozesse, Werkzeuge und Verfahren um Releaseplanung, Test und die Bereitstellung von Software zu vereinfachen und zu optimieren.

Wie ein Unternehmen *Wertschöpfung* definiert, ist von Organisation zu Organisation verschieden und kann sogar von Projekt zu Projekt variieren. Unabhängig davon ist das Ziel von DevOps eine schnellere und vor allem effizientere Wertschöpfung.

Wie DevOps funktioniert

Das DevOps-Konzept folgt mehreren Prinzipien, die sich mit der Zeit verändert haben und noch immer verändern. Einige Softwareunternehmen, darunter auch IBM, haben eigene Varianten entwickelt. All diese Prinzipien folgen einem ganzheitlichen DevOps-Ansatz, den Unternehmen jeder Größe umsetzen können. Zu diesen Prinzipien gehören

- Lösungen entwickeln und in produktionsähnlichen Umgebungen testen
- Wiederholbar und sicher implementieren
- Operative Qualität überwachen und validieren
- Feedbackschleifen erweitern

Diese Prinzipien werden im folgenden Abschnitt noch genauer behandelt.

Lösungen entwickeln und in produktionsähnlichen Umgebungen testen

Dieses Prinzip rührt von dem DevOps-Konzept des *Shift Left* her, wodurch Bedenken seitens des Operations-Teams schon früher im Softwarelebenszyklus, sprich in der Entwicklung, Rechnung getragen wird (siehe Abbildung 1-1).

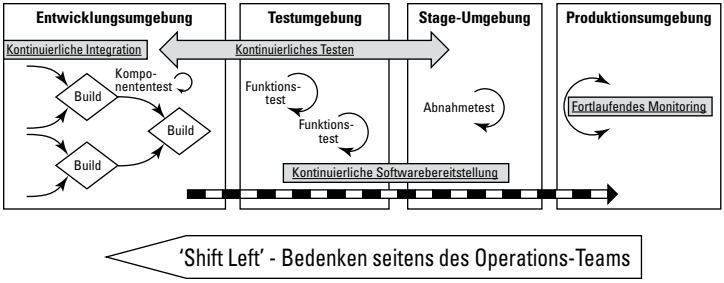


Abbildung 1-1: Das Shift Left-Konzept sorgt dafür, dass operative Aspekte früher im Softwareentwicklungszyklus berücksichtigt werden.

Ziel ist es, der Entwicklungsabteilung und der Qualitätskontrolle (QA) das Entwickeln und Testen in produktionsähnlichen Umgebungen zu ermöglichen, damit die Verantwortlichen wissen, wie sich die Anwendung verhält und richtig funktioniert, bevor sie bereitgestellt wird.

Der erste Kontakt einer Anwendung mit einem produktionsähnlichen System sollte so früh wie möglich im Softwareentwicklungszyklus stattfinden, um zwei wichtigen Herausforderungen gerecht zu werden. Zuerst kann die Anwendung in einer Umgebung getestet werden, die der tatsächlichen Produktionsumgebung sehr nahe kommt, und anschließend können die Bereitstellungsprozesse der Anwendung selbst getestet und validiert werden.

Dies hat auch aus operativer Sicht einen enormen Wert. Denn somit kann das Operations-Team schon frühzeitig abschätzen, wie sich ihre Umgebung verhalten wird, wenn die Anwendung unterstützt wird. Auf diese Weise kann eine fein abgestimmte und anwendungssensitive Umgebung erzeugt werden.

Wiederholbar und sicher implementieren

Wie es die Überschrift schon sagt, sorgt dieses Prinzip dafür, dass Entwickler und Operations-Team einen schlanken (oder zumindest iterativen) Softwareentwicklungsprozess von Anfang bis Ende begleiten. Dabei spielt natürlich die Automatisierung eine große Rolle. Nur so können Prozesse iterativ, regelmäßig, reproduzierbar und sicher gestaltet werden. Aus diesem Grund sollte das Unternehmen eine Delivery Pipeline aufbauen, die das kontinuierliche sowie automatisierte Entwickeln und Testen ermöglicht. Das Thema Delivery Pipelines wird in Kapitel 3 näher beleuchtet.

Durch regelmäßige Bereitstellung können die Mitarbeiter ihre Bereitstellungsprozesse zudem selbst testen und somit das Risiko von Implementierungsfehlern zum Zeitpunkt des Release minimieren.

Operative Qualität überwachen und validieren

Unternehmen können Anwendungen und Systeme in der Produktion in der Regel gut überwachen, da sie über Tools zur Echtzeit-Erfassung von Systemkennzahlen verfügen. Diese Überwachung findet jedoch sehr isoliert statt und die Ergebnisse sind nicht miteinander verknüpft. Dank dieses Prinzips wird auch dem Monitoring schon früher im Entwicklungszyklus Rechnung getragen, dafür müssen automatisierte Tests früher und häufiger im Lebenszyklus erfolgen, um funktionale und nicht-funktionale Merkmale der Anwendung zu überwachen. Wann immer eine Anwendung implementiert und getestet wird, sollte eine Qualitätsmessung und -analyse erfolgen. Ein regelmäßiges Monitoring zeigt frühzeitig Warnhinweise zu operativen und qualitativen Problemen, die in der Produktion auftauchen könnten.

Diese Kennzahlen sollten so erfasst werden, dass alle Akteure sie nachvollziehen und damit arbeiten können.

Feedbackschleifen erweitern

DevOps soll es Unternehmen ermöglichen, schneller auf Veränderungen reagieren und diese zudem zügiger vornehmen zu können. Für die Softwareentwicklung eines Unternehmens bedeutet das, dass Feedback zeitnah verfügbar sein muss, damit die Mitarbeiter dann wiederum aus jeder Maßnahme, die ergriffen wird, etwas lernen können. Dieses Prinzip verlangt von einem Unternehmen, Kommunikationskanäle aufzubauen, über die alle Akteure ihr Feedback geben können.

- Die Entwicklung kann durch Anpassung der Projektpläne oder Prioritäten reagieren
- Die Produktion kann durch Verbesserung der Produktionsumgebung reagieren
- Das Management kann durch Modifizierung der Releasepläne reagieren.

Kapitel 2

Einsatzmöglichkeiten von DevOps kennenlernen

In diesem Kapitel

- Referenzarchitektur von DevOps verstehen
- Vier Möglichkeiten für DevOps-Einführung kennenlernen

Die Einsatzmöglichkeiten von DevOps umfassen den gesamten Softwareentwicklungszyklus. Wo ein Unternehmen mit DevOps beginnt, hängt von seinen Unternehmenszielen ab, welchen Herausforderungen es sich also gegenüber sieht und welche Lücken bei der Softwarebereitstellung bestehen.

In diesem Kapitel lernen Sie die Referenzarchitektur von DevOps kennen und erfahren welche verschiedenen Möglichkeiten es für ein Unternehmen gibt, DevOps einzusetzen.

Möglichkeiten der DevOps-Einführung

Eine *Referenzarchitektur* bietet eine Vorlage für eine bewährte Lösung durch den Einsatz bevorzugter Methoden und Ressourcen. Mit der DevOps-Referenzarchitektur, die Sie in diesem Kapitel kennenlernen, erhalten Praktiker Zugriff auf Richtlinien, Vorschriften und weitere Informationen, die Sie für die Planung oder den Entwurf einer DevOps-Plattform benötigen, die Benutzern, Prozessen und Technologie Rechnung trägt (siehe Kapitel 3).

Über verschiedene Komponenten bietet eine Referenzarchitektur verschiedene Funktionen und Möglichkeiten. Diese Funktionen oder Möglichkeiten können sowohl durch eine einzelne Komponente als auch durch mehrere zusammenwirkende Komponenten zur Verfügung gestellt werden. Daher können Sie die DevOps-Referenzarchitektur in Abbildung 2-1 aus dem Blickwinkel der Kernfunktionen sehen, die diese bieten soll. Während sich die abstrakte Architektur

wird bis zur konkreten Form weiterentwickelt, werden diese Funktionen durch eine Reihe von Experten, systematischen Praktiken und automatisierten Tools bereitgestellt.

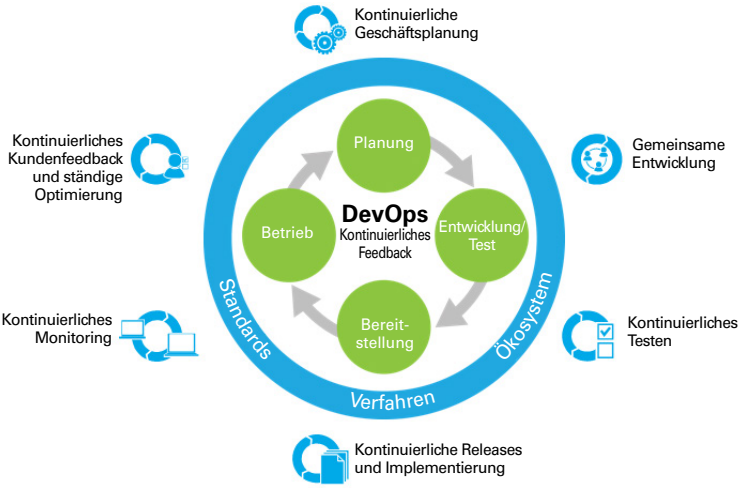


Abbildung 2-1: Die DevOps-Referenzarchitektur.

Die DevOps-Referenzarchitektur aus Abbildung 2-1 sieht die folgenden vier Einführungsmodelle vor:

- Planung
- Entwicklung/Test
- Bereitstellung
- Betrieb

Diese vier Einführungsmöglichkeiten lernen Sie in den folgenden Abschnitten genauer kennen.

Planung

Dieses Modell basiert darauf, sich auf die Festlegung von Unternehmenszielen zu konzentrieren und diese auf Grundlage des Kundenfeedbacks anzugleichen: *Kontinuierliche Geschäftsplanung*.

Unternehmen müssen heute agil sein und schnell auf das Feedback Ihrer Kunden reagieren können. Indem ein Unternehmen dieses Ziel erreicht, konzentriert es sich auf seine Fähigkeiten, die Dinge richtig anzugehen. Unglücklicherweise sind die herkömmlichen Ansätze für

die Produktauslieferung zu langsam für unsere heutige Wirtschaftswelt. Dies beruht zum Teil darauf, dass diese Ansätze auf einer kundenindividuellen Entwicklung und manuellen Prozessen basieren, zum anderen darauf, dass Arbeitsgruppen isoliert voneinander agieren. Informationen, die für eine schnelle Planung und Umplanung erforderlich wären, während gleichzeitig eine größere Wertschöpfung möglich wäre, liegen nur fragmentiert vor und stimmen nicht überein. Häufig trifft das notwendige Feedback zu spät ein, um das Maß an Qualität zu erreichen, das für eine größere Wertschöpfung von Nöten wäre.

Zudem mühen sich Mitarbeiter, erhaltenes Feedback einzubeziehen, das wiederum darüber Auskunft geben könnte, welche Investitionen oberste Priorität haben sollen. Zusätzlich fällt es vielen Mitarbeitern schwer, als eine Organisation zusammenzuwirken und die Erfüllung ihrer Aufgaben in ein kontinuierliches Bereitstellungsmodell zu überführen. Einige Abteilungen empfinden Planung als zudringliche Kontrolle von oben, die sie eher bremst als darin bestärkt, schneller eine größere Wertschöpfung zu liefern.

Dabei bedeutet eine schnellere Bereitstellung größere Flexibilität. Doch diese Geschwindigkeit müssen Sie auch bewerkstelligen können, damit Sie und Ihre Mitarbeiter darauf vertrauen können, dass Sie das Richtige zeitgerecht liefern. Sie können Software nicht schnell entwickeln, wenn Sie nicht darauf vertrauen können, dass Ihre Unternehmensziele, Ihre Kennzahlen und Ihre Plattformen funktionieren.

Mit DevOps können diese konkurrierenden Perspektiven in Einklang gebracht werden, damit Mitarbeiter gemeinsam Geschäftsziele festlegen und anhand des Kundenfeedbacks kontinuierlich an deren Veränderung arbeiten können und somit sowohl die Agilität des Unternehmens als auch die Umsätze verbessert werden. Denn gleichzeitig müssen Unternehmen stets auf die Kosten achten. Indem ein Team Überflüssiges im Entwicklungsprozess erkennt und streicht, wird es effizienter und senkt gleichzeitig Kosten. Mit diesem Ansatz kann ein Team das optimale Gleichgewicht zwischen all diesen Überlegungen und über alle Phasen des DevOps-Lebenszyklus hinweg schaffen und damit zu einem kontinuierlichen Bereitstellungsmodell gelangen.

Entwicklung/Test

Diese Möglichkeit umfasst zwei Methoden: gemeinsame Entwicklung und kontinuierliches Testen. Damit bildet es den Kern der Entwicklungs- und Qualitätssicherungsaufgaben.

Gemeinsame Entwicklung

Die Entwicklung von Software erfordert die bereichsübergreifende Zusammenarbeit in einem Unternehmen, dazu gehören die Führungsebene, Wirtschaftsanalytikern, Enterprise- und Softwarearchitekten, Entwickler, Qualitätssicherung (QA), Operations-Team, Sicherheitsspezialisten, Zulieferer und Partner. Die Fachkräfte aus diesen Bereichen arbeiten auf verschiedenen Plattformen und können auf zahlreiche Standorte verteilt sein. Das Prinzip *gemeinsame Entwicklung* ermöglicht Ihnen eine Zusammenarbeit, indem sie gemeinsam auf einer Plattform und anhand einer Reihe gleicher Verfahren Software planen und entwickeln.

Eine der Kernkompetenzen der gemeinsamen Entwicklung ist die *kontinuierliche Integration* (siehe Abbildung 2-2), eine Methode, wonach Softwareentwickler ihre Arbeit fortlaufend oder regelmäßig in die Arbeitsergebnisse der übrigen Teammitglieder einbauen.

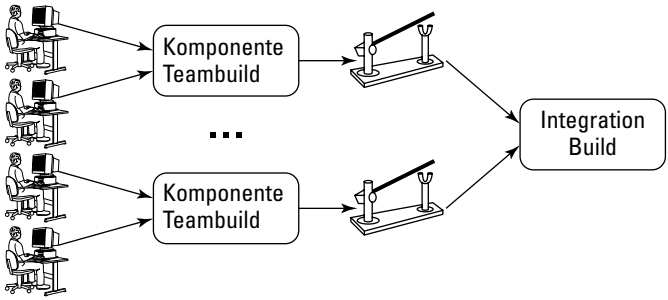


Abbildung 2-2: Zusammenarbeit durch fortlaufende Integration.

Die Idee der kontinuierlichen Integration wurde durch die agile Bewegung populär. Dahinter verbirgt sich das Konzept, die eigene Arbeit regelmäßig in die der übrigen Entwickler eines Teams einfließen zu lassen und anschließend das integrierte Arbeitsergebnis zu testen. Im Falle komplexer Systeme, die sich aus mehreren Einzelsystemen oder Services zusammensetzen, sollen die Entwickler ihre Arbeitsergebnisse auch regelmäßig in die anderen Systeme und Services integrieren. Das regelmäßige Einbinden von Arbeitsergebnissen führt dazu, dass Integrationsrisiken frühzeitig erkannt und umgangen werden können. Bei komplexen Systemen können auf diese Weise zudem bekannte und unbekannte Risiken, sowohl technischer Natur als auch in Bezug auf den Zeitplan, umgangen werden.

Kontinuierliches Testen

Die kontinuierliche Integration (siehe vorheriger Abschnitt) hat mehrere Ziele:

- Ermöglicht fortlaufende Tests und Verifizierung des Codes
- Validierung des erzeugten Codes - In diesem Schritt kann überprüft werden, ob der Code mit den von anderer Entwicklern sowie mit anderen Anwendungskomponenten integriert wurde, funtioniert und wie beabsichtigt arbeiten kann.
- Kontinuierliches Testen der zu entwickelnden Anwendung

Kontinuierliches Testen bedeutet, dass die Anwendung frühzeitig und fortlaufend über den ganzen Lebenszyklus hinweg getestet wird. Das wiederum führt zu niedrigeren Kosten, verkürzten Testläufen und einem kontinuierlichen Feedback zur Qualität. Dieser Prozess wird auch als *Shift Left-Testing* bezeichnet. Dabei werden Entwicklungs- und Testaufgaben integriert, um sicherzustellen, dass die Qualität der Anwendung von Anfang an stimmt und nicht erst später in den Fokus kommt. Dies wird ermöglicht durch Funktionen wie Testautomatisierung und Servicevirtualisierung. Servicevirtualisierung ist eine neue Möglichkeit zur Simulation produktionsähnlicher Umgebungen, die das kontinuierliche Testen erst möglich macht.

