Die 5 grundlegenden **DevOps-Praktiken**

Wie man sie etabliert und darauf aufbaut

splunk>
turn data into doing

Inhalt

Einleitung 3

Auf eigenen Wegen zum DevOps-Erfolg 4

Die grundlegenden Praktiken und CAMS 5

Das Prinzip gemeinsamer Nutzung 6

Die fünf grundlegenden Praktiken im Einzelnen 7

Das Team, das den Service betreibt, kann auch Monitoring und Benachrichtigungen konfigurieren 8

Die Bereitstellungsmuster zur Erstellung von Apps oder Services werden wiederverwendet 14

Die Testmuster zur Erstellung von Apps oder Services werden wiederverwendet 16

Die Teams tragen zur Verbesserung der Tools bei, die andere Teams bereitstellen 17

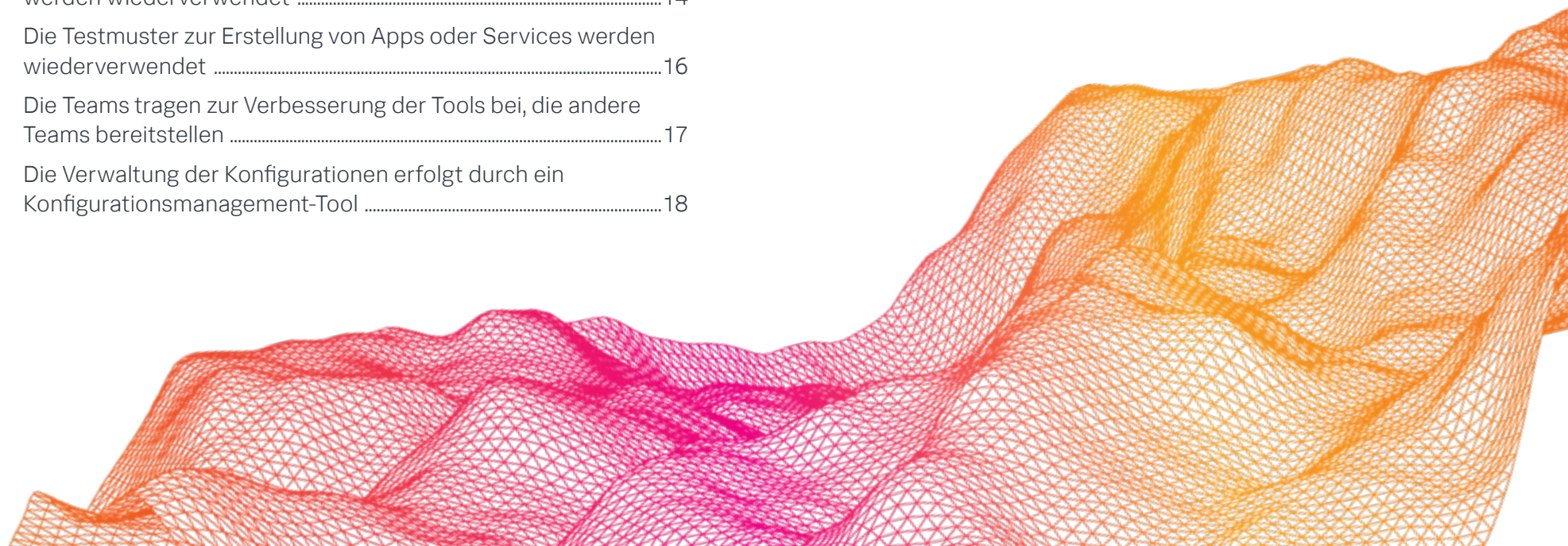
Die Verwaltung der Konfigurationen erfolgt durch ein Konfigurationsmanagement-Tool 18

Die grundlegenden Praktiken im eigenen Unternehmen verankern 20

Am Ende fangen Sie dort an, wo Sie sind 21

Der beste Weg ist der Weg, der schnelle Ergebnisse bringt 22

Über den Autor 23



Einleitung

Anfangs kannten es die meisten nur als Insider-Buzzword. Ein knappes Jahrzehnt später ist DevOps allgemein akzeptierte Praxis geworden. Unternehmen bewältigen damit den technologischen Wandel und stellen Software selbst in schnell drehenden, wettbewerbsintensiven Geschäftsumfeldern bereit. Es gibt heute zahlreiche Musterbeispiele, die auf den Tech-Konferenzen weltweit vorgestellt werden, in der Fachliteratur und in den Blogs der Branche. Und dennoch sind die meisten Teams immer noch unsicher, wo sie anfangen sollen. Und wie es nach den ersten Schritten weitergeht.

Wir haben uns angesehen, wie Unternehmen auf ihrer DevOps-Reise Fortschritte machen. Die Analyse der Daten hat gezeigt, das erfolgreiche Unternehmen ganz bestimmte Phasen durchlaufen. Bei dieser Untersuchung haben sich auch entscheidende Verfahren herauskristallisiert – eben die fünf grundlegenden Praktiken –, die über den Erfolg der DevOps-Evolution entscheiden.

Auf eigenen Wegen zum DevOps-Erfolg

Jedes Unternehmen hat seinen eigenen Ausgangspunkt. Jedes Business hat bestimmte Traditionstechnologien im Einsatz, folgt eigenen Verfahrensweisen, hat seine spezielle Mission und seine eigene Kultur. Es gibt also nicht den einen Weg der DevOps-Transformation, sondern eine Vielzahl möglicher Entwicklungspfade.