Bereitstellung

Aus dem Deploymentmodell ergeben sich die grundlegenden Einsatzmöglichkeiten von DevOps. Kontinuierliches Release und Implementierung sind der nächste Schritt einer fortlaufenden Integration. Mit der gleichen Methode, mit der ein Unternehmen neue Software fortlaufend veröffentlichen und bereitstellen kann, kann auch eine so genannte Delivery Pipeline (siehe Kapitel 3) aufgebaut werden. Mithilfe dieser Pipeline kann eine neue Anwendung kontinuierlich sowie äußerst effizient und automatisiert zur QA und anschließend zur Produktion weitergeleitet werden. Das Ziel von kontinuierlichen Release und Implementierung ist es, neue Features und Funktionen so schnell wie möglich an den Kunden und Nutzer geben zu können.

Die meisten Tools und Prozesse, die den Kern der DevOps-Technologie ausmachen, dienen dazu Integration, Release und Implementierung fortlaufend zu ermöglichen. Dazu erfahren Sie später noch mehr.

Betrieb

Dieses Modell beinhaltet zwei Methoden, die es Unternehmen ermöglichen, die Performance einer freigegebenen Anwendung zu überwachen und Feedback vom Kunden zu erhalten. Auf Basis dieser Daten können Unternehmen auf agile Art und Weise reagieren und Geschäftsvorhaben anpassen, falls dies erforderlich wird.

Kontinuierliches Monitoring

Ein *kontinuierliches Monitoring* stellt Daten und Kennzahlen zur Verfügung, die Operations-Teams, QA, Führungsebene und weitere Akteure über eine Anwendung in unterschiedlichen Phasen des Lieferzyklus nutzen können.

Diese Kennzahlen sind nicht auf die Produktion beschränkt. Anhand derartiger Kennzahlen können die beteiligten Akteure reagieren, indem zu entwickelnde Funktionen und/oder Geschäftsvorhaben, die für eine solche Entwicklung erforderlich sind, erweitert oder geändert werden.

Kontinuierliches Kundenfeedback und laufende Optimierung

Die beiden wichtigsten Informationen für ein Softwareentwicklungsteam sind Daten darüber, wie Kunden die Anwendung nutzen sowie das Feedback, das diese Kunden zu einer Anwendung geben. Neue Technologien machen es möglich, dass Unternehmen das Verhalten sowie Kritikpunkte Ihrer Kunden erfassen können, während diese die Anwendung nutzen. Dank dieses Feedbacks können die verschiedenen Akteure entsprechende Maßnahmen ergreifen, um eine Anwendung zu verbessern und die Kundenerfahrung zu erweitern. Einzelne Sparten können zudem Ihre Geschäftsvorhaben anpassen. So kann die Entwicklung geplante Funktionen angleichen und das Operations-Team kann die Umgebung erweitern, in der die Anwendung implementiert werden soll. Laufendes Feedback ist für DevOps essenziell, denn auf diese Weise können Unternehmen viel agiler sein und schneller auf die Bedürfnisse Ihrer Kunden reagieren.

Kapitel 3

Einführung von DevOps

In diesem Kapitel

- Arbeitsweise der Mitarbeiter effizienter machen
- Prozesse optimieren
- Die richtigen Tools auswählen

Die Einführung einer neuen Funktion erfordert normalerweise eine genaue Planung, die Benutzer, Prozess und Technologie umfasst. Das kann, insbesondere in einem Unternehmen mit vielen - vielleicht auch noch an verschiedenen Standorten ansässigen - Akteuren, nur erfolgreich gelingen, wenn alle drei Aspekte berücksichtigt werden.

In diesem Kapitel beschäftigen wir uns mit dem Thema Benutzer, Prozess und Technologie im Hinblick auf DevOps.

Auch wenn der Name *DevOps* auf den Einsatz im Bereich Entwicklung (engl. Development) und Betrieb (engl. Operations) hinweist, bezieht DevOps alle Akteure eines Unternehmens einschließlich Führungsebene, Architektur, Entwurf, Entwicklung, Qualitätssicherung (QA), Operations-Team, Sicherheit, sowie Partner und Zulieferer ein. Wird ein Akteur, intern oder extern, außen vorgelassen, können wir nicht mehr von einer vollständigen DevOps-Implementierung sprechen.

DevOps - wo anfangen?

In diesem Abschnitt befassen wir uns mit der Frage, wo wir mit DevOps beginnen sollen. Dazu gehören auch das Etablieren der richtigen Unternehmenskultur, das Erkennen von Herausforderungen und das Identifizieren von Engpässen.

Geschäftsziele ermitteln

Um eine Unternehmenskultur etablieren zu können, müssen zunächst alle in dieselbe Richtung gehen und auf dasselbe Ziel hin arbeiten. Sie müssen daher gemeinsame Geschäftsziele für das Team sowie die Organisation als Ganzes ermitteln. Sie müssen Anreize für das gesamte Team schaffen, die auf den Gesamtergebnissen basieren, statt widersprüchliche Team-Ziele zu setzen. Wenn Mitarbeiter wissen, auf welches gemeinsame Ziel sie hin arbeiten und wie der dahingehende Fortschritt ihrer Arbeit gemessen wird, werden ganze Abteilungen oder einzelne Fachkräfte weniger mit ihren eigenen Prioritäten beschäftigt sein.

DevOps ist nicht das Ziel! DevOps ist der Weg zum Ziel.

In Kapitel 4 und 5 finden Sie eine Reihe weiterer Herausforderungen für Unternehmen, die sich mit DevOps lösen lassen. Diese Herausforderungen können Ihrem Unternehmen als Ausgangsbasis dienen, um Ziele zu erkennen und festzulegen, die Sie erreichen wollen. Auf dieser Grundlage können dann diverse Meilensteine bestimmt werden, anhand derer alle Beteiligten diese Ziele erreichen sollen.

Engpässe in der Delivery Pipeline identifizieren

Die schwerwiegendsten Ursachen für Ineffizienz in der Delivery Pipeline kategorisieren wir wie folgt:

- Unnötiger Aufwand (dieselben Informationen und Erkenntnisse müssen wiederholt kommuniziert werden)
- Unnötige Nacharbeit (Fehler, die während der Test- oder Produktionsphase unentdeckt geblieben sind, müssen nun zurück an das Entwicklungsteam verwiesen werden)
- Überproduktion (es wurden Funktionen entwickelt, die nicht benötigt werden)

Eines der größten Probleme in der Delivery Pipeline, das immer wieder zu Engpässen führt, ist das Einrichten der Infrastruktur. Die Einführung des DevOps-Ansatzes führt dazu, dass eine Anwendung schneller bereitgestellt werden kann. Gleichzeitig muss aber auch die Infrastruktur schneller reagieren können. Softwaredefinierte Umgebungen ermöglichen es daher, Infrastruktur als eine Art programmierbares und wiederholbares Muster zu erfassen und das Einrichten damit zu beschleunigen. Sehen Sie sich dazu den Abschnitt „Infrastruktur als Code“ an späterer Stelle in diesem Kapitel genauer an.

Um noch effizienter zu sein, werden Sie dafür sorgen wollen, dass die Pipeline von Anfang bis Ende reibungslos läuft. Um Arbeitsrückstände zu vermeiden, muss der Durchlauf jedes Prozesses gleich sein. Damit Sie dieses Gleichgewicht erreichen können, müssen Sie die Delivery Pipeline an strategisch wichtigen Punkten bewerten können, damit Wartezeiten im Fall von Arbeitsrückständen minimiert, die in Arbeit befindlichen Aufgaben optimiert sowie Kapazität und Fluss jederzeit angepasst werden können.

Der Benutzer bei DevOps

In diesem Abschnitt geht es um die Rolle des Menschen bei der Einführung von DevOps. Dazu gehört auch eine entsprechende Unternehmenskultur.

Die DevOps-Kultur

Mit Blick auf die Wurzeln ist DevOps eine kulturelle Bewegung, es geht bei allem stets um den Menschen. Eine Organisation kann die effizientesten Prozesse oder weitestgehend automatisierte Tools einführen - ohne Benutzer, die diese Prozesse ausführen und diese Tools verwenden, ist das alles völlig wertlos. Das Etablieren einer DevOps-Kultur ist daher wesentlich für die Einführung von DevOps.

Eine DevOps-Kultur zeichnet sich durch einen hohen Grad der Zusammenarbeit über verschiedene Rollen hinweg aus. Es geht dabei immer um das Unternehmen als Ganzes und nicht um die Ziele einzelner Abteilungen, um Vertrauen und eine hohe Wertschöpfung, die auf Lernen durch Experimentieren setzt.

Das Etablieren einer Kultur ist etwas völlig anderes als das Einführen eines Prozesses oder eines Tools. Dafür braucht es Social Engineering (ein besseres Wort gibt es dafür leider nicht) von Teams und Einzelpersonen, die alle ganz unterschiedliche Veranlagungen, Erfahrungen und Vorurteile mitbringen. Diese Diversität ist die eigentliche Herausforderung bei der Etablierung einer solchen Unternehmenskultur.

DevOps basiert auf Transformationsmethoden nach Lean-Prinzipien oder agilen Grundsätzen wie Scaled Agile Framework (SAFe), Disciplined Agile Delivery (DAD) und Scrum. Wenn Ihr Unternehmen bereits nach diesen Prinzipien arbeitet, haben Sie bereits eine Grundlage für die DevOps-Einführung.

Eine Kultur bewerten

Eine Kultur zu bewerten ist extrem schwierig. Denn wie können Sie eine verbesserte Zusammenarbeit oder Moral korrekt messen? Sie könnten Verhaltensweisen und Teammoral direkt beurteilen, indem Sie Umfragen durchführen. Doch Umfragen bergen das Risiko einer hohen statistischen Abweichung, da die Teams normalerweise recht klein sind.

Umgekehrt könnten Sie indirekt Informationen erfassen, indem Sie bspw. messen, wie häufig ein Entwickler ein Mitglied des Operations-Teams oder der QA aufsucht, um gemeinsam ein Problem zu lösen, ohne dafür offizielle Kanäle oder mehrere Managementebenen anzurufen.

Zusammenarbeit und Kommunikation zwischen den Akteuren - das ist die DevOps-Kultur.

Um eine DevOps-Kultur zu etablieren, muss die Führungsebene eines Unternehmens gemeinsam mit ihren Mitarbeitern daran arbeiten, eine Umgebung und Kultur der Zusammenarbeit und des Teilens zu schaffen. Dafür müssen Führungskräfte jede selbst errichtete Barriere, die einem kooperativen Handeln im Wege steht, bei Seite räumen. Typisch ist es, das Operations-Teams zu belohnen, wenn das System verfügbar und stabil läuft, und Entwickler für das Bereitstellen neuer Funktionen. Diese Vorgehensweise führt jedoch zu einer Konkurrenzsituation. Denn das Operations-Team weiß, dass es die Produktion am besten dadurch schützt, keinerlei Änderungen anzunehmen. Auf der anderen Seite gibt es wenig Anreize für Entwickler, sich auf die Qualität zu konzentrieren. Ersetzen Sie diese Maßnahmen daher mit geteilter Verantwortung für das schnelle und sichere Bereitstellen neuer Funktionen.

Zudem sollten die Führungskräfte eines Unternehmens die Zusammenarbeit ihrer Mitarbeiter durch eine bessere Sichtbarkeit weiter fördern. Etablieren Sie eine Reihe gemeinsamer Tools, insbesondere wenn die Mitarbeiter an verschiedenen Standorten tätig sind und nicht vor Ort zusammenarbeiten können. Für eine DevOps-Kultur, die auf Vertrauen und Zusammenarbeit basiert, ist es entscheidend, dass alle Beteiligten die Ziele und den Status eines Projekts kennen.

Für eine DevOps-Kultur müssen sich Menschen in der Regel ändern. Dabei kann es passieren, dass Mitarbeiter, die sich nicht ändern wollen, also die DevOps-Kultur nicht übernehmen möchten, umbesetzt werden müssen.

DevOps-Team

Die Argumente für oder gegen ein spezielles DevOps-Team sind so alt, wie die Idee selbst. Einige Unternehmen wie bspw. Netflix, haben keine seperaten Entwicklungs- und Operations-Teams. Stattdessen verantwortet ein einzelnes „NoOps"-Team beide Aufgabenbereiche. Andere Unternehmen haben DevOps-Teams erfolgreich zusammengeführt, was Konflikte löst und die Zusammenarbeit fördert. Eine solche Gruppe kann eine bestehende Programm- oder Prozessgruppe sein oder es ist ein neu zusammengesetztes Team aus Vertretern aller Abteilungen, die einen Anteil an der zu liefernden Anwendung haben.

Wenn Sie ein DevOps-Team aufbauen möchten, muss es Ihr wichtigstes Ziel sein, dass dieses als Exzellenzzentrum fungiert und die Zusammenarbeit leichter macht. Sie sollten keine weitere Hierar-chiestufe einrichten oder das Team nur als Anlaufstelle für alle DevOps-relevanten Probleme sehen. Denn ein solches Entwicklungsteam würde den eigentlichen Zweck – sprich die Einführung einer DevOps-Kultur - nicht erfüllen.

Die Rolle des Prozesses bei DevOps

Im vorangegangenen Abschnitt haben wir uns mit der Rolle des Menschen bzw. des Benutzers und der Kultur bei DevOps beschäftigt. Über Prozesse definieren wir, was diese Menschen tun. Ihr Unternehmen kann eine tolle Kultur der Kooperation haben, doch wenn Ihre Mitarbeiter das Falsche tun oder das Richtige auf falsche Art und Weise, ist Ihre Organisation nach wie vor fehleranfällig.

Eine riesige Anzahl an Prozessen kann mit DevOps beschrieben werden. Eine Aufzählung dieser Prozesse würde den Rahmen dieses Buches sprengen. In diesem Abschnitt sehen wir uns daher einige der wichtigsten Prozesse hinsichtlich ihrer Einführung in einem Unternehmen an.

DevOps als Geschäftsprozess

In seinen Einsatzmöglichkeiten wirkt sich DevOps auf das gesamte Unternehmen aus. Mit DevOps wird ein Unternehmen agiler und kann die Bereitstellung von Produkten oder Dienstleistungen optimieren. Sie können DevOps aber auch noch weiter ausbauen, indem Sie es betrachten wie einen *Geschäftsprozess:* Eine Reihe von Aktivitäten oder Aufgaben, an deren Ende ein spezifisches Ergebnis (Dienstleistung oder Produkt) für einen Kunden steht.

In der Referenzarchitektur, die Sie in Kapitel 2 kennengelernt haben, bezieht sich der DevOps-Geschäftsprozess auf alles - von der ersten Idee (normalerweise durch die Unternehmensleitung bestimmt) über die Entwicklung bis hin zu Test und Produktion.

Auch wenn dieser Geschäftsprozess noch nicht ausgereift genug ist, um in einzelnen Prozessabwicklungen erfasst werden zu können, sollten Sie die Prozessbeschreibungen genau dokumentieren, die Ihr Unternehmen bereits zur Auslieferung nutzt. Dann können Sie Bereiche ermitteln, in denen Sie Verbesserungen vornehmen müssen, einerseits durch Verbesserung des Prozesses selbst als auch andererseits durch Automatisierung (siehe „Die Rolle der Technologie bei DevOps“ später in diesem Kapitel).

Änderungsmanagementprozess

Unter *Änderungsmanagement* verstehen wir eine Reihe von Aktivitäten, mit deren Hilfe wir Veränderungen steuern, managen und nachverfolgen können, indem wir die Arbeitsprodukte ermitteln, die sich am wahrscheinlichsten verändern werden sowie die Prozesse, die bei der Implementierung dieser Veränderungen eine Rolle spielen. Der Änderungsmanagementprozess, den ein Unternehmen verwendet, ist ein wesentlicher Bestandteil des weiteren DevOps-Prozessflusses. Das Änderungsmanagement lenkt die Art und Weise, wie der DevOps-Prozess Änderungsanfragen und Kundenfeedback aufnimmt und auf diese reagiert.

Unternehmen, die bereits Application-Lifecycle-Management (ALM) eingeführt haben, verfügen über gut definierte und (wahrscheinlich) automatisierte Änderungsmanagementprozesse.

Ein Änderungsmanagement sollte Prozesse beinhalten, die Folgendes ermöglichen:

- Arbeitsaufgabenverwaltung
- Konfigurierbare Arbeitsabläufe
- Projekt-Konfigurationsmanagement
- Planung (agil und iterativ)
- Rollenbasierte Zugangskontrolle

Herkömmliche Ansätze im Änderungsmanagement beschränken sich häufig auf Änderungsanfragen oder das Mängelmanagement. Dabei bleiben häufig die Ereignisse außen vor, die zwischen einer solchen Veränderung oder einem solchen Mängel liegen sowie der damit verbundene Code oder die damit verbundenen Anforderungen. Diese Ansätze bieten keine integrierte Arbeitsaufgabenverwaltung im gesamten Lebenszyklus oder können mithilfe einer eingebauten Funktion allen Arten von Assets nachgehen. DevOps hingegen verlangt von allen Beteiligten, dass sie alle Veränderungen über den gesamten Softwareentwicklungszyklus hinweg im Blick haben und daran mitarbeiten.

DevOps oder ein ALM-zentriertes Änderungsmanagement beinhaltet Prozesse, die eine Arbeitsaufgabenverwaltung für alle Projekte, Aufgaben und damit verbundenen Assets einschließt, und nicht nur die, die von einer Änderungsanfrage oder einem Fehler betroffen sind. Dazu gehören auch Prozesse, die es dem Unternehmen ermöglichen, alle Aufgaben mit allen Artefakten, Projektwerten und anderen Aufgaben zu verbinden, die durch einen an diesem Projekt beteiligten Mitarbeiter erstellt, verändert, referenziert oder gelöscht werden. Mithilfe dieser Prozesse verfügen die Teammitglieder über rollenbasierte Zugänge auf alle änderungsrelevanten Informationen und können zudem die iterative und agile Projektentwicklung unterstützen.

DevOps-Techniken

Im Folgenden finden Sie eine Reihe von Techniken, die Sie bei der Einführung von DevOps brauchen:

- Kontinuierliche Optimierung
- Releaseplanung
- Kontinuierliche Integration
- Kontinuierliche Bereitstellung
- Kontinuierliches Testen
- Kontinuierliches Monitoring und Feedback

Zu diesen Techniken erfahren Sie im Folgenden nähere Einzelheiten.

Kontinuierliche Optimierung

Entsprechend den Lean-Grundsätzen ist die Übernahme eines Prozesses keine einmalige Aktion, sondern ein fortschreitender Prozess. Ein Unternehmen sollte über eingebaute Prozesse verfügen, die auf Bereiche aufmerksam machen, die einer Optimierung bedürfen, während sich das Unternehmen weiterentwickelt und aus bewährten Prozessen lernt. Viele Unternehmen haben spezielle Prozessoptimierungsteams, die an der Verbesserung von Prozessen aufgrund von Beobachtungen und Lektionen arbeiten. Andere setzen darauf, dass das Team, das einen neuen Prozess einführt, sich selbst kontrolliert und auf diese Weise eine Prozessoptimierung vornimmt. Unabhängig von der jeweiligen Methode ist das Ziel dasselbe: eine ständige Optimierung.

Releaseplanung

Releaseplanung ist ein wichtiger Punkt im Geschäftsprozess, der bestimmt wird durch die Notwendigkeit, seinen Kunden etwas anzubieten und dies innerhalb einer bestimmten Zeit tun zu müssen. Aus diesem Grund muss ein Unternehmen eine genaue Releaseplanung vornehmen und entsprechende Managementprozesse vorhalten, um

strategische Leitpläne, Projektpläne und Lieferpläne zu gewährleisten, ebenso wie eine durchgängige Rückverfolgbarkeit über all diese Prozesse hinweg.

Die meisten Unternehmen arbeiten heute mit Datenblättern und Meetings (die meist sehr zeitaufwendig sind), an denen alle Beteiligten aus dem gesamten Unternehmen teilnehmen, um alle Erfordernisse, Entwicklungsstand und Releasepläne aller derzeit in der Entwicklung befindlichen Anwendungen im Blick zu behalten. Genau beschriebene Prozesse und eine Automatisierung ersetzen diese Datenblätter und Meetings jedoch mühelos und ermöglichen eine gestraffte, vor allem aber vorhersagbare Releaseplanung. Der Einsatz von Lean-Prinzipien und agilen Grundsätzen führt zudem zu kleineren, dafür aber häufigeren Releases, was wiederum deren Qualität in den Mittelpunkt rückt.

Kontinuierliche Integration

Kontinuierliche Integration (siehe Kapitel 2) bedeutet für DevOps einen enormen Mehrwert. Auch großen Entwicklerteams, die mit verschiedenen technischen Komponenten an unterschiedlichen Standorten arbeiten, gelingt damit eine agile Softwareentwicklung. Zudem ist auf diese Weise sichergestellt, dass jedes Team seine Arbeitsergebnisse fortlaufend mit den Ergebnissen anderer Entwicklerteams integrieren und validieren kann. Somit können durch eine fortlaufende Integration Risiken minimiert und Probleme frühzeitig im Softwareentwicklungszyklus erkannt werden.

Kontinuierliche Bereitstellung

Eine kontinuierliche Integration führt zwangsläufig zu einer kontinuierlichen Bereitstellung: das Deployment der Software in die Test-, Systemtest-, Staging- und Produktionsumgebungen wird automatisiert. Obwohl einige Unternehmen DevOps nicht in der Produktion umsetzen, verwenden Unternehmen, die nach DevOps arbeiten, in der Regel dieselben automatisierten Prozesse in allen Umgebungen, um die Effizienz zu erhöhen und das Risiko inkonsistenter Prozesse zu reduzieren.

Indem Umgebungen getestet, die Konfiguration automatisiert, Testdaten aktualisiert und anschließend die Software in der Testumgebung implementiert und danach automatisch getestet wird, können Testergebnisse viel früher zurück an die Entwicklung kommuniziert werden.

Dabei ist dieser Schritt einer kontinuierlichen Bereitstellung der kritischste Punkt bei der DevOps-Einführung. Für viele DevOps-Praktiker ist DevOps nur auf diesen Schritt beschränkt. Aus diesem Grund sind auch die meisten verfügbaren DevOps-Tools nur auf die

Bereitstellung zugeschnitten. Doch wie Sie in diesem Buch bereits erfahren haben, ist DevOps viel mehr. Die kontinuierliche Bereitstellung ist eine wesentliche Komponente von DevOps, aber bei weitem nicht die Einzige!

Je nach den Anforderungen und Herausforderungen, denen sich Ihr Unternehmen gegenüber sieht, können Sie mit der DevOps-Einführung auch bei einem anderen Prozess oder mit einem anderen Einführungsmodell wie in Kapitel 2 beschrieben beginnen.

Kontinuierliches Testen

In Kapitel 2 haben wir uns bereits mit dem Thema kontinuierliches Testen beschäftigt. Aus Prozesssicht müssen Sie neue Prozesse in drei verschiedenen Bereichen einführen, um kontinuierliche Tests zu ermöglichen:

- Testumgebung einrichten und konfigurieren
- Testdatenmanagement
- Integrations-, Funktions-, Leistungs- und Sicherheitstests

In einem Unternehmen sollte die Qualitätssicherung (QA) festlegen, welche Prozesse in welchem Bereich übernommen werden sollten. Diese Prozesse können von Projekt zu Projekt variieren, je nach individuellen Testerfordernissen und den Voraussetzungen eines Service-Level-Agreements. Kundenseitige Anwendungen müssten zum Beispiel im Hinblick auf ihre Sicherheit intensiver getestet werden als interne Anwendungen. Das Einrichten einer Testumgebung und das Testdatenmanagement sind insbesondere wichtige Herausforderungen für Projekte, die auf agile Methoden und eine fortlaufende Integration setzen. Anders liegt der Fall bei Projekten, die nur alle paar Monate einen Test durchlaufen. Ähnlich verhält es sich mit Funktions- und Leistungstests für komplexe Anwendungen mit Komponenten, die unterschiedliche Lieferzyklen haben, im Vergleich zu einfachen, monolithischen Web-Anwendungen.

Kontinuierliches Monitoring und Feedback

Kundenfeedback kann unterschiedliche Formen haben, z.B. das Einreichen von Tickets, formale Änderungsanfragen, informelle Beschwerden oder Ratings in App-Stores. Insbesondere durch die Beliebtheit von Social Media und App-Stores (siehe Kapitel 5) brauchen Unternehmen gut beschriebene Prozesse, um das Feedback aus all diesen Quellen erfassen und entsprechend in ihre Lieferpläne einbeziehen zu können. Zudem müssen diese Prozesse agil genug sein, um Marktveränderungen und Änderungen des Gesetzgebers berücksichtigen zu können.

Prozesseinführung bewerten

Sie können den Erfolg einer Prozesseinführung bewerten, indem Sie die Verbesserung einer Reihe von Effizienz- und Qualitätskennzahlen heranziehen. Für diese Art der Beurteilung gibt es zwei Voraussetzungen:

- Sie müssen die richtigen Effizienz- und Qualitätskennzahlen bestimmen. Diese Werte sollten für Ihr Unternehmen wirklich von Bedeutung sein.
- Sie müssen einen Ausgangswert festlegen, von dem aus eine Optimierung erfolgen soll.

Sie können die Prozessreife anhand unterschiedlicher, gut beschriebener Bezugssysteme ermitteln. Für DevOps-Prozesse eignet sich bspw. das IBM DevOps Maturity Model. Weitere Informationen zum IBM Maturity Model unter `ibm.biz/adoptingdevops`.

Auch mithilfe von Überwachungsdaten können Sie Feedback erfassen. Diese Daten werden von jenen Servern, über die die Anwendung läuft, oder durch das Entwicklungsteam, QA oder die Produktion geliefert. Feedback ist auch über in die Anwendung eingebettete Analysetools möglich, die die Aktivität der Nutzer erfassen.

Bei so vielen Daten verliert man schnell den Überblick. Unternehmen benötigen Prozesse, die Daten erfassen und nutzen können. Mithilfe dieser Prozesse können Sie Ihre Anwendungen und Laufumgebungen verbessern.

Die Rolle der Technologie bei DevOps

Dank moderner Technologien können sich Menschen auf wertvolle kreative Aufgaben konzentrieren, während Routineaufgaben automatisiert erfolgen. Der Technik sei Dank können Fachkräfte zudem ihre Zeit und Verfügbarkeit optimal ausnutzen.

Wenn ein Unternehmen an der Entwicklung oder Wartung verschiedener Anwendungen arbeitet, muss alles wiederholbar sein, gleichzeitig aber auch zuverlässig laufen, um eine gleich bleibende Qualität in allen Anwendungen zu gewährleisten. Sie können nicht bei jedem neuen Release oder jeder Fehlerbehebung für eine Anwendung von vorn beginnen. Stattdessen müssen Ressourcen, Code und Methoden wiederverwendbar sein, um kostengünstig und effizient arbeiten zu können.

Automatisierte Aufgaben zu standardisieren macht Ihre Mitarbeiter zudem effizienter (siehe „Die Rolle des Benutzers bei DevOps“ in diesem Kapitel). Unternehmen erleben Fluktuation unter den Mitarbeitern, Auftragnehmern oder Ressourcenanbietern, auch können sich die Beteiligten von Projekt zu Projekt ändern. Doch durch ein gemeinsames Instrumentarium können Fachkräfte überall arbeiten und neue Teammitglieder müssen sich immer nur ein Tool-Set aneignen. Damit verfügen Sie unter dem Strich über einen effizienten, kostengünstigen, wiederholbaren und skalierbaren Prozess.

Infrastruktur als Code

Infrastruktur als Code ist eines der Kernthemen von DevOps. Unternehmen können damit eine kontinuierliche Bereitstellung gewährleisten, in dem Sie einfach den Umfang und die Geschwindigkeit der Umgebung bestimmen können, die eingerichtet und konfiguriert werden muss.

Hinter der Idee einer Infrastruktur als Code steht das Konzept von *softwaredefinierten Umgebungen.* Während Infrastruktur als Code etwas mit der Erfassung von Knotenpunktdefinitionen und der Konfiguration als Code zu tun hat, setzen softwaredefinierte Umgebungen auf eine Technologie, die ganze Systeme aus vielen Knoten bestehend definiert, also nicht nur deren Konfigurationen, sondern auch deren Definitionen, Topologien, Rollen, Beziehungen, Workloads und Workload-Strategien sowie deren Verhalten.

Um eine Infrastruktur als Code zu verwalten, sind drei Arten von Automatisierungstools verfügbar:

- **Anwendungs- oder Middleware-Tools:** Diese Tools können in der Regel sowohl die Anwendungsserver als auch die Anwendungen, die auf diesen laufen, als Code verwalten. Solche Tools erledigen spezielle, anhand von Bibliotheken gebündelte typische Automatisierungsaufgaben für die Technologien, die sie unterstützen. Für weniger anspruchsvolle Aufgaben wie das Konfigurieren von Einstellungen des Betriebssystems sind sie nicht geeignet, dafür aber für vollautomatische Aufgaben auf Server- und Anwendungsebene.
- **Umgebungs- und Implementierungstools:** Diese neuartigen Tools können sowohl den Infrastrukturkonfigurations- als auch den Anwendungscode implementieren.
- **Allgemeine Tools:** Diese Tools sind nicht speziell für eine Technologie konzipiert und können für unterschiedliche Arten von Aufgaben eingesetzt werden - von der Konfiguration eines Betriebssystems auf einem virtuellen oder physischen

Knoten bis hin zur Konfiguration eines Firewall-Ports. Dafür ist zwar mehr Aufwand erforderlich als mit Anwendungs- oder Middleware-Tools, dafür decken sie aber auch eine größere Bandbreite von Aufgaben ab.

Mithilfe eines Umgebungsverwaltungs- und Implementierungstools wie IBM UrbanCode Deploy with Patterns können Unternehmen Umgebungen schnell entwerfen, implementieren und wiederverwenden und somit die Delivery Pipeline beschleunigen.

Delivery Pipeline

Eine *Delivery Pipeline* besteht aus Phasen oder Etappen, die eine Anwendung von der Entwicklung bis zur Produktion durchläuft. Abbildung 3-1 zeigt den typischen Aufbau. Diese Phasen können von Unternehmen zu Unternehmen variieren und sich selbst von einer Anwendung zur anderen unterscheiden, je nachdem welche Erfordernisse das Unternehmen hat, wie der Bereitstellungsprozess sowie die Prozessreife aussehen. Auch das Ausmaß der Automatisierung kann unterschiedlich sein. Einige Unternehmen automatisieren ihre Delivery Pipelines komplett, andere überprüfen ihre Software aufgrund gesetzlicher oder unternehmensinterner Vorgaben manuell. Sie müssen nicht alle Phasen auf einmal angehen. Konzentrieren Sie sich zunächst auf die wichtigsten Aspekte Ihres Unternehmens, und beziehen Sie dann nach und nach alle Etappen ein.

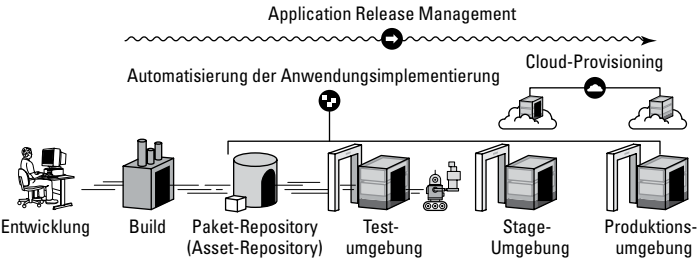


Abbildung 3-2: Phasen einer typischen Delivery Pipeline nach DevOps.