Eine Analyse der Daten aus dem Splunk Lagebericht Observability hat aufgezeigt, welche grundlegenden Praktiken erfolgreiche Teams auszeichnen. Diese Praktiken korrelieren so eng mit dem DevOps-Erfolg, dass sie unserer Beobachtung nach in jeder Phase der DevOps-Evolution unerlässlich sind. Mit anderen Worten: Die Praktiken, die man in einer bestimmten Phase etablieren muss, um die nächste Phase zu erreichen, bleiben auch dann noch wichtig, wenn ein Unternehmen auf seiner DevOps-Reise schon weit fortgeschritten ist und bereits beispielhaften Erfolg vorweisen kann.

Jede dieser grundlegenden Praktiken lässt sich in einem Satz beschreiben:

- Das Team, das den Service betreibt, kann auch **Monitoring und Benachrichtigungen** konfigurieren.
- Die **Bereitstellungsmuster** zur Erstellung von Apps oder Services werden wiederverwendet.
- Die **Testmuster** zur Erstellung von Apps oder Services werden wiederverwendet.
- Die Teams **tragen zur Verbesserung der Tools bei,** die andere Teams bereitstellen.
- Die **Verwaltung der Konfigurationen** erfolgt durch ein Konfigurationsmanagement-Tool.

Als wir diese fünf Punkte einzeln unter die Lupe nahmen, stellten wir fest, dass Unternehmen mit einem hohen Reifegrad diese Praktiken tendenziell im gesamten Verlauf ihrer DevOps-Entwicklung anwenden als weniger entwickelte Unternehmen. Die Lehre aus diesen Beobachtungen ist: Die genannten grundlegenden Praktiken sind integrale DevOps-Komponenten und K.o.-Kriterien des DevOps-Erfolgs.

Die grundlegenden Praktiken und CAMS

Wer sich eingehender mit DevOps beschäftigt, wird kaum überrascht sein, dass gerade diese grundlegenden Praktiken so wichtig sind. Andere bewährte Modellierungen beruhen praktisch auf denselben Grundlagen, namentlich das CAMS-Modell[1], eine der frühesten DevOps-Beschreibungen. Es enthält de facto dieselben bewährten Grundregeln: durchgängig messen, gemeinsam nutzen, intelligent automatisieren etc. Auch andere Begriffe und Paradigmen, auf die man im DevOps-Diskurs immer wieder stößt (etwa der „Shift-Left-Ansatz“, der bestimmte Aufgaben zu einem früheren Zeitpunkt der Entwicklung einbindet, die „Empowered Teams“ mit Eigenverantwortung oder „Test-driven Development“), bestätigen die Relevanz dieser grundlegenden Muster und Praktiken.

Kultur

Automatisierung

Messen

Teilen

1 CAMS ist das Kürzel für „Culture, Automation, Measurement, Sharing “(Kultur, Automatisierung, Messen, Teilen). CAMS wurde 2010 von Damon Edwards und John Willis erstmals definiert. Eine eingehende Erörterung von CAMS und DevOps unternimmt das Kapitel „CAMS and the DevOps evolutionary model“ im State of DevOps Report 2018 von Puppet.

Das Prinzip gemeinsamer Nutzung

Bei der Untersuchung der grundlegenden Praktiken wird deutlich, dass sie alle von der gemeinsamen Nutzung abhängen und dass sie alle die gemeinsame Nutzung fördern. Wenn wir uns nun überlegen, was in der DevOps-Praxis gemeinsam genutzt wird, dann beginnen wir am besten mit der Konfiguration sowie mit Monitoring und Benachrichtigungen.

Monitoring und Benachrichtigungen sind

- konfigurierbar durch das Team, das den Service betreibt,
- wesentlich für den Austausch von Informationen darüber, wie Systeme und Anwendungen laufen, und
- so angelegt, dass alle Beteiligten sich einig darüber sind, wie die Systeme laufen sollten.

Gerade dieser Konsens ist die entscheidende Voraussetzung für Verbesserungen, sowohl innerhalb der einzelnen Teams und Funktionen als auch team- und funktionsübergreifend. Gemeinsame Nutzung ist aber ebenso das Prinzip der folgenden Praktiken:

- Die Bereitstellungsmuster und Testmuster zur Entwicklung von Anwendungen oder Services werden wiederverwendet: Die gemeinsame Nutzung bewährter Muster in verschiedenen Anwendungen oder Diensten bedeutet oft auch, dass verschiedene Teams diese Muster teilen. So entstehen einvernehmliche Arbeitsweisen, die wiederum die Basis weiterer Verbesserungen bilden.
- Die Teams tragen zur Verbesserung der Tools bei, die andere Teams bereitstellen: Diese Form des Austauschs fördert die teamübergreifende Diskussion von Prioritäten und Plänen für weitere Verbesserungen bei Tools, Prozessen und Messungen.
- Die Verwaltung der Konfigurationen erfolgt durch ein Konfigurationsmanagement-Tool: Mit einem solchen Tool können auch Entwicklung, Security und weitere Teams Änderungen an den System- und Anwendungskonfigurationen vornehmen – nicht nur das Operations-Team. Einsatzfähigkeit und Sicherheit werden damit im Unternehmen zu einer gemeinsam geteilten Verantwortung.

Die Entdeckung, dass alle grundlegenden Praktiken auf gemeinsamer Nutzung beruhen oder darauf ausgerichtet sind, zeigt uns, worin der Schlüssel zu skalierbarem DevOps-Erfolg besteht: Es geht darum, Praktiken zu etablieren, die das Teilen und die gemeinsame Nutzung fördern.

Das leuchtet auch ein. Wer sieht, dass etwas gut läuft, will diesen Erfolg wiederholen. Und natürlich lassen wir andere an unseren Erfolgen teilhaben. Angenommen, Ihr Team hat eine Anwendung zehnmal erfolgreich bereitgestellt, und nehmen wir außerdem an, dass diese Art der Bereitstellung Ihrem Team (und anderen) den sonst üblichen Ärger erspart. Dann stehen die Chancen gut, dass es jemandem auffällt, der wissen will, wie Sie das anstellen. Auf diese Weise wachsen DevOps-Praktiken nach und nach von einem Team ins nächste.