Eine typische Delivery Pipeline beinhaltet nachfolgend beschriebene Phasen.

Entwicklungsumgebung

Die eigentliche Entwicklungsarbeit an einer Anwendung findet in einer *Entwicklungsumgebung* statt, welche den Entwicklern verschiedene Tools an die Hand gibt, um einen Code zu schreiben und zu testen. Neben den Integrated-Development-Environment-(IDE-)Tools, mit deren Hilfe Entwickler einen Code schreiben, beinhaltet diese Phase Tools, die das gemeinsame Entwickeln ermöglichen, dazu

gehören bspw. Tools zur Quellcodeverwaltung, Arbeitsaufgabenverwaltung, Zusammenarbeit, für Komponententests und Projektplanung. Tools in dieser Phase sind in der Regel plattform- und technologieübergreifend nutzbar, je nach Art der anstehenden Entwicklungsaufgabe.

Build-Phase

In der *Build-Phase* wird der Code erzeugt, um die zu implementierenden Binärdateien zu erstellen und zu testen. In dieser Phase können unterschiedliche Build-Tools zum Einsatz kommen abhängig von den plattform- und technologieübergreifenden Erfordernissen. Softwareentwickler arbeiten in der Regel mit Build-Servern, um die große Anzahl erforderlicher Builds, die mit Blick auf eine fortlaufende Integration ebenfalls fortlaufend erzeugt werden, handhaben zu können.

Paket-Repository

Ein *Paket-Repository* (auch als *Asset-Repository* oder *Artefakt-Repository* bezeichnet) ist ein üblicher Speichermechanismus für Binärdateien, die während des Build-Prozesses erstellt wurden. Dieses Repository muss auch die mit den Binärdateien verbundenen Assets speichern, um deren Implementierung zu erleichtern. Dazu gehören Konfigurationsdateien, Infrastruktur-als-Code-Dateien und Implementierungsskripte.

Testumgebung

In einer *Testumgebung* führen Mitarbeiter aus den Bereichen QA, Nutzerakzeptanz und Entwicklung/Test die tatsächlichen Tests durch. In dieser Phase werden eine Reihe unterschiedlicher Tools eingesetzt, die den Anforderungen der Qualitätssicherung entsprechen. Hier einige Beispiele:

- **Testumgebungsverwaltung:** Diese Tools ermöglichen die Bereitstellung und Konfiguration der Testumgebungen. Dazu gehören Infrastruktur-als-Code-Technologien und (sofern die Umgebung in der Cloud liegt) Cloud-Bereitstellungs- und Verwaltungstools.
- **Testdatenverwaltung:** Für jedes Unternehmen, das kontinuierliche Tests durchführen will, ist eine Testdatenverwaltung unerlässlich. Die Anzahl der durchführbaren Tests und die Häufigkeit der Durchführung sind durch die Datenmenge beschränkt, die für Tests verfügbar ist sowie durch die Geschwindigkeit, mit der diese Daten aktualisiert werden können.
- **Integrations-, Funktions-, Leistungs- und Sicherheitstests:** Für jede dieser Testarten sind automatisierte Tools verfügbar. Diese Tools sollten mit einem gemeinsamen Verwaltungstool oder Repository integriert werden, in dem alle Testszenarien,

Testskripte und entsprechende Testergebnisse hinterlegt werden können und wiederum eine Rückverfolgbarkeit zu Code, Anforderungen und Fehlern ermögLichen.

- **Servicevirtualisierung:** Moderne Anwendungen sind nicht mehr einfach und monolithisch aufgebaut. Sie sind stattdessen komplexe Systeme, die von anderen Anwendungen, Anwendungsservern, Datenbanken und selbst Anwendungen und Datenquellen von Drittanbietern abhängig sind. Unter normalen Testbedingungen sind diese Komponenten aber eventuell nicht verfügbar oder zu kostenintensiv. Virtualisierungslösungen simulieren das Verhalten einer Anwendung. Dabei werden Funktionalität und Performance ausgewählter Komponenten innerhalb der Anwendung virtualisiert, um so ein durchgängiges Testen der Anwendung als Ganzes zu ermöglichen. Diese Tools erzeugen so genannte Stubs (virtuelle Komponenten) der Anwendungen und Services, die für die Durchführung des Tests erforderlich sind. Indem die Anwendung mit diesen Stubs interagiert, kann ihr Verhalten und ihre Performance getestet werden. IBM's Rational Test Virtualization Server bspw. stellt eine solche Virtualisierungsmöglichkeit zur Verfügung.

Staging- und Produktionsumgebungen

Anwendungen werden in Staging- und Produktionsumgebungen implementiert. Zu den Tools in dieser Phase gehören Umgebungsverwaltungs- sowie Bereitstellungstools. Tools für eine *Infrastruktur als Code* spielen ebenso eine wichtige Rolle, das ist bedingt durch den großen Umfang einer solchen Umgebung in diesen Phasen. Mit dem Aufkommen von Virtualisierungs- und Cloud-Technologien, können Staging- und Produktionsumgebungen heute auf hunderten oder sogar tausenden von Servern implementiert werden. Mithilfe von Monitoringtools können die implementierten Anwendungen in der Produktion überwacht werden.

Automatisierte Implementierung und Releasemanagement

Um die Automatisierung der Anwendungsimplementierung von einer Phase in die nächste zu verwalten, sind spezielle Tools erforderlich. Einige werden wir uns im Folgenden genauer ansehen.

Automatisierte Bereitstellung

Für DevOps sind Tools für die automatisierte Bereitstellung unerlässlich. Mithilfe dieser Tools können Sie Deployments orchestrieren und zurückverfolgen, welche Version auf welchem Knoten von

welcher Etappe des Build und der Delivery Pipeline bereitgestellt wurde. Zudem können Sie damit alle Konfigurationen der Umgebungen sämtlicher Phasen verwalten, in denen die Anwendungskomponenten bereitgestellt werden müssen.

Einführung der Technologie beurteilen

Tools und Technologien, mit denen Sie den Return on Investment messen können, funktionieren recht einfach. In der Regel können Sie messen, welchen Grad an Effizienz die Automatisierung gebracht hat. Zudem können Sie mithilfe von Automatisierungstools die Skalierbarkeit und Zuverlässigkeit von Aufgaben verbessern, sprich etwas, das mit manuellen Tools nicht immer möglich ist. Mit einem integrierten Set an Automatisierungstools können Sie schließlich die Zusammenarbeit optimieren, ebenso wie die Rückverfolgbarkeit und die Qualität.

Werkzeuge zur Deployment-Automatisierung verwalten die bereitzustellenden Softwarekomponenten, Middleware-Komponenten sowie Middleware-Konfigurationen, die aktualisiert werden müssen, die Datenbankkomponenten, die geändert werden müssen und die Konfigurationsänderungen in den Umgebungen, in denen diese Komponenten implementiert werden sollen. Diese Tools erfassen und automatisieren zudem die Prozesse, die der Ausführung dieser Implementierungen und Konfigurationsänderungen dienen. IBM UrbanCode Deploy ist eine solches Tool zur Deployment-Automatisierung.

Releasemanagement

Wenn Releasepläne und die jeweiligen Deployments für einen Release orchestriert werden, müssen die verschiedenen Abteilungen von der Entwicklung, über QA bis hin zum Operations-Team kooperieren. Releasemangement-Tools ermöglichen Unternehmen die Planung und Durchführung von Releases, bieten ein Kooperationsportal für alle an einer Release beteiligten Personen sowie eine entsprechende Rückverfolgbarkeit für dieser Release und ihre Komponenten über alle Phasen des Build-Prozesses und der Delivery Pipeline hinweg. IBM UrbanCode Release bietet diese Möglichkeiten zum Releasemanagement.

Kapitel 4

Wie die Cloud DevOps voranbringt

In diesem Kapitel

- Cloud als Wegbereiter für DevOps
- Full-Stack-Deployments verstehen
- Unterschiedliche Cloud-Service-Modelle kennenlernen
- Hybride Cloud-Technologie

DevOps und Cloud sind jeweils Katalysator und Wegbereiter des anderen. Wenn ein Unternehmen bereits auf die Cloud-Technologie setzt, wird der Nutzen von DevOps in der Cloud erst richtig offensichtlich. Die Flexibilität, Stabilität und Agilität sowie die Services, die eine Cloud-Plattform bietet, können durch das Einrichten einer Delivery Pipeline in der Cloud weiter optimiert werden. Sämtliche Umgebungen, von der Entwicklung über die Testumgebung bis hin zur Produktion, können so eingerichtet und konfiguriert werden, wie diese und wann diese benötigt werden. Dieser Prozess minimiert umgebungsbedingte Engpässe im Entwicklungsprozess. Zudem können Unternehmen mithilfe der Cloud-Technologie die Kosten für Entwicklungs- und Testumgebungen senken oder ihren Fachkräften eine moderne und optimierte Entwicklungserfahrung bieten. Damit ist die Einführung der Cloud-Technologie mit und für DevOps äußerst attraktiv.

In diesem Kapitel sehen wir uns unterschiedliche Cloud-Modelle für DevOps an und untersuchen, welchen Mehrwert DevOps als Workload in der Cloud hat.

Cloud als Wegbereiter für DevOps

Das wesentliche Ziel von DevOps ist die Beseitigung von Engpässen in der Delivery Pipeline, damit diese effizienter und schlanker wird. Eines der größten Probleme, denen sich Unternehmen gegenüber sehen, ist die Verfügbarkeit und Konfiguration von Entwicklungs-umgebungenEs ist nicht unüblich für Experten, insbesondere für Entwickler und Anwendungstester, eine Umgebung in der Regel nicht über ein formales Ticket anzufordern, zumal eine solche Anfrage Tage, wenn nicht sogar Wochen dauern kann.

Ein Grundprinzip von DevOps ist das Entwickeln und Testen in einer produktionsähnlichen Umgebung. Denn nicht nur Engpässe hinsichtlich der Verfügbarkeit von Umgebungen können eine Herausforderung sein, sondern auch die Frage, ob die verfügbare Umgebung der Produktionsumgebung entspricht. Es kann sein, dass es lediglich Unterschiede in der Konfiguration der Umgebung gibt, bspw. auf Ebene des Betriebssystems (OS) oder der Middleware, oder aber der OS- oder Middleware-Typ der Entwicklungsumgebung unterscheidet sich komplett von der in der Produktion verwendeten Umgebung.

Sind solche Umgebungen aber nicht verfügbar, führt das möglicherweise zu langen Wartezeiten auf Seiten der Fachkräfte. Eine solche Nichtübereinstimmung zwischen Entwicklungs- und Produktionsumgebungen kann zu enormen Qualitätsproblemen führen, denn die Entwickler können nicht verifizieren, wie sich die entwickelte Anwendung in der Produktionsumgebung verhält, oder ob eine Implementierung in der Produktion überhaupt über die Prozesse möglich ist, die zum Implementieren in der Testumgebung eingesetzt wurden.

Mithilfe der Cloud können Sie solche Probleme wie folgt lösen:

- Die Geschwindigkeit, mit der Umgebungen auf Cloud-Plattformen bereitgestellt werden können, ermöglichen einen Self-Service auf Seiten der Entwickler inkl. Verfügbarkeit und Zugriff der Umgebung on Demand.
- Das dynamische Provisioning bzw. De-Provisioning solcher Umgebungen je nach Bedarf führt zu einer besseren Umgebungsverwaltung und niedrigeren Kosten, indem Sie auf permanent verfügbare, statische Testumgebungen verzichten können.
- Indem sie sich die „Pattern“-Technologie zu Nutze machen, können Unternehmen Umgebungen definieren und versionieren wie es mit einer Software zur Einrichtung von Umgebungen möglich wäre, die sich an den Erfordernissen der Experten

orientiert. Was jedoch noch wichtiger ist, es handelt sich dabei um produktionsähnliche Umgebungen.

- Dank der Implementierungsautomatisierung können Sie mit Technologien wie IBM UrbanCode Deploy die Cloud-Umgebung mit nur einem Tool einrichten und die richtige Version einer Anwendung in dieser Umgebung implementieren, so wie Sie es brauchen und wann Sie es brauchen. Zudem können diese Umgebung und Anwendung schnell konfiguriert werden, damit Sie den Anforderungen Ihrer Experten entspricht.
- Mithilfe von Servicevirtualisierungs-Technologien, wie dem IBM Rational Test Virtualization Server, können Sie im Zusammenspiel mit Cloud-Umgebungen Services simulieren, die Sie zu Testzwecken benötigen, ohne wirkliche Instanzen dieser Services bereitstellen zu müssen.

Abbildung 4-1 zeigt das Zusammenspiel von Cloud-Umgebungen, Implementierungsautomatisierung und Servicevirtualisierungs-Technologien für das Einrichten von durchgängigen Entwicklungs-/Testumgebungen.

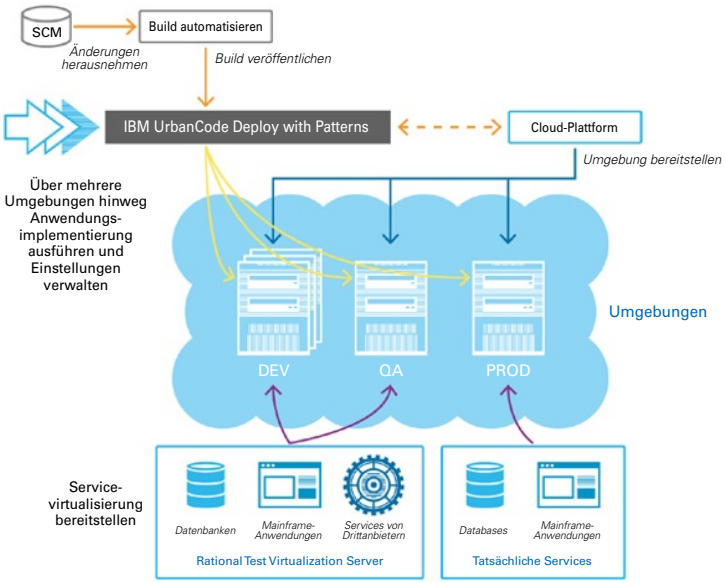


Abbildung 4-1: **Durchgängige Entwicklung/Test in der Cloud.**

Eine Cloud ohne DevOps bedeutet, nicht alle Möglichkeiten einer Cloud auszuschöpfen. Mit der Einführung von DevOps mit Umgebungen in der Cloud können Softwareanbieter von sämtlichen Vorteilen der Cloud-Technologie profitieren.

Full-Stack-Implementierungen

Das Implementieren einer Cloud-Anwendung umfasst zwei Aufgaben: das Implementieren der Anwendung und das Konfigurieren der Cloud-Umgebung, in der die Anwendung läuft. Diese beiden Aufgaben können separat durchgeführt werden oder man kombiniert sie. Das nennt man eine *Full-Stack-Implementierung*. Das sehen wir uns im Folgenden genauer an.

Man kann die Einrichtung der Cloud-Umgebung zum einen von der Anwendungsimplementierung trennen. In diesem Szenario gibt es keine einzelne Orchestrierung der Cloud-Umgebung und der Anwendung, die dort implementiert wird, denn das Implementierungsautomatisierungstool der Anwendung betrachtet die Cloud-Umgebung einfach als statische Umgebung. Auf diese Weise können Sie jedoch nicht von allen Vorteilen der Implementierung in der Cloud profitieren.

Daher gibt es einen zweiten Ansatz. Hierbei wird das Implementierungsautomatisierungstool als ein Orchestrierungstool für das Einrichten der Cloud-Umgebung und die Anwendungsimplementierung in den bereitgestellten Umgebungen genutzt. Das ist mithilfe von so genannten „Blueprints“ möglich, die die Cloud-Umgebungsdefinition und Topologie erfassen und anschließend den in der Cloud-Umgebung definierten Knoten ihre Anwendungskomponenten und Konfigurationen zuordnen.