Die fünf grundlegenden Praktiken im Einzelnen

In diesem Abschnitt beschreiben wir die grundlegenden Praktiken nacheinander im Detail und erläutern, was deren Umsetzung zur DevOps-Entwicklung beiträgt.

01

Das Team, das den Service betreibt, kann auch Monitoring und Benachrichtigungen konfigurieren

Das Wesen der DevOps-Bewegung zeigt sich am besten als zwei Seiten einer Medaille: Empowerment (Befähigung) und Accountability (Verantwortlichkeit). Amazon-CTO Werner Vogels hat das in seinem berühmten Diktum so zusammenfasst: „You build it, you run it."[2] Also etwa: „Was Sie bauen, das betreiben Sie auch." Wir haben daher untersucht, wie viele der Teams, die Anwendungen und Services in der Produktion betreiben – und zwar egal, ob sie zu Entwicklung, Operations oder Software-Release zu zählen sind –, in der Lage sind, ihre eigenen Kriterien für Monitoring und Benachrichtigungen zu definieren.

Befähigte Teams (mit Empowerment), die Anwendungen und Services in der Produktion betreiben, können selbst definieren, was als guter Service gilt, und selbst bestimmen, wie sie feststellen, ob er ordnungsgemäß funktioniert, und wie sie herausfinden, wann er nicht funktioniert. Dieser eigenverantwortliche Monitoring-Ansatz kann ganz unterschiedlich aussehen. Zum Beispiel so:

- „Lege deine Monitoring-Konfiguration hier ab, wir greifen sie dann auf."
- „Melde dich bei dieser Web-Oberfläche an und konfiguriere dort dein Monitoring selbst."
- „Ergänze deinen Infrastrukturcode um Ausgaben, die sich abfragen lassen."
- „Hier hast du eine API, mit der du dein Monitoring per Code konfigurieren kannst."

Wir haben festgestellt, dass Unternehmen in der Lage sind, eigene Monitoring- und Benachrichtigungskriterien für Apps und Services in der Produktion zu definieren, sobald sie merken, dass DevOps wirkt. Die Teams bestimmen dann selbst, wann ein Service gesund ist, und sie legen selbst die Kriterien fest, anhand derer sie herausfinden, ob er ordnungsgemäß funktioniert oder nicht. Darüber hinaus nutzen sie die Möglichkeiten der Automatisierung, sodass sie effizienter arbeiten und sich die Plackerei wiederkehrender Aufgaben von geringem Wert sparen.

2 Jim Gray: A Conversation with Werner Vogels. In: ACM Queue 4 (2006).

Wenn die Teams dazu ermächtigt werden, ihre eigenen Mess- und Warnfunktionen zu definieren, zu verwalten und zu teilen, wird damit die DevOps-Transformation in mehrfacher Hinsicht unterstützt. Von Bedeutung sind u. a. diese Aspekte:

- Austausch von Metriken als Ansporn zu kontinuierlicher Verbesserung.
- Schaffung und Förderung einer Kultur kontinuierlichen Lernens.
- Teamübergreifende Zusammenarbeit und verantwortliche Teams.
- Entwicklung von systemischem Denken, sowohl bei den Einzelnen als auch in den Teams.

Wie gesagt sind die Aspekte gemeinsamer Nutzung von zentraler Bedeutung für eine starke DevOps-Kultur. Insofern ist es kaum verwunderlich, dass in unserer Studie diejenigen Unternehmen, die einen hohen DevOps-Reifegrad aufweisen, solche Praktiken bereits frühzeitig etabliert haben.

Was Sie messen sollten – und wie

Unsere Untersuchungen zeigen, worauf sich Monitoring und Benachrichtigung richten sollten:

- **Wichtige Systemmetriken** wie Latenz, Antwortzeit, Ressourcenauslastung etc. Diese Zahlen erfasst die Mehrheit der Befragten (56 %).
- **Geschäftsziele,** die sich aus Systemmetriken ableiten lassen. Ein Beispiel wäre die Messung der Reaktionszeit als eines Indikators der Kundenzufriedenheit. Bei mehr als einem Drittel der Befragten (37 %) ist das Monitoring darauf ausgelegt.
- **Ad-hoc-Zugriff auf genuine Geschäftsmetriken:** Verweildauer auf der Website, geöffnete Anwendungen, Neukundenanmeldungen, Erlösraten usw. Diese Fähigkeit haben 20 % der Befragten.

Wer diese Daten erfasst, hat damit die besten Möglichkeiten der Analyse und kann in modernen, komplexen Systemen am ehesten herausfinden, wo ein Problem aufgetreten ist. Wenn alle diese Daten zur Verfügung stehen, lässt sich sehr viel einfacher feststellen, wo ein Fehler vorliegt und wie er zu beheben ist.

Warum das Monitoring genuiner Geschäftsmetriken wichtig ist

Unser Lagebericht Observability hat aufgezeigt, welche Konsequenzen unerkannte Probleme haben können: Bei 45 % der Befragten leidet die Kundenzufriedenheit, 37 % klagen über Umsatzeinbußen aufgrund von Serviceausfällen. Wenn die Geschäftsmetriken gemessen werden, sind solche Probleme leichter zu erkennen, bevor sie sich nachteilig auf Ihr Business auswirken.

Manche IT-Teams verwenden Geschäftsmetriken als die primären Messgrößen der IT-Leistung.

Geschäftsmetriken können Servicefragen beantworten.

Und zwar ganz alltägliche, die sich jeden Tag neu stellen, z. B. diese:

- Kann unsere Kundschaft bei uns kaufen?
- Wie viele Menschen bedienen wir?
- Wie viel Geld verdienen wir?
- Ist dieses Muster normal?

Geschäftsmetriken sind auch für die langfristige Erfolgsmessung wichtig.

Und zwar mit Antworten auf Fragen wie diese:

- Leistet die neue Version das, was wir uns von ihr erwarten?
- Trägt sie zu besseren Geschäftsergebnissen bei?
- Wie wirkt sie sich im Vergleich zu anderen Versionen auf den Geschäftsgang aus?

Für fortschrittliche Unternehmen, die Datenanalysen, maschinelles Lernen und künstliche Intelligenz einsetzen, ist die Messung der Geschäftsmetriken eine Art Frühwarnsystem, das auf eine niedrigere Umsatzrate pro Minute oder eine überdurchschnittlich hohe Absprungrate hinweisen kann. Dergleichen ist unter Umständen auf unentdeckte Fehler auf der Website zurückzuführen.

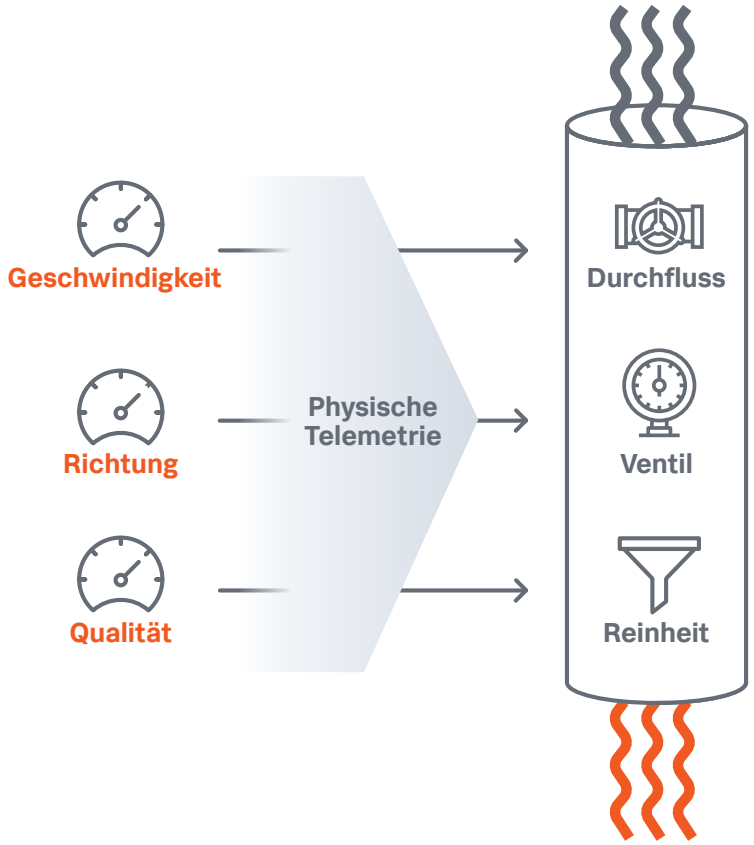
Nicht zuletzt kann das Monitoring von Geschäftsmetriken einem Team auch den Weg zu weiteren Ressourcen für DevOps frei machen, denn damit lassen sich die bisher erzielten Erfolge in Begriffen darlegen, die das Business-Ende des Unternehmens versteht.

Observability, Telemetrie und strukturierte Protokollierung

In den meisten Unternehmen mit hohem Reifegrad werden zu einer Anwendung bzw. einem Service gleich Metriken, Traces und Logs bereitgestellt, was Monitoring und Benachrichtigungen im späteren Produktionsbetrieb deutlich einfacher macht. Auf diese Weise schafft man Observability – und ist damit in der Lage, Antworten auf nahezu beliebige geschäftliche Fragen geben, auch auf solche, von denen im Vorfeld niemand geahnt hätte, dass sie sich stellen würden.

Der Begriff „Observability“ stammt ursprünglich aus der Welt der industriellen Systeme. Manchmal ist Observability einfach dadurch zu erreichen, dass man am Wasserrohr einen Durchflusssensor montiert, die innen gemessenen Werte an eine äußere Anzeige überträgt und so im Prinzip Telemetriedaten erhält. Das Bedienpersonal kann dann eine interne Eigenschaft des Systems (wie schnell das Wasser im Rohr fließt) anhand der externen Ausgabe (am Durchflussmesser) beobachten.