Unterschiedliche Pattern-Technologien wie IBM Virtual System Patterns und OpenStack-HOT-Vorlagen können verwendet werden, um die Cloud-Umgebungen als Vorlagen zu definieren. Implementierungsautomatisierungstools wie IBM UrbanCode Deploy with Patterns bieten zudem eine Full-Stack-Implementierung mithilfe solcher Blueprints. Dazu gehört das Einrichtung einer Cloud-Umgebung, die in den Blueprints definiert ist, und das Implementieren der Anwendung in der bereitgestellten Umgebung. Wurde die Umgebung einmal eingerichtet, können weitere Anwendungen, Konfigurationen und inhaltliche Änderungen in die Cloud-Umgebung als Update implementiert werden.

Alternativ können Unternehmen immer auf eine Full-Stack-Implementierung setzen, wenn eine Umgebung und die entsprechende

Anwendung stets zusammen als ein Asset implementiert werden. In diesem Fall werden an der bestehenden Umgebung keine Updates vorgenommen.

Ein Cloud-Service-Modell für DevOps auswählen

Bei der Migration in die Cloud müssen Sie zunächst entscheiden, welchen Verantwortungsbereich Sie an die Cloud übertragen möchten und welche Bereiche Sie selbst verantworten möchten. Für die Cloud gibt es hauptsächlich zwei Service-Modelle: Infrastructure as a Service-Lösungen (IaaS) und Platform as a Service-Lösungen (PaaS).

IaaS

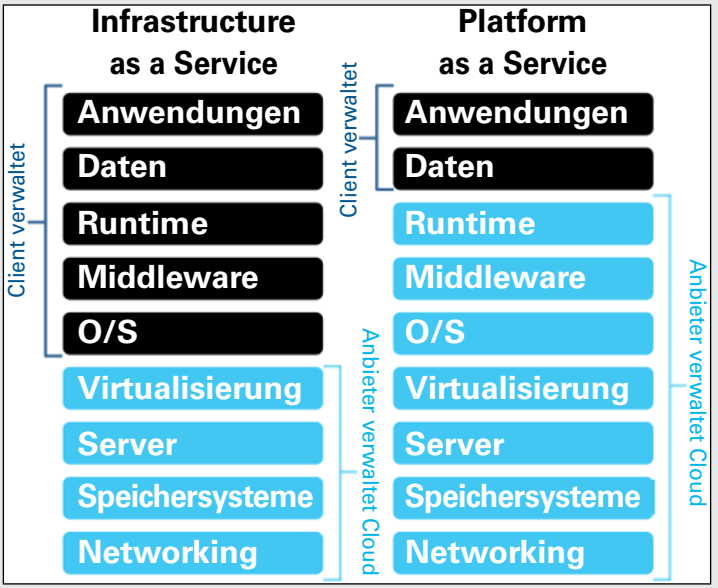
Wenn Sie auf eine Cloud mit einem IaaS-Service-Modell setzen, verwaltet die Cloud-Plattform die zugrunde liegende Infrastruktur und stellt alle Ressourcen und Services zur Verfügung, die Sie zum Verwalten der virtualisierten Infrastruktur benötigen. Installation, Patching und Verwaltung von OS, Middleware, Daten und Anwendung verbleiben in der Verantwortung des Benutzers.

Wenn Sie DevOps als einen Workload in der Cloud einführen möchten, ist die Frage nach dem Cloud-Service-Modell ausschlaggebend für die Art der DevOps-Umsetzung. Bei einem IaaS-Service-Modell ist die Benutzerorganisation für die Verwaltung der gesamten Delivery Pipeline zuständig. Alle Tools und Integrationen der Delivery Pipeline liegen in der Verantwortung des Unternehmens, dazu gehören auch die Auswahl der richtigen Tools und deren Integration in die Delivery Pipeline. Zudem muss die Organisation gewährleisten, dass die Zusammenarbeit zwischen Entwicklung und Operations-Teams nach den Grundsätzen von DevOps erfolgt. Denn nur weil Sie mit einer Cloud-Plattform agieren, heißt das noch lange nicht, dass die Beteiligten nicht weiterhin isoliert voneinander arbeiten, sprich die Entwickler den Code liefern und das Operations-Team die Infrastruktur, jetzt eben nur als cloud-basierter Service.

Nur weil die Cloud-Technologie im Hinblick auf IaaS für die Anwendungsentwickler mit einer enormen Wertsteigerung verbunden ist, benötigen diese dennoch all die erforderlichen DevOps-Funktionen und -Möglichkeiten, damit das Unternehmen voll und ganz von DevOps profitieren kann.

Trennung von Aufgaben

Eine der zentralen Fragen von DevOps in Verbindung mit IaaS ist die Frage nach der Trennung der Aufgaben zwischen Cloud-Plattform und Anwendungsimplementierungstool. Welches Tool übernimmt welche Verantwortungsbereiche? Am einfachsten sieht man sich das aus der Perspektive langsamer und schneller Assets im CloudStack an. In der Seitenleiste sehen Sie die unterschiedlichen Ebenen eines Anwendungsstacks von OS-, Speicher und Netzwerkschicht bis hin zur Anwendung.



Die Konfigurationsschichten von Anwendung, Daten und Middleware bewegen sich von Natur aus schnell. Diese verändern sich häufig, da sich die Anwendung, deren Daten und Verwendung wiederholen. Die Geschwindigkeit dieser Änderungen kann auch für eine noch in der Entwicklung befindliche Anwendung sehr hoch sein. Die unteren Schichten darunter beinhalten Middleware (Anwendungsserver, Datenbank, usw.), OS und Speicher und diese verändern sich nicht so häufig. Da das Aktualisieren oder Re-Provisioning aller Schichten für eine einfache Änderung, die sich nur in der Anwendung, deren Inhalt oder Konfiguration auswirkt, nicht sehr effizient wäre, ist es sinnvoll, die Aufgaben nach schnellen Schichten und langsamen Schichten bzw. Anwendungsimplementierungstool und Cloud-Management-Tool zu trennen. Die sich schnell bewegenden Schichten werden also mithilfe des Anwendungsimplementierungstools verwaltet und automatisiert und die langsamen Schichten mithilfe einer Cloud-Management-Software, die durch die Cloud-Plattform zur Verfügung gestellt wird.

PaaS

Mit einem PaaS-Cloud-Modell sind Sie als Benutzer nur für die Anwendung und Daten verantwortlich. Alles andere übernimmt die Cloud-Plattform. Im Ergebnis können Sie Ihren Entwicklerteams eine verbesserte Anwendererfahrung bieten. Die Anwendungsimplementierungs- und Testtools sind nun als Services über die Plattform verfügbar, auf die Ihre Fachkräfte zugreifen können. Das Unternehmen selbst ist nicht länger für das Management der Delivery Pipeline zuständig. Stattdessen ist diese Aufgabe in die PaaS-Lösung eingebunden und Ihre Experten können sich voll und ganz auf die schnelle Bereitstellung von Anwendungen konzentrieren. Entwicklungs- und Testtools sowie die Infrastruktureinrichtung stehen Ihren Fachkräften bereits als Services zur Verfügung, diese können sich also einzig und allein um ihre Kernaufgabe, nämlich die Entwicklung von Anwendungen, kümmern.

IBM Bluemix ist eine solche PaaS-Lösung. IBM und seine Partnerunternehmen verwalten die Plattform und alle darauf zur Verfügung stehenden Services. Die Plattform verfügt über eingebettete IBM DevOps Services, sprich eine Reihe von Services, welche Ihre Mitarbeiter zur Einführung von DevOps benötigen, sowie insbesondere eine anwendungsbereite Delivery Pipeline in Form verschiedener Services. Entwicklerteams können diese Services nutzen, ohne sich Gedanken darüber machen zu müssen, wie diese gehostet sind und zur Verfügung gestellt werden. Zu den DevOps Services gehört Folgendes:

- Webbasierte IDE-Umgebung (Integrated Development Environment) als ein Service
- Build als ein Service
- Planungs- und Aufgabenverwaltung als ein Service
- Sicherheitsscans als ein Service
- Implementierung als ein Service
- Monitoring und Analyse als ein Service

Die Plattform bietet zudem skalierbare Laufzeit-Umgebungen für Anwendungen, die innerhalb des Entwicklungszyklus, sprich von Entwicklung, über Test und Staging bis hin zur Produktion, in unterschiedlichen Umgebungen laufen.

Was ist eine Hybrid Cloud

Der Begriff Hybrid Cloud ist in Bezug auf die Cloud-Technologie weit verbreitet. Wahrscheinlich ist er mittlerweile schon ein wenig abgenutzt, um verschiedene Cloud-Szenarios zu beschreiben, in denen entweder mehrere Cloud-Technologien nebeneinander existieren oder in denen Cloud- und physische Infrastrukturen nebeneinander existieren. Sehen wir uns die unzähligen Cloud-Szenarios einmal näher an, um den Begriff Hybrid Cloud zu definieren:

- **Cloud- und physische Infrastruktur:** Das ist ein weit verbreitetes hybrides Cloud-Szenario. Sofern ein Unternehmen nicht schon in der Cloud gegründet wird, ist das der Standard. Alle Unternehmen haben Workloads und Anwendungen, die derzeit über ihre bestehenden physischen Infrastrukturen laufen. Häufig werden einige Anwendungen auch weiterhin über diese Infrastrukturen betrieben. Typische Beispiele sind Mainframe-Anwendungen und datenintensive Speicheranwendungen, die aufgrund von technologischen Zwängen oder aus Kostengründen nicht in die Cloud verlagert werden. Selbst wenn ein Unternehmen alle Workloads in die Cloud verlagert, kann die Migration nicht über Nacht erfolgen, demzufolge existieren physische und Cloud-Infrastruktur eine längere Zeit nebeneinander.
- **On-Premise- und Off-Premise-Cloud:** Bei diesem Szenario setzt ein Unternehmen für einige Anwendungen und Workloads auf eine Off-Premise-Cloud (also öffentlich oder virtuell privat) und bei anderen Aufgaben auf eine On-Premise-Lösung (also eine private Cloud). Ein Beispiel: Ein Unternehmen möchte seine Entwicklungsumgebung in einer kostengünstigen Off-Premise-Cloud einrichten und eine selbst verwaltete private (On-Premise-)Cloud im eigenen Rechenzentrum für alle Produktionsaufgaben nutzen.
- **IaaS und PaaS:** Dieses Szenario gilt für Kunden, die für einige Workloads mit einem PaaS-Cloud-Service-Modell arbeiten, bspw. neue innovative Systeme für so genannte Engagement Apps, und IaaS für traditionellere Aufgaben wie Speicherworkloads einsetzen.

Bei der Einführung von DevOps stellt eine Hybrid Cloud eine neue Herausforderung dar, da dies zu einer Delivery Pipeline führt, die sowohl auf einer komplexen hybriden Cloud als auch in physischen Umgebungen beheimatet ist. Beispiele für eine solche Hybrid Cloud-Umgebung sind:

- Ein Unternehmen nutzt eine öffentliche Cloud für Entwicklung, Tests und weitere Nicht-Produktions-Umgebungen, während es gleichzeitig bei der Produktion auf eine On-Premise-Cloud oder sogar eine physische Infrastruktur zurückgreift.
- Ein Unternehmen hat verschiedene Engagement Apps entwickelt und in einer Cloud-Umgebung implementiert, während das System der Speicheranwendungen im Backend für die wichtigsten Unternehmensanwendungen noch immer auf einer physischen Infrastruktur liegt, z. B. einem Zentralrechner.
- Ein Unternehmen nutzt eine öffentliche PaaS-Lösung für das Experimentieren mit innovativen Anwendungen und möchte diese auf eine private Cloud verlagern, sobald diese erfolgreich funktionieren.
- Ein Unternehmen möchte seine Anwendungsworkloads über verschiedene Cloud-Plattformen verteilen können, um einen Lock-in-Effekt auszuschließen oder die Möglichkeit zu haben, kritische Workloads in Clouds verschiedener Anbieter bereitstellen zu können.

Die wichtigste Voraussetzung für die Einführung von DevOps in einer hybriden Cloud ist die Frage der Anwendungsimplementierung in diesen unterschiedlichen Cloud- und physischen Umgebungen. Anwendungen wie IBM UrbanCode Deploy with Patterns verwenden Anwendungs-Blueprints, um Anwendungen und Konfigurationen in unterschiedlichen Umgebungen, ob physisch oder Cloud, zuordnen zu können und damit eine automatisierte Anwendungsimplementierung in komplexen hybriden Cloud-Umgebungen zu ermöglichen.

Kapitel 5

Mit DevOps neue Herausforderungen bewältigen

In diesem Kapitel

- Mobile Anwendungen ermöglichen
- ALM-Prozesse unterstützen
- Agiles Handeln ausweiten
- Mehrschichtige Anwendungen verwalten
- DevOps auf der Unternehmensebene
- Mit Lieferketten arbeiten
- Durch das Internet of Things (Internet der Dinge) navigieren

DevOps entstand in so genannten *Born on the Web*-Unternehmen (also Firmen, die es ohne das Internet überhaupt nicht gäbe) wie Etsy, Flickr und Netflix. Diese Unternehmen beschäftigen sich mit der Lösung hochkomplexer technologischer Probleme, arbeiten aber selbst in vergleichsweise einfachen Architekturen. Große Konzerne hingegen, die rund um ihre alten IT-Systeme immer größer wurden und/oder aufgrund von Übernahmen und Zusammenschlüssen wuchsen, setzen auf komplexe multitechnologische Systeme, die dann wieder miteinander koordiniert werden müssen. Diese Situation wird durch Anforderungen weiter verschärft, denen sich moderne Unternehmen durch neue Technologien, wie mobile Services und Modelle zur Anwendungsbereitstellung sowie Software-Lieferketten, gegenüber sehen.

In diesem Kapitel beschäftigen wir uns mit einigen dieser Herausforderungen und erfahren, wie Unternehmen diese mithilfe von DevOps bewältigen können.

Mobile Anwendungen

In einem Unternehmen sind mobile Anwendungen in der Regel keine alleinstehenden Apps. Das mobile Endgerät selbst erfordert wenig Logik und dient mehr als Frontend diverser Anwendungen, die das Unternehmen bereits verwendet. Diese Unternehmensanwendungen im Backend können Transaktionsverarbeitungssysteme, Mitarbeiterportale oder Systeme zur Kundengewinnung umfassen. Mobile Entwicklung und Bereitstellung ist ein komplexes Thema und erfordert eine Reihe voneinander abhängiger Services, die sowohl miteinander koordiniert werden als auch zuverlässig und effizient sein müssen.

Für mobile Apps müssen Release-Zyklen und neu freizugebende Funktionen mit denjenigen der Unternehmensanwendungen und Services koordiniert werden, auf die die mobile App zugreift. Aus diesem Grund sollte im Rahmen einer DevOps-Einführung ein spezialisiertes Entwicklungsteam für mobile Apps neben den übrigen Entwicklungsteams des Unternehmens absolute Priorität haben.

DevOps und App-Stores

Ein einzigartiges Merkmal mobiler Apps ist die Notwendigkeit, diese über so genannte App-Stores bereitstellen zu müssen. Die meisten mobilen Apps können nicht direkt auf einem mobilen Gerät implementiert werden, stattdessen läuft ihre Installation über einen zwischengeschalteten App-Anbieter. Apple hat diese Vertriebsform mit seinem AppStore begründet (und gleichzeitig seine Geräte gesperrt, um eine direkte Installation von Apps durch andere App-Entwickler oder -Anbieter zu verhindern). Gerätehersteller wie Research In Motion, Google und Microsoft, die zunächst die direkte Installation von Apps zuließen, setzen inzwischen ebenfalls auf das Modell von Apple.

Dadurch wird der Implementierungsprozess unterbrochen und erfordert einen zusätzlichen Schritt, denn Entwickler können damit eine App nicht einfach nach Bedarf updaten. Selbst bei wichtigen Fehlerbehebungen muss eine neue App-Version zunächst zur Prüfung an den App-Store eingereicht werden. Damit funktioniert das Modell der kontinuierlichen Bereitstellung nicht mehr wie gewünscht und wird verzögert. Die kontinuierliche Implementierung zu Entwicklungs- und Testzwecken bleibt jedoch bestehen und die Testumgebungen fungieren nach wie vor als Simulator für die Geräte, auf denen die Anwendung schlussendlich laufen soll.