Observability in industriellen Systemen

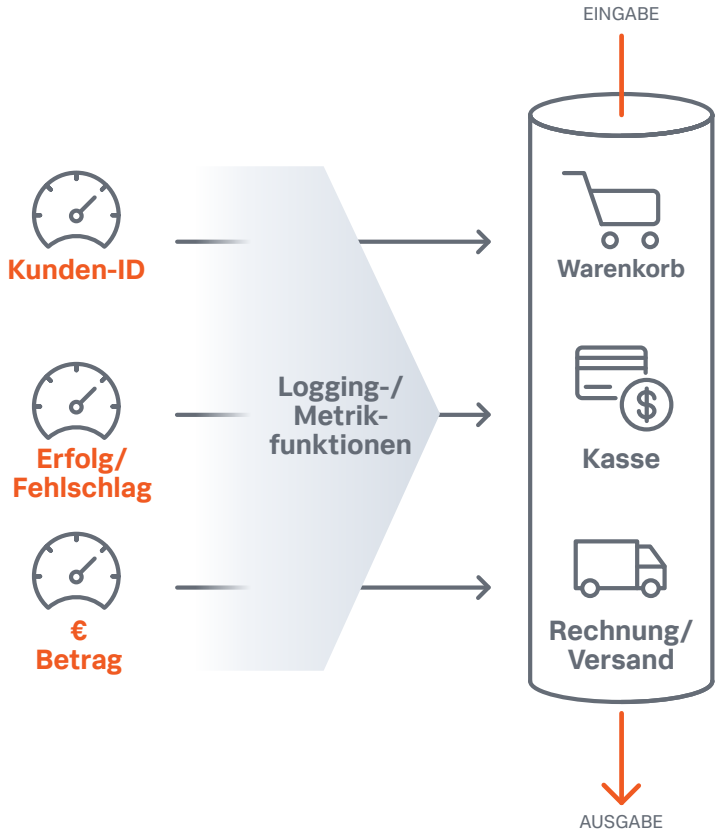


Bei Anwendungen der IT wird Observability oft mit Performance-Metriken, Microservice-Aufrufverfolgung und strukturierter/semantischer Protokollierung erzielt, sodass sich damit die tatsächliche Aktivität der Anwendung beschreiben lässt. Das können sowohl technische Metriken wie die End-to-End-Antwortzeit als auch Geschäftsmetriken wie der Erlös pro Transaktion sein. Auf diese Weise können die Teams ihr Monitoring anhand von echten Messwerten des technischen und geschäftlichen Erfolgs konfigurieren und müssen sich nicht mit Schätzungen und Vermutungen begnügen.

Dank Observability können IT-Teams mit modernen zusammengesetzten Architekturen aus Clouds, Microservices, Containern, APIs und serverlosen Infrastrukturen arbeiten. Denn trotz all ihrer Vorteile bieten diese Architekturen unter Umständen nur wenig Einblick in die Leistung der Infrastruktur, keine Möglichkeiten eigener Instrumentierung (und nur minimale native Instrumentierung) und der Ausführung synthetischer Transaktionen.

In solchen komplexen und undurchsichtigen Systemen braucht es Metriken, Traces und Logs, die von den DevOps-Teams bereits beim Release mit ausgespielt werden. Dann bekommen die Teams volle Kontrolle über ihre Systeme und sind in der Lage, künftig für jedes Szenario ein sinnvolles Monitoring zu definieren und zu konfigurieren.

Observability in Softwaresystemen



Messungen sind ein DevOps-Wesensmerkmal

Die Befähigung zum Monitoring ist keineswegs nur etwas für Operations- oder für neu geschaffene DevOps-Teams, sondern für alle Teams, die mit Technologie arbeiten. Wer den Teams die Möglichkeit zum Monitoring gibt, hat einen der wichtigsten DevOps-Gedanken erfasst: dass man kein neues Team mit neuen Superkräften schaffen muss, sondern stattdessen alle bestehenden Teams befähigt, damit sie auf eine neue Art und Weise zusammen arbeiten können.

Monitoring und Benachrichtigungen nach dem Selfservice-Prinzip sind ebenso einfach und sinnvoll für

- einzelne Entwicklerinnen und Entwickler, die ihren eigenen Code überwachen, der in der Produktion läuft,
- Entwicklungsteams, die mit den entsprechenden Operations-Teams zusammenarbeiten, um funktionsfähige Anwendungen zu entwickeln,
- DevOps-Teams, die als Einheit arbeiten und ihre eigene Monitoring-Praxis definieren, oder
- komplexe Meta-Teams, bei denen Operations-Fachleute im Rahmen eines umfassenderen Systems die von der Entwicklung bereitgestellten Anwendungen überwachen.

Monitoring und Benachrichtigungen, die sich selbstbestimmt einrichten lassen, sind unabhängig von den besonderen Umständen ein wirksames Mittel gegen die überkommene Praxis, dass Entwicklung und Operations in ihren Silos vor sich hinarbeiten. Die einfache Freigabe des Zugriffs auf diese wichtigen Metriken ermöglicht eine Kultur des Austauschs, schafft lebendige Feedback-Schleifen und fördert teamübergreifend kontinuierliches Lernen.

02

Die Bereitstellungsmuster zur Erstellung von Apps oder Services werden wiederverwendet

Dies ist die zweite grundlegende Fähigkeit, die sich in unseren Untersuchungen als entscheidend für den DevOps-Erfolg gezeigt hat: die Fähigkeit zur Wiederverwendung von einmal definierten Bereitstellungsroutinen, von Prozessen, Systemen und Tools zur Erstellung von Anwendungen oder kompletten End-to-End-Services.

Die Wiederverwendung von Code ist für eine qualitativ anspruchsvolle Softwareentwicklung unerlässlich. Dasselbe gilt auch für den Code von Bereitstellung und Monitoring. Durch Monitoring as Code und Infrastructure as Code verkürzen Sie Ihre Release-Zyklen, senken die Fehlerquote und eröffnen sowohl Entwicklung als auch Operations neue Wege, ihr Bestes zu geben.

Eine jüngere Puppet-Umfrage zum Thema wollte wissen, wie oft die grundlegenden Praktiken in den Unternehmen zum Einsatz kommen, wenn sich erste DevOps-Erfolge gezeigt haben. Hier die Ergebnisse zu „Die Bereitstellungsmuster zur Erstellung von Apps oder Services werden wiederverwendet“:

	NIEDRIG	MITTEL	HOCH
IMMER	2 %	14 %	46 %
MEISTENS	7 %	44 %	47 %
MANCHMAL	34%	33 %	6 %
SELTEN	40 %	8 %	1 %
NIE	18 %	1 %	—

Die Anschlussfrage: „Wie häufig kamen die folgenden Praktiken beim Stand erweiterter DevOps Verfahren zum Zuge?“ Auch hier wieder die Auswertung für die Wiederverwendung von Bereitstellungsmustern:

	NIEDRIG	MITTEL	HOCH
IMMER	—	11 %	52 %
MEISTENS	7 %	45 %	47 %
MANCHMAL	30 %	35 %	1 %
SELTEN	45 %	8 %	—
NIE	18 %	1 %	—

> Wir fragten aus einem bestimmten Grund nach der Wiederverwendung von Bereitstellungsmustern: Die Anwendungsbereitstellung spielt nämlich in den meisten, wenn nicht sogar in allen Unternehmen eine besondere Rolle. Sie markiert den Übergang von der Entwicklung in die Produktion und ist die „Wall of Confusion“[3], an der sich Dev und Ops am häufigsten begegnen – und wo es am meisten schmerzt, wenn sie zusammenstoßen. Die Optimierung der Anwendungsbereitstellung ist daher ein wesentlicher DevOps-Bestandteil, denn sie durchbricht diese „Mauer der Missverständnisse“ zwischen Entwicklung und Betrieb.

Der Einsatz wiederholbarer Muster – ganz gleich, ob sie intern erstellt oder von einer externen Quelle übernommen werden – behebt nicht nur die direkte Not und Verwirrung der Bereitstellung. Auf diese Weise wird es außerdem möglich, das Wissen um die Bereitstellung im Unternehmen zu teilen und weiterzugeben, sodass künftig mehr Teams und Einzelne ihren Beitrag dazu leisten können. Schließlich sollte die Anwendungsbereitstellung in jedem Unternehmen eine Kernkompetenz sein.

3 Damon Edwards: What is DevOps? In: dev2ops, 23. 02. 2010.

03

Die Testmuster zur Erstellung von Apps oder Services werden wiederverwendet

Dass das Testen der Software automatisiert wird, ist mindestens so wichtig wie die Automatisierung von Bereitstellung und Monitoring – wenn nicht noch wichtiger. Denn schon kleine Änderungen können unvorhergesehene Auswirkungen auf andere Services haben. Und ohne ein solides Testsystem bleiben Ihnen diese Folgen unter Umständen verborgen. Hier einige Überlegungen zur Prioritätensetzung:

- Wenn es eine separate Qualitätskontrolle gibt, dann sollten Sie dieses Team erst zu einem späteren Zeitpunkt auf die DevOps-Reise mitnehmen. Konzentrieren Sie sich zunächst darauf, **innerhalb des eigenen Teams gute Testmuster** zu etablieren. Operations-Teams könnten z. B. einen Prozess zum Testen von Infrastrukturänderungen finden, bevor diese in die Produktion gelangen. Entwicklungsteams könnten TDD (Test-driven Development) oder eine andere Methodik aufgreifen und in ihrem agilen Workflow anwenden.
- Was dicht an der Produktion ist (Provisioning, Monitoring, Benachrichtigungen etc.), hat für die Teams oft höhere Priorität, weil dort die Probleme eher deutlich werden. Lösen Sie **zuerst die Aufgaben der Bereitstellung** – so gewinnen Sie Zeit, die Sie dann zur Optimierung Ihrer Testverfahren nutzen können.
- Testmuster sind manchmal **nicht so gut wiederverwendbar** wie Bereitstellungsmuster, weil das Testen die Besonderheiten von Apps bzw. Services betrifft und viele verschiedene Prozesse umfasst (Smoke-Tests, Modultests, Funktions-, Konformitäts- und Komplexitätstests), und das sowohl in statischen (Whitebox) als auch in dynamischen (Blackbox) Umgebungen.
- Das Testen in der Produktion ist **schwieriger und oft komplexer** als das Testen in der Vorproduktion, da die Ziele dort andere sind. Mit Continuous Delivery können die Teams neue Ideen in der Produktion austesten (z. B. als Blue-Green- oder Canary Releases). Das ist zwar hilfreich, aber wiederum nicht dasselbe wie Tests in der Produktion, wo es auf Customer Experience, Qualität, Funktionalität, Resilienz, Stabilität usw. ankommt.
- Wiederverwendbare Testmuster gehören zwar zu den grundlegenden Praktiken, kommen in der Regel aber erst in späteren Phasen der DevOps-Reise zum Zuge, nachdem die Bereitstellungsmuster etabliert sind. Sind wiederverwendbare Testmuster aber erst einmal eingeführt, dann sollten Sie sicherstellen, dass sie **auch gemeinsam genutzt werden** können, z. B. indem Sie die Muster in automatisierten Testtools codieren und den Zugriff – samt Reports und Dashboards – für alle Beteiligten freigeben.

04

Die Teams tragen zur Verbesserung der Tools bei, die andere Teams bereitstellen

Dass Einzelne und Teams zur Optimierung der Tools beitragen können, die andere Teams zur Verfügung stellen, ist ebenfalls eine wichtige grundlegende Fähigkeit.

Verbesserungen am Toolset geschehen in der Regel von Hand, finden mehr oder weniger spontan statt und bleiben zunächst auf das jeweilige Team beschränkt, bis irgendeine Veränderung die Ausdehnung auf andere Teams erzwingt. Das können abteilungsweite kulturelle oder organisatorische Veränderungen sein, neue Datenquellen jenseits der bisherigen Grenzen wie bei Semantic Logging oder neue Formen der teamübergreifenden Zusammenarbeit über Funktionsbereiche wie Bereitstellung oder Release-Automatisierung hinweg.

Letztlich setzen teamübergreifende Beiträge zu den verwendeten Tools aber voraus, dass die Teams zuerst bei sich selbst aufgeräumt haben. Darum sind wir der Ansicht, dass man diese grundlegende Praxis bei identischen Erfolgsaussichten auf eine spätere Phase der DevOps-Reise verschieben kann.

05

Die Verwaltung der Konfigurationen erfolgt durch ein Konfigurationsmanagement-Tool

Konfigurationsmanagement-Tools setzen sich erfahrungsgemäß schnell durch, sobald ein Unternehmen erste Erfolge bei der DevOps-Entwicklung verzeichnen kann.