Weltweit laufen 80 Prozent aller Unternehmensdaten über einen Zentralrechner und 70 Prozent aller Transaktionen werden in irgendeiner Form über solche Großrechner abgewickelt. Ein mobiler Pfad zu diesen Ressourcen kann die Art und Weise, wie Ihr Unternehmen bislang Geschäfte gemacht oder mit seinen Kunden interagiert hat, völlig verändern. Der Weg dahin kann allerdings steinig und lang sein. Eventuell fehlen Ihren Mitarbeitern die richtigen Fähigkeiten oder einzelne Abteilungen arbeiten zu isoliert voneinander und operieren über diverse Plattformen. Das alles kann zu langen Release-Zyklen, unnötigen Verzögerungen und der Verschwendung von Ressourcen führen. Doch mithilfe von DevOps, einem Ansatz zur Softwarebereitstellung, der auf Geschwindigkeit und Effizienz setzt, ohne dabei die Stabilität und Qualität eines Angebots zu gefährden, können Unternehmen in die mobile Zukunft aufbrechen.

Es gibt kein spezielles DevOps-Konzept oder -Prinzip, das einzig und allein für mobile Apps gilt. Bei mobilen Apps ist es allerdings noch wichtiger, den Entwicklungsprozess möglichst kurz zu halten und unverzüglich auf Veränderungen zu reagieren.

ALM-Prozesse

Application Lifecycle Management (kurz ALM für Anwendungslebenszyklusmanagement) umfasst eine Reihe von Prozessen, die der Verwaltung eines Anwendungslebenszyklus dienen, und von der ersten Idee (einer geschäftlichen Notwendigkeit) über die Bereitstellung der Anwendung und ggf. bis hin zur Wartung reichen. Wenn man DevOps als einen durchgängigen Prozess im Unternehmen betrachtet, ist ALM folglich der Grundgedanke hinter dem DevOps-Prozess. DevOps erweitert die Bedeutung von ALM und schließt Unternehmer, Kunden und den eigentlichen Betrieb als Teil des Gesamtprozesses ein.

Die Einsatzmöglichkeit von DevOps im Bereich Entwicklung/Test (weitere Infos siehe Kapitel 2) kommt dem herkömmlichen ALM-Ansatz mit Anforderungsmanagement, Veränderungsmanagement, Versionskontrolle, Rückverfolgbarkeit und Testmanagement am nächsten. Weitere ALM-Prozesse wie Tracking und Planung sind Teil des Plan-Einführungsmodells. Themen, wie Dashboards und Berichterstattung sind im Operations-Einführungsmodell beinhaltet.

Agiles Handeln ausweiten

Entwicklung nach Lean-Prinzipien und agilen Grundsätzen sind das Fundament des DevOps-Ansatzes, was u. a. in der Reduzierung überflüssiger Aufgaben durch effizientere Teams resultiert. Effizienz und der wiederholte Einsatz bewährter Praktiken führen zu kürzeren Entwicklungszyklen und ermöglichen es Mitarbeitern, innovativer und

verantwortungsvoller zu handeln - und dies wiederum führt zu einer größeren Wertschöpfung auf Seiten der Kunden. Agiles Handeln und Lean-Prinzipien auf andere Teams jenseits der Entwicklungsabteilung auszuweiten und dieses Denken im gesamten Produkt- und Softwarelebenszyklus zu verankern, das ist der Kern von DevOps.

Viele Abteilungen denken und handeln bereits agil und möchten dies im Rahmen der DevOps-Einführung auf weitere Prozesse ausdehnen. Es gibt eine Reihe weit verbreiteter Ansätze, um genau das zu tun. Dazu gehören Ideen wie Scaled Agile Framework (SAFe) und Disciplined Agile Delivery (DAD). In einigen Unternehmen hat sich auch der Scrum-Prozess für sehr große Abteilungen bewährt. Sinn und Zweck aller dieser Ansätze ist es, Ihnen ein Regelwerk an die Hand zu geben, mit dessen Hilfe Sie agiles Denken und Handeln in Ihrem Unternehmen umsetzen können. Das bedeutet, dass nicht nur die Entwicklung des Codes von Bedeutung ist, sondern genauso die Architektur, die Projektfinanzierung und die Steuerung aller durch das Management vorgesehenen Prozesse und Rollen. Dabei gelten für alle Bereiche dieselben Lean-Prinzipien und agilen Grundsätze, die zuvor schon auf Ebene des einzelnen Teams funktioniert haben. Ganz gleich, auf welchen Ansatz Sie dabei vertrauen, das Prinzip ist immer dasselbe: Mithilfe dieser agilen Grundsätze und dem Einsatz bewährter Praktiken kann Ihr gesamtes Unternehmen noch effizienter werden.

Mehrschichtige Anwendungen verwalten

In einer IT-Abteilung normaler Größe treffen wir in der Regel auf mehrschichtige Anwendungen, die viele Plattformen umfassen, wobei jede eines einzigartigen Entwicklungsprozesses, eigener Tools und Fähigkeiten bedarf. Diese mehrschichtigen Systeme integrieren häufig Anwendungen im Web, auf dem Desktop sowie mobile Anwendungen im Frontend. Desweiteren integrieren sie Systeme, die im Backend laufen, wie Anwendungspakete, Data-Warehouse-Systeme oder Anwendungen, die über Zentral- oder Midrange-Rechner laufen. Selbst eine äußerst gut organisierte IT-Abteilung kann daran scheitern, Releases der einzelnen Teile dieser mehrschichtigen Systeme zu verwalten und zu koordinieren, zumal wenn diese auf unterschiedlichen Plattformen angesiedelt sind.

Ein vernünftiger Ansatz wäre es daher, sämtliche Build-, Konfigurations- und Bereitstellungsprozesse in allen Entwicklungsphasen zu automatisieren und einheitlich zu organisieren. Damit stellen Sie sicher, dass Sie alle Bestandteile berücksichtigen, die Sie benötigen - aber eben auch nur diese. Zudem ist somit gewährleistet, dass die Anwendung auch bei Änderungen als Ganzes bestehen bleibt

während das Projekt Test, QA und Produktion durchläuft. IBM UrbanCode Deploy verfügt über ein Anwendungsmodell, mit dessen Hilfe das komplexe Deployment von mehrschichtige Anwendungen automatisiert werden kann.

Unterschiedliche Tools für unterschiedliche Teams, die nur auf einer Plattform funktionieren, sind in unserer heutigen Welt mit zahlreichen Plattformen und Anbietern Realität. Doch mit einer offenen Plattform wie IBM Jazz können verschiedenartige Tools integriert werden, um eine einheitliche Lösung zu bieten. Indem Sie Bereitstellungsverfahren vereinheitlichen, können Ihre Mitarbeiter neue Software zuverlässig, wiederholbar und plattformunabhängig entwickeln und damit zu einer echten Wertschöpfung beitragen.

DevOps auf der Unternehmensebene

Heutige Unternehmen sind stark davon abhängig, wie schnell die IT Software bereitstellen kann. Viele Unternehmen setzen auf Speicheranwendungen (selbst entwickelt oder Anwendungspakete), die auf Zentral- oder Midrange-Rechnern laufen. Dieses Vorgehen stellt ein Unternehmen vor diverse Herausforderungen:

- Regulatorische Hürden
- Prozesskomplexität
- Mangel an Fähigkeiten
- Mitarbeiter arbeiten isoliert voneinander
- Plattformen und Tools führen zu langen Release-Zyklen, unnötigen Verzögerungen und der Verschwendung von Ressourcen

DevOps auf Unternehmensebene ermöglicht den Beteiligten in Planung, Entwicklung, Test und Operations eine kontinuierliche Softwareentwicklung innerhalb ihrer Organisationen. Unternehmen implementieren heute Anwendungen, die in der Regel wirklich plattformübergreifend sind, sprich von mobilen Anwendungen bis zum Zentralrechner reichen. Mithilfe des DevOps-Ansatzes und Lean-Prinzipien bei der Entwicklung ist eine effiziente und wirksame Delivery Pipeline möglich. Diese ermöglicht es, dass Anwendungen qualitativ hochwertiger und schneller entwickelt, getestet und geliefert werden können, während gleichzeitig die Entwicklungskosten sinken.

Angesicht der diversen Plattformen, der Verbreitung von mobilen und Cloudanwendungen, der verteilten Architektur und Zentralrechneraufgaben, die allesamt entwickelt, integriert, implementiert und betrieben werden müssen, kann DevOps heute in Sachen Effizienz, Rationalisierung und Teamarbeit den Unterschied ausmachen.

Lieferketten

Lieferketten bei der Softwareentwicklung werden inzwischen zur Norm, da immer mehr Unternehmen auf Outsourcing und strategische Partnerschaften setzen, um alle notwendigen Kenntnisse und Funktionen abzudecken. Eine *Lieferkette* ist ein System von Organisationen, Menschen, Technologien, Aktivitäten, Informationen und Ressourcen, die dazu dient, ein Produkt oder eine Dienstleistung vom Hersteller zum Kunden zu bringen. Die diversen Zulieferer in dieser Kette können unternehmensintern oder -extern arbeiten.

Für ein Unternehmen, das als Softwareanbieter auf ein solches Lieferkettenmodell setzt, kann die Einführung von DevOps eine Herausforderung werden, da die Beziehungen zu den einzelnen Zulieferern mehr über Verträge und Service-Level-Agreements geregelt sind als über Kooperation und Kommunikation ablaufen. Nichtsdestotrotz kann auch ein solches Unternehmen von DevOps profitieren. Das Kernprojektteam verantwortet die Planung und Bewertung, während andere Aufgaben durch Zulieferer übernommen werden. In der Delivery Pipeline selbst können unterschiedliche Zulieferer unterschiedliche Phasen übernehmen. Es ist daher zwingend, dass alle Beteiligten auf gemeinsame Tools und ein Asset-Repository zurückgreifen können. Mithilfe eines Tools zur Arbeitsaufgabenverwaltung sind bspw. Berichte über die Aufgaben, die gerade in Arbeit sind, für sämtlichen Zulieferer verfügbar. Zudem können den Zulieferern darüber Aufgaben zugewiesen werden. Mit einem gemeinsamen Asset-Repository können Assets über die Delivery Pipeline weitergegeben und damit eine kontinuierliche Bereitstellung gewährleistet werden.

Internet of Things (Internet der Dinge)

Der nächste Schritt mit DevOps ist die Ausweitung auf Systeme oder eingebettete Geräte. In diesem Zusammenhang spricht man auch von *Continuous Engineering.* Zu Beginn des Internets wurden die meisten Daten von Menschen erzeugt. Heute jedoch werden viel mehr Daten durch unzählige Geräte (wie Sensoren und Aktuatoren) erzeugt als durch den Menschen selbst. Dieses Netzwerk miteinander verbundener Geräte im Internet wird auch als das *Internet of Things (Internet der Dinge)* bezeichnet.

In diesem Zusammenhang wird DevOps wahrscheinlich sogar noch mehr Bedeutung erlangen, da die Hardware und die darin eingebettete Software nebeneinander existieren. Im Continuous Engineering spielen die DevOps-Grundsätze eine große Rolle, da gewährleistet sein muss, dass die eingebettete Software auf den Geräten höchsten Qualitätsansprüchen und korrekten technischen Spezifikationen entspricht.

Der Bereich „Operations“ wird im Continuous Engineering durch Hardware ersetzt oder durch Systemingenieure, die für die Geräte kundenspezifische Hardware entwickeln. Entscheidend ist die Zusammenarbeit zwischen Entwicklungs- und Testteam sowie den Systemingenieuren, damit die Entwicklung von Hardware und Software koordiniert werden kann, anstelle einer Entwicklung, bei der Hardware und Software unterschiedlichen Lieferzyklen folgen. Die Anforderungen einer kontinuierlichen Bereitstellung und von kontinuierlichen Tests bleiben für die Umsetzung einer kontinuierlichen Auslieferung bestehen. Während der Entwicklung werden Software und Hardware mithilfe von Simulatoren getestet.

Anti-Patterns

In der Praxis stehen der DevOps-Einführung natürlich immer Einschränkungen entgegen. Einige sind in der Branche oder Umgebung begründet, in der ein Unternehmen agiert, wie bspw. die Einhaltung gesetzlicher Vorgaben, komplexe Hardwaresysteme oder unausgereifte Lieferfähigkeiten. In einem solchen Fall muss bei der Einführung von DevOps das Thema *Anti-Patterns* (ineffiziente oder kontraproduktive Muster) berücksichtigt werden, die für ein Unternehmen aufgrund seiner Notwendigkeiten schlicht nicht akzeptabel sind.

Water-SCRUM-Fall

Forrester (`www.forrester.com`), ein global agierendes Marktforschungs- und Beratungsunternehmen hat den Begriff *Water-SCRUM-Fall* geprägt, um den aktuellen Stand bei der Einführung agiler Software-Entwicklungsmethoden zu beschreiben. Aus DevOps-Sicht bedeutet das, dass die Entwicklungsabteilung schon nach agilen Methoden arbeitet, alle anderen rings herum aber noch manuell nach dem Prinzip des Wasserfalls arbeiten und damit keine kontinuierliche Bereitstellung gewährleisten können. Diese Situation entsteht in einigen Unternehmen durch Ihre Unternehmenskultur. Daher muss ein Unternehmen, dass DevOps einführen will, solche manuellen Prozesse in die umfassendere DevOps-Verfahren einbetten.

NoOps

In so genannten NoOps-Organisationen ist das Operations-Team eine separate Abteilung, deren Verantwortungsbereich mit dem der Entwicklung verschmolzen wird. Das Unternehmen Netflix, das Online-TV auf Abruf anbietet, ist ein Beispiel dafür. NoOps kann für einige Unternehmen durchaus eine Option sein. Allerdings bleibt wohl noch abzuwarten, ob dieses Modell auch auf breiterer Ebene überzeugen kann.

Kapitel 6

So funktioniert DevOps: Aus der Sicht von IBM

In diesem Kapitel

- Best Practices für Führungskräfte
- Mein Team organisieren
- DevOps-Ziele bestimmen
- Der DevOps-Transformation Beachtung schenken
- Von den DevOps-Ergebnissen lernen

IBM arbeitet derzeit unternehmensweit an der Umsetzung des DevOps-Ansatzes, der sich unterdessen stetig weiterentwickelt. Diese Umsetzung basiert insbesondere auf einem DevOps-Ansatz, der zuerst bei Rational in der IBM Software Group (SWG) entwickelt sowie erfolgreich erprobt wurde und nun auch bei Watson, Tivoli, Global Business Services und weiteren Konzernsparten zum Einsatz kommt. In diesem Kapitel finden Sie eine Fallstudie zur Umsetzung von DevOps durch das IBM Rational Collaborative Lifecycle Management Produkt-Team.

Diese Art der Softwareentwicklung ist einzigartig, da sie *offen* stattfindet. Das Softwareentwicklungsteam liefert all seine Entwicklungsartefakte und laufenden Arbeitsprozesse einschließlich aller Einzelheiten über `jazz.net`. Diese Webseite ist öffentlich zugänglich und jeder registrierte Benutzer kann sich über geplante Projekte, laufende Prozesse und über den Verlauf der gesamten Entwicklungsarbeit zu einem Softwareprodukt informieren.

Welche Rolle spielt die Führungskraft

Die Unternehmenskultur zieht sich wie ein roter Faden durch eine Organisation. Dazu gehören Werte und Verhaltensweisen, wie sie

Management aber auch Mitarbeiter vorleben. Meistens versteht man die Kultur eines Unternehmens nicht zu 100 Prozent, erst wenn eine große Veränderung ansteht, wird sie überdeutlich. Es gibt die Gruppe der Skeptiker, die sich erst einmal zurück lehnen und abwarten, ob es sich nicht nur um eine vorübergehende Modeerscheinung handelt. Führungspersönlichkeiten treten hervor. Es ist unabdingbar, dass Sie ein Konzept entwickeln, um diese Dynamik in der Gruppe zu verstehen und herauszufinden, wer wohin gehört - damit Sie diejenigen angehen können, die der Veränderung im Wege stehen.

Das IBM SWG Management hat sich diverse Ansätze zu Nutze gemacht, um diese kulturelle Dynamik in den Griff zu bekommen:

- **Die richtige Führungskraft auswählen:** In der Rolle der Führungskraft muss man die unterschiedlichen Standpunkte auf einen Nenner bringen, damit das Team sich über Ziele, Hindernisse, Prozessveränderungen und Entscheidungen einig ist, mit denen begonnen werden soll.
- **Akteure einbeziehen:** Die Unterstützung für solche Veränderungen muss aus dem Vorstand, dem Management und auch von einzelnen Projektpartnern aus den unterschiedlichen Entwicklungsdisziplinen kommen. Achten Sie darauf, Akteure aus allen Sparten, also Architekten, Entwickler, Tester und Operations-Team einzubeziehen und mit namhaften Experten aus diesen Abteilungen zusammenzuarbeiten, die in Sachen Veränderung positiv hervortreten.
- **Verbesserungen und Ergebnisse bewerten:** Es ist entscheidend, dass Sie Kennzahlen festlegen, die sich sowohl an den gewünschten Auswirkungen als auch den Geschäftsergebnissen orientieren müssen. Diese Ziele und Kennzahlen sollten möglichst hoch gesteckt werden und alle Mitwirkenden in die Verantwortung nehmen, sie sollten aber gleichzeitig nicht unerreichbar sein.
- **Impulse für frühzeitige Erfolge setzen:** Indem Sie in jedem Bereich ineffizientes Arbeiten nachvollziehbar und Verbesserungen sichtbar machen, setzen Sie Impulse für eine Veränderung.
- **Kommunizieren und zuhören:** Als Führungskraft müssen Sie erkennen, mit welcher Dynamik sich die Veränderung in Ihrem Team wirklich durchsetzt. Verwenden Sie Ihre Zeit darauf, mit den technischen Teams, dem Management und der Geschäftsführung im persönlichen Einzelgespräch sowie bei regelmäßigen Meetings über den zusätzlichen Nutzen für das Team zu sprechen und Sichtweisen auf mögliche Hindernisse zu erkunden. Genauso wichtig: Auf diese Weise kann auch das Management seine Sicht auf Prioritäten und Fortschritte kundtun.

Als Führungskraft sollten Sie einerseits die Teams unterstützen und andererseits zur Verfügung stehen, um Probleme zu klären und Hindernisse zu überwinden. Nur wenn Sie als ein Team mit klaren Geschäftszielen arbeiten, können Sie alle gemeinsam an einem Strang ziehen.

Das Team zusammenstellen

Das IBM SWG Rational Collaborative Lifecycle Management Produkt-Team ist Teil einer größeren Gruppe von Mitarbeitern, die an mehr als 80 Softwareentwicklungstools in den Kategorien SoftwareDelivery, Planung, Softwareentwicklung, Anwendungsbereitstellung, Software-Qualitätsmanagement sowie Anwendungsmonitoring und -analyse arbeiten.

Dieses IBM SWG Produkt-team ist eine große, global tätige Organisation mit vier Kernteams, die an mehr als 25 verschiedenen Orten in 10 Ländern tätig sind. Bevor das Team auf den DevOps-Ansatz umstieg, arbeitete es auf Grundlage eines jährlichen Releaseplans, einschließlich einer Vorlaufzeit von drei bis sechs Monaten, in der festgelegt wurde, was tatsächlich in den Release hineinkommt.

DevOps-Ziele festlegen

Das IBM SWG Team hatte den Eindruck, dass es zu lange dauerte, um auf Veränderungen im Markt als auch auf neue Anforderungen seitens der Kunden reagieren zu können. Daher entschloss sich das Team zu einer Verkürzung des Delivery Zyklus und das nicht nur bei Entwicklung und Testphasen, sondern bereits in der Zusammenarbeit und Interaktion mit Partnern und Kunden. Das Ziel: Statt einem jährlichen Releaseplan sollte es einen Quartalsreleaseplan geben.

Um häufiger neue Funktionen entwickeln zu können, musste das Team seine Entwicklungsarbeit nicht nur schneller vorantreiben, sondern auch zügig Cloud-Bereitstellungsmodelle, mobile Entwicklungs- und Testmöglichkeiten sowie weitere Fähigkeiten ausbauen, um auf technologische Neuerungen reagieren zu können. Aus diesem Grund entschied sich das Team zur Einführung der Prinzipien und Methoden von DevOps, um die gesamte Softwareentwicklung der Gruppe zu verändern und damit den Kunden schneller und häufiger einen echten Mehrwert zu bieten.

Eine solch weitreichende Veränderung erfordert einen kulturellen Wandel innerhalb der Organisation. Deshalb wurden vier Arbeitsgruppen gebildet, die sich aus Mitarbeitern des Managements und technischen Leitern zusammensetzen. Diese Arbeitsgruppen überprüften den Prozess der Softwareentwicklung von A bis Z und verantworteten

die Veränderungen in der Arbeitsweise des Unternehmens. Um die wichtigsten Punkte im Entwicklungsprozess angehen zu können, wurden eine Reihe von Maßnahmen und Aktionsplänen festgelegt. Es wurde ein so genanntes Delivery-Champion-Team etabliert, dem auch ein Evangelist angehört, der andere Teams ausbildete und Best Practices im gesamten Unternehmen weitergab.

Das IBM SWG Team hat sich die Umsetzung des DevOps-Ansatzes auf die Fahnen geschrieben und will dieses Ziel wie folgt erreichen:

- Rationalisierung des Prozesses und Einführung neuer Methoden
- Wirksamer Einsatz von Tools mit dem Ziel der Übereinstimmung und Skalierbarkeit für andere Teams sowie Nachvollziehbarkeit und Messbarkeit
- Eine Kultur der ständigen Optimierung etablieren

Von der DevOps-Transformation lernen

In diesem Kapitel erfahren Sie, welche Schritte das IBM SWG Team unternommen hat, um die DevOps-Transformation auf den Weg zu bringen.

Agile Prozesse ausbauen

Bereits bestehende agile Prozesse wurden erweitert und auch außerhalb der der eigentlichen Entwicklungs- und Testverfahren eingesetzt. Durch die gezielte Einbeziehung von Kunden, Entwicklungs- und Operations-Teams gelang es verhärtete Strukturen aufzubrechen und die Ergebnisse wesentlich zu verbessern. Dieses umfassende agile Modell ermöglicht Teams eine engere Zusammenarbeit, um dauerhaft eine qualitativ hochwertige Software entwickeln zu können, die dem Unternehmen einen Mehrwert bringt, indem für jeden einzelnen Arbeitsschritt eine Reihe integrierter Prozessen genutzt wird.

Nach dem Motto „Alles von einem Team“ wurden Produktmanagement, Design und Entwicklung kombiniert. Das Entwicklerteam brachte die traditionelle Rollenverteilung von Entwicklungsmanagern und -teamleitern ein, übernahm zusätzlich aber auch Teile der operativen Führung und arbeitet mit Softwarearchitekten zusammen, um der Idee eines durchgängigen Lebenszyklus gerecht zu werden.

Es wurden dedizierte Ausbilder und Mentoren bereitgestellt, um den Teams das Konzept der agilen und kontinuierlichen Bereitstellung schnell und trotzdem nachhaltig zu vermitteln. Indem der Fokus auf Funktionen versus Produktkomponenten gelenkt wurde, arbeiten die

Mitarbeiter nicht mehr isoliert voneinander, sondern ermöglichen mit jedem Lauf und jeder Automatisierung eine Bereitstellung. Diese Featureteams wurden zudem unterstützt durch Entwicklungsmanager, die dezidiert für diese Aufgabe eingesetzt wurden. Auf allen Ebenen des Entwicklungsprozesses wurden regelmäßige Scrum-Meetings abgehalten, um Probleme zu erkennen und zu lösen, wesentliche Kennzahlen im Blick zu behalten, Livedaten einzubeziehen und besonders wichtige Informationen zu kommunizieren.

Damit Entwicklungsprioritäten mit aktuellen Marktveränderungen Schritt halten können, wurde ein strategisches Produktkomitee einberufen, das aus Produktmanagement, Entwicklungsleitern, Softwarearchitekten und Unternehmensleitung besteht. Dieses Gremium ist verantwortlich für Folgendes:

- Finanzierung für erfolgreiche Programmumsetzung bereitstellen und absichern
- Programmumsetzung vorantreiben, fördern und unterstützen
- Langfristige Vision und Zielrichtung für das Unternehmen etablieren
- Anwenderberichte für jährliche Releases entsprechend der langfristigen Vision priorisieren

Testautomatisierung voranbringen

Um die herkömmlich langwierigen Backend-Testzyklen zu eliminieren und die Releasequalität zu verbessern, wurde ein agiler durchgängiger Testansatz übernommen, der auf Automatisierung und Virtualisierung setzt. Dafür wurde ein Rhythmus etabliert, der alle vier Wochen mit einer Demo endet sowie Vier-Wochen-Meilensteine vorsieht, die zur Fertigstellung einer für den Kunden nutzbaren Version geführt haben. Ein Rückblick auf jeden einzelnen Meilenstein und das Ausräumen technischer Probleme führen auf lange Sicht zu weniger Ausschuss bei zukünftigen Projekten. Das Motto des IBM SWG Teams war: „Frühzeitig und häufig testen."

Für die Testautomatisierung setzte das Team auf diese Best Practices:

- Automatisieren repetitiver und arbeitsintensiver Tests.
- Automatisieren in Bereichen, die anfällig sind für Fehler.
- Automatisierung für jeden Build; frühzeitig und häufig durchführen.
- Automatisierung entwickeln, die unanfällig für Veränderungen der Benutzeroberfläche (UI) ist - sprich ein Framework, das die UI von den Tests trennt.

- Einfaches Erstellen, Bereitstellen und Aufrechterhalten der Automatisierung.
- Entwicklungsarbeit an den Automatisierungsprozessen in die Planung miteinbeziehen und sicherstellen, dass Entwickler genügend Zeit dafür haben.
- Kennzahlen festlegen, um die Automatisierung bewerten zu können (Sie können nur verbessern, was Sie auch messen können).
- Regelmäßige Neubewertung, ob Ihre Automatisierung Fehler findet und wenn nicht, Umgestaltung vornehmen.

Um die Testautomatisierung zu unterstützen, hat das Team die IBM Rational Test Workbench für Funktions- und Leistungstests eingesetzt. Um häufiger Tests durchführen zu können, war es entscheidend, die Bereitstellung von Builds zu automatisieren. Mithilfe von IBM UrbanCode Deploy konnte das Team die Testdurchführungskosten um 90 Prozent senken, indem die Bereitstellung von Builds automatisiert wurde, was die Automatisierung aller erforderlichen Anwendungen sowie Konfigurationseinstellungen der Datenbankserver einschließt.

Eine Delivery Pipeline einrichten

Das IBM SWG Team richtete eine Delivery Pipeline ein, um von der Idee „Tools-as-a-Service“ zu profitieren und ermöglichten es den Entwicklern auf diese Weise, in gerade einmal 60 Minuten einen Code zu schreiben, diesen zu testen und in einer Produktionsumgebung einzusetzen. Diese Arbeitsschritte haben zuvor ein bis zwei Tage in Anspruch genommen. Dadurch sind weniger Nachbesserungen erforderlich und die Produktivität wurde maximiert.

Bei der Bereitstellung über eine kontinuierliche Delivery Pipeline haben sich folgende Best Practices bewährt:

- Shift-Left-Integration und Automatisierung soweit es möglich ist
- Überall dieselben Entwicklungsmechanismen verwenden
- Der aktuelle Arbeitsstand sollte jederzeit auslieferbar sein
- Infrastruktur wie Code behandeln

In Abbildung 6-1 können Sie die Produkte und Funktionen sehen, die als Bestandteil der kontinuierlichen Delivery Pipeline des IBM SWG Teams bereitgestellt wurden.

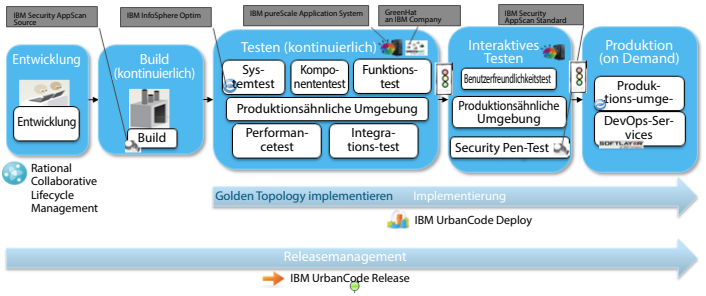


Abbildung 6-1: Die kontinuierliche Delivery Pipeline

Eine der wichtigsten Praktiken bei der Implementierung der kontinuierlichen Delivery Pipeline ist die Maxime „Infrastruktur wie Code behandeln". Das heißt, dass der Entwickler Skripte schreiben kann, um die gewünschte Infrastruktur für diese Anwendung als Teil des Anwendungscodes zu konfigurieren. In der Vergangenheit wurde diese Aufgabe in der Regel von einem Systemadministrator oder IT-Mitarbeiter übernommen, nun jedoch kann der Entwickler dies direkt selbst ausführen. Puppet, Chef und IBM UrbanCode Deploy with Patterns sind Beispiele für neue Infrastrukturautomatisierungstools, die die Idee „Infrastruktur als Code" in die Praxis umsetzen.

Das IBM SWG Team behandelt die Infrastruktur nun wie einen Code und setzt auf folgende bewährte Verfahren:

- Musterdefinitionen, Skript-Pakete und Services als Code behandeln.
- Alles versionieren.
- Automatisierung der Bereitstellung von Topologiemustern für die Cloud.
- Musterversionen in unterschiedlichen Cloud-Umgebungen verwalten.
- Mustertests automatisieren.
- Katalogressourcen bereinigen, um Streuung zu vermeiden.

Schnell experimentieren

Das Konzept einer kontinuierlichen Auslieferung beinhaltet nicht nur die eigentliche Softwareentwicklung wie z. B. die fortlaufende Integration und kontinuierliche Implementierung, sondern auch das Lernen als fundamentalen Bestandteil. Und das kann nur durch

häufiges Experimentieren und das Bewerten der Ergebnisse erreicht werden.

Wenn Sie eine Anwendung um bestimmte Features und Funktionen ergänzen, können Sie vorher nie genau wissen, ob der Kunde dadurch die erwarteten oder gewünschten Vorteile erzielt. Deshalb ist es für IBM so wichtig, frühzeitig und häufig zu experimentieren und die Kunden um Feedback zu bitten. Nur so können Sie erfahren, was tatsächlich funktioniert hat. Diese Vorgehensweise erleichtert Ihnen die anschließende Entscheidung, jene Features zu verwerfen, die wenig sinnvoll oder vielleicht sogar ein Hinderniss sind. Diese Strategie finden Sie in Abbildung 6-2 bildlich dargestellt.

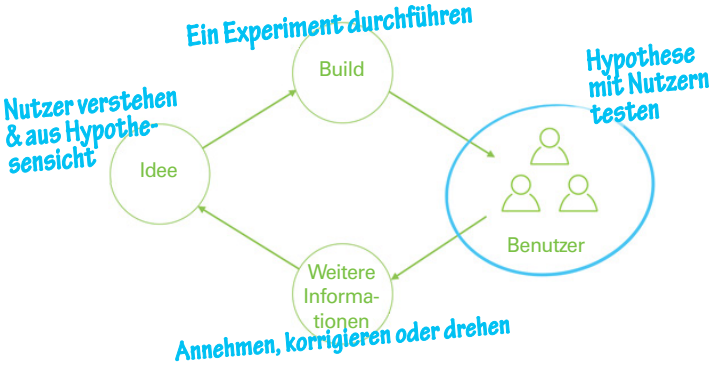


Abbildung 6-2: So sieht eine hypothesengetriebene Entwicklung aus.