Für altgediente DevOps-Verfechter ist es keine Überraschung, dass sich diese Praxis als wesentlicher Erfolgsfaktor herauskristallisiert hat. Automatisiertes Konfigurationsmanagement war etliche Jahre lang der Haupttreiber von DevOps, vor allem in den Anfangstagen, als man begann, mit Infrastructure as Code zu arbeiten und sich die DevOps-Bewegung vornehmlich um die frühen Innovatoren des automatisierten Konfigurationsmanagements scharte. In letzter Zeit haben vor allem die Migration in die Cloud und die Komplexitäten der dortigen Bereitstellung deutlich gemacht, wie wichtig Managementsysteme für Konfiguration und Bereitstellung sind.

Sehen wir uns die DevOps-Treiber der Anfangstage noch einmal an; so lässt sich am besten darstellen, warum das Konfigurationsmanagement von so großer Bedeutung ist.

> Mit der Consumerization der IT kam es zu einem beispiellosen Bedarf an Geschwindigkeit und Services. Dass die Dev-Teams nicht gut in der Lage waren, wichtige Ressourcen für Entwicklung und Tests bereitzustellen und zu konfigurieren – ganz zu schweigen von der Produktion –, erwies sich als Hindernis auf dem Weg zu einer raschen Bereitstellung qualitativ hochwertiger Software, die den Kunden- und Geschäftsanforderungen entspricht.

Die IT-Operations-Teams waren zur gleichen Zeit ebenso frustriert, weil es ihnen unmöglich war, wichtige Änderungen an den von ihnen ausgeführten Anwendungen und Services vorzunehmen – Änderungen, die diese Apps und Dienste stabiler, effizienter und handlicher gemacht hätten.

Mit dem Aufkommen von Cloud Computing bekamen die Entwicklungsteams eine Möglichkeit, rasch den eigenen Bedarf zu decken – mehr als eine Kreditkarte war dazu nicht mehr nötig. Das brachte wiederum Probleme und neuen Ärger für die Ops-Teams mit sich. Wenn die Dev-Teams Schatten-IT anschafften, dann standen die Operations-Teams vor einer Unmenge von sonderkonfigurierten, schwer zu verwaltenden, äußerst brüchigen und fehleranfälligen Software-Sensibelchen in der Produktion.

Als Dev und Ops dann anfingen, gemeinsam nach einer Lösung dieser Bereitstellungsprobleme zu suchen, war es nur natürlich, dass viele DevOps-Projekte als Erstes bei der Automatisierung von Konfiguration und Bereitstellung ansetzten. Damit hatten sie eine Lösung, bei der beide, die Entwicklung und der Betrieb, Gewinner waren: Die Dev-Teams konnten endlich mit dem Tempo der Märkte mithalten, und durch die gekapselte Kodifizierung bewährter Konfigurationen hatten die Ops-Teams weniger Aufwand.

Je mehr DevOps sich weiterentwickelt und wächst, desto deutlicher bewegen sich die durch das automatisierte Provisioning befreiten Dev-Teams nun in Richtung kontinuierlicher Bereitstellung. Die Operations-Teams, die Uptime, Performance und Verfügbarkeit in der Produktion gewährleisten müssen, stehen dadurch wieder unter Druck. Mit zunehmendem Reifegrad der Unternehmensprozesse tauchen auch Fragen nach der Auditierbarkeit auf. So sind Automationslösungen für Konfiguration und Provisioning in der Produktion letztlich genauso wichtig geworden wie das Provisioning für Entwicklung und Tests. Und erst wenn diese Abläufe durch ein Konfigurationsmanagement wiederholbar gemacht werden, sind stabile, zuverlässige und auditierbare Produktionsumgebungen gewährleistet. Damit ist dann auch der Ausblick auf weitere Fähigkeiten eröffnet – Stichwort: Selfservice –, die sich als nächste Stationen auf der DevOps-Reise abzeichnen.

Am Ende geht es bei der Automatisierung von Provisioning und Konfiguration aber nicht nur um Geschwindigkeit. Es geht auch darum, der Furcht zu begegnen, die Führungsetagen beim Stichwort DevOps immer als Erstes befällt: dass ein Entwickler mit all seinem Empowerment in der Produktion alles Mögliche anstellen könnte, ohne Absprache, ohne Prüfung, ohne Kontrolle. Wenn aber bewährte Prozesse als Code fixiert sind und voreingestellte Routinen die Ausführung übernehmen, dann zerstreut das diese Befürchtungen – und die Dev- und Ops-Teams können dennoch agiler arbeiten.

Vor allem für größere Unternehmen und solche aus streng regulierten Branchen wie dem Gesundheits- oder dem Finanzwesen sowie für Behörden bietet die Automatisierung den zusätzlichen Vorteil, dass aufgezeichnet wird, wer wann und wo im System war. (Und oft sogar: warum.)

Die grundlegenden Praktiken im eigenen Unternehmen verankern

Es gibt genügend Belege dafür, dass die fünf grundlegenden Praktiken zum DevOps-Erfolg führen. Wir vermuten aber, dass Sie auch eine Anleitung zur Umsetzung erwarten. Zum Glück geben unsere Untersuchungen einige Hinweise darauf, mit welchen Praktiken Sie beginnen sollten.

Im Allgemeinen, das legen die Forschungsergebnisse nahe, sollten Unternehmen mit den Grundlagen starten, die einerseits das Fundament für Wachstum und weitere Entwicklung in der Zukunft legen, andererseits unmittelbaren Nutzen bringen. Mit welchen Praktiken Sie konkret beginnen, hängt stark davon ab, wo Ihr Unternehmen zu Beginn der Reise steht und wo der Schuh am heftigsten drückt. Aber bitte denken Sie daran: Jede der fünf grundlegenden Praktiken wird bei fortschrittlichen Unternehmen bereits zu einem frühen Zeitpunkt umgesetzt.