Das IBM SWG Team hat beim Thema häufiges Experimentieren enorm dazugelernt und daraus folgende Best Practices entwickelt:

- Kennzahlen und Kriterien für Erfolg/Misserfolg festlegen.
- Durch Experimente herausfinden, was funktioniert und was nicht - also kleine Tests für ein kleine Benutzergruppe, um die Nützlichkeit eines Features zu bestimmen.
- Kontinuierlich Mehrfachexperimente ausführen.
- Schnelle Entscheidungen auf Grundlage von Fakten treffen.
- Schneller entwickeln bedeutet schneller experimentieren können.
- Mechanismen entwickeln, um systemweit experimentieren zu können (Google Analytics, IBM Digital Analytics usw.).
- Unterschiedliche Experimentiermöglichkeiten bedenken (Klassisches A/B-Testen, Multi-Armed-Bandit-Experimente usw.).

- Für vergleichbare Projekte gleichzeitig zweigleisig fahren: An einem Cloud-basierten Projekt experimentieren und die Daten aus diesen Experimenten nicht nur für dieses Projekt nutzen, sondern ebenso für ähnliche interne Projekte.

Kontinuierlich optimieren

Das IBM SWG Team wollte ein Umfeld schaffen, in dem es einfach dazugehört, sich ständig zu verbessern. Das Gleiche gilt für das Messen von Wirksamkeit und Effizienz, um eine tatsächliche Optimierung sicherzustellen. Das Team managt seine kontinuierlichen Optimierungsanstrengungen wie ein agiles Projekt. Indem sich das Team fristgebundene Ziele setzt, Schwachpunkte anspricht und damit verbundene Optimierungsmaßnahmen bestimmt. Durch diese Methode werden Probleme angegangen und jeder Einzelne kann sich so ständig verbessern. Diese ständige Optimierungsarbeit wird genauso beobachtet wie andere Entwicklungsaufgaben, um zu gewährleisten, dass jeder diese Aufgabe auch ernst nimmt. Es kann drei Monate oder länger dauern, bis fristgebundene Ziele (z.B. bestimmte Fertigkeiten) tatsächlich entwickelt und umgesetzt werden. Große Schwachpunkte können vielleicht erst nach Monaten abgemildert oder gar eliminiert werden. Doch in jedem Fall sollten ganz spezifische Verbesserungsmaßnahmen in kleinen Teilschritten bestimmt werden, die innerhalb von einem Monat zu erreichen sind.

Mithilfe von Retrospektiven institutionalisiert das IBM SWG Team seine ständige Optimierung. Eine *Retrospektive* ist ein regelmäßig stattfindender Rückblick auf Dinge, die gut liefen, die weniger gut liefen und was für eine Verbesserung durchgeführt werden muss. Wenn Sie nicht auf solche Retrospektiven setzen, müssten Sie in Ihrer Entwicklungsarbeit bereits eine Perfektion erreicht haben, die es so noch nicht gibt. In einem großen Team sollten Sie eine Hierarchie für Retrospektiven berücksichtigen. Im IBM SWG Team nimmt jedes Komponententeam eine Retrospektive für sich vor. Diese Erkenntnisse dienen als Input für eine Retrospektive auf Anwendungsebene, die wiederum als Grundlage für einen Rückblick auf höherer Lösungsebene dient. Maßnahmen, die aus diesen Retrospektiven hervorgehen, werden als Schwachpunkte zusammen mit den entsprechenden Verbesserungsmaßnahmen dokumentiert, um diese Punkte abzumildern oder zu eliminieren.

Um sicherzustellen, dass sich die Teams auch tatsächlich verbessern, hat das IBM SWG Team Kennzahlen aus unternehmerischer und operativer Sicht festgelegt, um die Wirksamkeit der DevOps-Transformation bewerten zu können. Die Unternehmenskennzahlen beinhalten messbare Verbesserungen in Bezug auf

- Schnellere Auslieferung
- Höhere Kundenzufriedenheit
- Geringere Wartungskosten und gleichzeitig höhere Innovationsausgaben
- Verbesserte Annahme von Seiten der Kunden

Die operativen Kennzahlen beeinflussen die Teameffizienz auf lange Sicht und sollen Folgendes messen:

- Zeit bis zum Start eines neuen Projekts
- Zeit für Builds
- Zeit für Iterationstests

Die Ergebnisse von DevOps

Mit einem DevOps-Ansatz konnte das IBM SWG Team die Kundenzufriedenheit erhöhen, die Annahme eines Produkts von Seiten der Kunden verbessern und ein zweistelliges Umsatzplus verzeichnen. Die kürzeren Zeitvorgaben haben den Teams bei IBM einen regelrechten Energieschub verliehen, was wiederum zu einer schnelleren Entwicklung erweiterter interner Lösungen und neuen Cloud-Services wie Bluemix, DevOps Services for Bluemix und Collaborative Lifecycle Management als ein Managed Service (CLM aaMS) geführt hat.

In Abbildung 6-3 finden Sie ein Beispiel für den Erfolg des DevOps-Ansatzes bei IBM: Die messbaren Ergebnisse, die das IBM SWG Rational Collaborative Lifecycle Management Produkt-Team erreicht hat.

Lebenszyklus Messgrößen	2008	2010	2012-2014	Verbesserung insgesamt
Projektstart	30 Tage	10 Tage	2 Tage	28 Tage
Groomed Backlog	90 Tage	45 Tage	Fortlaufend	89 Tage
Gesamtentwicklungszeit	120 Tage	55 Tage	3 Tage	117 Tage
Build-Erstellung insgesamt	36 Stunden	12 Stunden	5 Stunden	700%
BVT Verfügbarkeit	k.A.	18 Stunden	< 1 Stunde	17 Stunden
Iterationstest insgesamt	5 Tage	2 Tage	14 Stunden	4 Tage
Implementierungszeit insgesamt	2 Tage	8 Stunden	4 Stunden-> 20 Minuten	2 Tage
Produktionszeit insgesamt	3 Tage	3 Tage	2 Tage	7 Tage
Zeit zwischen Releases	12 Monate	12 Monate	3 Monate	9 Monate

Abbildung 6-3: Die messbaren Verbesserungen des IBM SWG Teams.

Kapitel 7

Zehn Mythen über DevOps

In diesem Kapitel

- Verstehen, was DevOps ist
- Verstehen, was DevOps nicht ist

Die DevOps-Bewegung ist noch jung und mitten in der Entwicklung, insbesondere in Unternehmen. Und wie bei jeder neuen Bewegung oder einem neuen Trend, ranken sich darum zahlreiche Mythen und Gerüchte. Einige davon sind sicher auf Unternehmen oder Projekte zurückzuführen, bei denen die DevOps-Einführung missglückt ist. Doch was auf die eine Situation zutrifft, muss nicht zwangsläufig für eine andere gelten. Daher lesen Sie im Folgenden einige der weit verbreiteten Mythen und Fakten zu DevOps.

DevOps ist nur etwas für „Born on the Web" Unternehmen

DevOps wird meist in Zusammenhang mit so genannten „Born on the Web" Unternehmen (Unternehmen, die es ohne das Internet überhaupt nicht gäbe) wie Etsy, Netflix und Flickr genannt wird. Unabhängig davon setzen große Unternehmen bei der Softwareentwicklung jedoch schon seit Jahrzehnten auf Prinzipien und Verfahren, die auch auf DevOps zutreffen. Darüber hinaus haben einige der heute gültigen DevOps-Prinzipien, wie sie in diesem Buch beschrieben werden, inzwischen einen Reifegrad erreicht, der eine Anwendung in Großunternehmen mit diversen Plattformen und verstreut arbeitenden Teams erst möglich macht.

Mit DevOps lernt das Operations-Team das Schreiben von Codes

Operations-Teams haben von jeher mit dem Schreiben von Skripten zu tun, bspw. im Hinblick auf die Umgebungsverwaltung oder repetitive Aufgaben. Doch mit der Entwicklung hin zu einer *Infrastruktur als Code* müssen Operations-Teams diese enormen Codemengen nun auch mithilfe von Software-Engineering-Praktiken wie Versionierung, Check-in, Check-out, Branching und Merging verwalten können. Ein Mitarbeiter des Operations-Teams erzeugt heute eine neue Umgebungsversion, indem er eine neue Version des Codes schreibt, der diese definiert. Das heißt aber nicht, dass das Operations-Team nun wissen muss, wie man einen Code in Java oder der Programmiersprache C# schreibt. Die meisten *Infrastruktur als Code*-Technologien verwenden Sprachen wie Ruby, die relativ einfach sind, insbesondere für jemanden mit Scripting-Kenntnissen.

DevOps ist nur etwas für die Entwicklung und das Operations-Team

Auch wenn der Name auf seinen Ursprung in der Entwicklung bzw. Operations hinweist, geht DevOps alle an. Alle an der Softwareentwicklung Beteiligten, sprich Unternehmensleitung, Fachkräfte, Führungskräfte, Partner, Zulieferer, usw., haben mit DevOps zu tun.

DevOps ist nicht geeignet für ITIL

Einige befürchten, dass DevOps-Ideen, wie die kontinuierliche Bereitstellung, mit den Überprüfungen und Prozessabläufen nach ITIL (Information Technology Infrastructire Library – einer Sammlung von Best Practices zur Umsetzung eines IT Service-Managements) unvereinbar sind. Der Lebenszyklus nach ITIL ist mit DevOps vereinbar. In Wahrheit passen die meisten der ITIL-Prinzipien sogar sehr gut zu denen von DevOps. ITIL hat in einigen Unternehmen jedoch einen schlechten Ruf, da es vorwiegend in Zusammenhang mit langsamen Wasserfall-Prozessen eingeführt wurde, die schnelle Änderungen und Verbesserungen nicht zulassen. Doch bei DevOps geht es genau darum, solche Ansätze zwischen der Entwicklung und dem Operations-Team zu koordinieren.

DevOps ist nichts für staatlich regulierte Branchen

Staatlich regulierte Branchen können ohne allumfassende Kontrollmechanismen nicht arbeiten und sind auf Freigaben durch Mitarbeiter angewiesen, die die Übereinstimmung und Prüfbarkeit solcher Vorgaben verantworten. Mit dem richtigen Einsatz von DevOps kann das Thema Compliance zum Beispiel mithilfe automatisierter Prozesse und Tools, die die Erfassung von Prüfdaten ermöglichen, optimiert werden.

Unternehmen in staatlich regulierten Branchen benötigen immer manuelle Kontrollpunkte oder Prüfstellen, das aber ist mit DevOps vereinbar.

DevOps ist nichts für outgesourcte Entwicklungsarbeit

Outgesourcte Teams können innerhalb der DevOps-Delivery Pipeline als Zulieferer oder Dienstleister betrachtet werden. Unternehmen müssen dabei jedoch sicherstellen, dass die Verfahren und Prozesse dieser Teams mit denen der internen Projektteams kompatibel sind.

Mit einer gemeinsamen Releaseplanung, Aufgabenverwaltung und einem gemeinsamen Asset-Repository können die Zusammenarbeit und Kommunikation zwischen diesen Sparten, Zulieferern und Projektteams entscheidend verbessert und damit auch DevOps ermöglicht werden. Mithilfe von Releaseplanungstools kann ein Unternehmen seine Fähigkeit zur Definition und Koordination des gesamten Releaseprozesses mit allen Beteiligten enorm optimieren.

Keine Cloud, kein DevOps

DevOps und Cloud werden häufig in einem Atemzug genannt, da die Cloud die dynamische Bereitstellung von Infrastrukturressourcen für Entwickler und Tester ermöglicht, um Testumgebungen schnell bereitstellen zu können ohne Tage oder Wochen auf eine manuelle Anfrage warten zu müssen. Dennoch ist eine Cloud nicht entscheidend für den DevOps-Ansatz, solange ein Unternehmen beim Bereitstellen von Ressourcen für das Implementieren und Testen von Anwendungsänderungen über effiziente Prozesse verfügt.

Eine Virtualisierung ist optional. Auch die kontinuierliche Bereitstellung auf physischen Servern ist möglich, solange diese Server der erforderlichen Geschwindigkeit entsprechend konfiguriert und implementiert werden können.

DevOps ist nicht geeignet für große, komplexe Systeme

Gerade komplexe Systeme erfordern die Disziplin und Kooperation, die DevOps mit sich bringt. Denn solche Systeme verfügen in der Regel über verschiedene Software-/Hardwarekomponenten, die jeweils eigene Lieferzyklen und Zeitschienen erfordern. Mit DevOps können diese Lieferzyklen und Releaseplanungen auf Systemebene besser koordiniert werden.

Bei DevOps geht es nur um Kommunikation

Innerhalb der DevOps-Community sind witzige Begriffe entstanden wie *ChatOps* (Teams wickeln die gesamte Kommunikation nur noch über Internet-Chats ab) und *HugOps* (von engl. hug für Umarmen, sprich DevOps legt den Schwerpunkt ausschließlich auf Kooperation und Kommunikation). Diese Begriffe entspringen der falschen Annahme, dass sich sämtliche Probleme allein mit Kommunikation und Kooperation lösen lassen könnten.

DevOps steht und fällt mit der Kommunikation, doch eine bessere Kommunikation in Verbindung mit überflüssigen Prozessen führt noch lange nicht zu besseren Implementierungen.

DevOps steht für Continuous Change Deployment

Diese falsche Annahme rührt von Unternehmen her, die nur Web-Anwendungen bereitstellen. Einige dieser Unternehmen führen auf ihren Webseiten stolz auf, dass sie täglich Implementierungen vornehmen. Doch in großen Unternehmen mit komplexen Anwendungen sind tägliche Implementierungen nicht nur unpraktisch, sondern können im Hinblick auf regulatorische Vorgaben oder Unternehmensrestriktionen auch schlicht unmöglich sein. Bei DevOps geht

es eben nicht nur um das Thema Implementierung und ganz sicher nicht nur darum, kontinuierliche Deployments anbieten zu können. Mit DevOps können Unternehmen die Produktion dann freigeben, wenn sie es möchten und nicht dann, wenn es ein bestimmtes Datum im Kalender verlangt.

Notes

Notes

Notes

Notes

Ohne Software läuft heute nichts mehr.

In unserer schnelllebigen Welt ist DevOps der Ansatz für alle Unternehmen, die schlank und agil sein möchten. DevOps ermöglicht Unternehmen, die schnell auf neue Kundenbedürfnisse und sich verändernde Marktbedingungen reagieren müssen, eine kontinuierliche Bereitstellung softwaregestützter innovativer Lösungen. In diesem Buch erfahren Sie mehr über DevOps und darüber, wie Ihr Unternehmen von diesem Ansatz profitieren kann. Lernen Sie DevOps aus einer ganzheitlichen Perspektive kennen, die den gesamten Softwareentwicklungsprozess umfasst - von der ersten Idee, über die Konzeption neuer Ressourcen bis hin zur Implementierung in der Produktion - und profitieren Sie vom Wettbewerbsvorteil einer kontinuierlichen Softwarebereitstellung.

- ***Übertreffen Sie die Erwartungen Ihrer Kunden - liefern Sie qualitativ noch hochwertigere Ergebnisse***
- ***Tragen Sie den Bedürfnissen Ihres Marktumfeldes Rechnung - reagieren Sie auf die großen Steigerungen bei der Häufigkeit und Umfang von Softwarebereitstellung***
- ***Profitieren Sie von neuen Technologietrends - setzen Sie auf mobile Anwendungen, Cloud, Big Data und Social Media***
- ***Etablieren Sie eine bessere Kooperation - beteiligen Sie alle Akteure im Softwareentwicklungszyklus***

Inhalt dieses Buchs:

- **Die Einsatzmöglichkeiten von DevOps**
- **DevOps und mobile Anwendungen, Cloud- und weitere führende Technologien**
- **DevOps und Application Lifecycle Management (ALM)-Prozesse**
- **Verwaltung von Mehrschichtenanwendungen**
- **IBM's Sicht von DevOps**

WILEY

ISBN : 978-1-119-17745-6
Artikel-Nr. : RAM14025-DEDE-00

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley’s ebook EULA.