Drei Praktiken sind bei der Mehrheit der fortschrittlichen Unternehmen immer aktiv im Einsatz, sobald sich erste DevOps-Erfolge zeigen:

- wiederverwendbare Bereitstellungsmuster,
- Konfigurationsmanagement-Tools und
- Teams, die für die Services, die sie betreiben, Monitoring und Benachrichtigungen selbst konfigurieren.

Diese drei Praktiken stellen zwar grundlegende Fähigkeiten dar und bilden die Basis für den Übergang zu höheren DevOps-Entwicklungsstufen, aber in welcher Reihenfolge sie eingeführt werden, ist unwichtig. Fangen Sie hier an, aber zerbrechen Sie sich nicht den Kopf darüber, was zuerst kommt. Vermutlich werden Sie rasch merken, dass Ihr Unternehmen eine der Praktiken dringender als die anderen braucht.

Unsere Untersuchungen legen eindeutig nahe, dass Unternehmen zunächst möglichst die Grundlagen gründlich etablieren müssen, bevor sich erste Ergebnisse zeigen. Arbeiten Sie also an der Umsetzung der drei oben genannten Praktiken, und weiten Sie sie dann so bald wie möglich auf andere Teams aus.

Die beiden anderen grundlegenden Praktiken sollten Sie sich vornehmen, nachdem die ersten drei gut etabliert sind:

- wiederverwendbare Testmuster für Apps und Services sowie
- Teams, die zur Verbesserung der Tools anderer Teams beitragen.

Am Ende fangen Sie dort an, wo Sie sind

Wie gesagt: Jedes Unternehmen ist anders. So oder so müssen Sie dort beginnen, wo Sie jetzt stehen – die Betonung liegt auf „jetzt“. Eruieren Sie also die derzeitigen Bedingungen im Unternehmen und wählen Sie den Ansatz, der am ehesten dort positive Wirkung zeigt, wo sie am dringendsten nötig ist. Mit den grundlegenden Praktiken, die besseres Monitoring, bessere Konfigurationskontrolle, bessere Bereitstellungsmuster, bessere Qualitätskontrollen und bessere Zusammenarbeit bei den Tools bieten, sollten Sie ausreichend Auswahl haben.

Legen Sie von Anfang an die grundlegenden Messwerte für System-, Kunden- und Geschäftsmetriken fest und machen Sie diese Zahlen bekannt, damit alle wissen, von wo aus sie starten, was verbessert werden muss und welche Auswirkungen die eigene Arbeit im gesamten End-to-End-System hat. Sie machen es damit allen Beteiligten leichter, in DevOps-Konzepte wie kontinuierliches Feedback und kontinuierliche Optimierung hineinzufinden.

Gewöhnen Sie sich frühzeitig an, Arbeitsabläufe und Prozesse zu teilen und weiterzugeben, damit sie gemeinsam als Team vorankommen. Automatisieren Sie und starten Sie die Shift-Left-Bewegung mit Automatisierungen für Konfiguration und Provisioning, sodass die Dev-Teams agiler arbeiten können und die Ops-Teams eine verlässliche, vorhersehbare Grundlage bekommen. Dann werden die Mitglieder beider Teams sehr viel eher verstehen, welchen Wert dieses neue Wagnis hat. Wenn Sie den Leuten das Arbeitsleben leichter machen, ist das der beste Weg, ihr Vertrauen zu gewinnen und ihre Mitarbeit in der Zukunft zu sichern.

> Shift-Left bedeutet grundsätzlich, in der Abfolge von links nach rechts etwas „weiter links“, also früher zu erledigen als bisher. Durch die Verlagerung der Konfigurations- und Provisioning-Planung an den Anfang des Prozesses wird sichergestellt, dass Sie ihr die Aufmerksamkeit widmen, die sie verdient.

Der beste Weg ist der Weg, der schnelle Ergebnisse bringt

Jedes Unternehmen tickt anders. Am besten fangen Sie mit dem an, was den Leuten, deren Zustimmung Sie brauchen, am wichtigsten ist. Wählen Sie ein Projekt, das so erfolgreich ist, dass Sie damit das Vertrauen der Unternehmensführung gewinnen. Dann haben Sie bereits Zeit gewonnen – und vermutlich auch zusätzliche Ressourcen und ein besseres Budget, sodass Sie von dort aus weitermachen und auf den ersten Erfolgen aufbauen können.

Noch etwas: Wenn Sie mit der Arbeit an der DevOps-Reise beginnen, werden Sie bald feststellen, dass Sie bessere Tools benötigen, um Zustand und Performance von Systemen und Geschäftsgang zu ermitteln. Die Antwort auf moderne Anwendungen lautet: Splunk Observability Cloud.

Mehr erfahren

Über den Autor

Greg Leffler

Greg leitet bei Splunk das Observability Practitioner Team und hat es sich zur Aufgabe gemacht, die frohe Botschaft von Observability in die Welt zu tragen. Gregs Laufbahn hat ihn aus dem Network Operations Center (NOC) zum Site Reliability Engineering (SRE) und von dort zum SRE-Management geführt, mit Zwischenstopps bei Security und Editor-Funktionen. Außer für Observability interessiert er sich beruflich für die Themen Fachkräftegewinnung, Weiterbildung, SRE-Kultur und die Steuerung effektiver Remote-Teams. Greg hat einen Master-Abschluss in Arbeits-/Organisationspsychologie der Old Dominion University in Norfolk, Virginia.

splunk>

Splunk Inc. verwandelt Daten in Taten. Mit einer Technologie, die darauf ausgelegt ist, dass Sie Daten in beliebigem Umfang untersuchen, überwachen und analysieren können – und auf dieser Grundlage Maßnahmen ergreifen.

puppet

Puppet treibt die Entwicklung in Richtung uneingeschränkter Softwareveränderung. Die revolutionäre Plattform ist der Branchenstandard bei Bereitstellungsautomatisierung und beim Betrieb der Software, mit der alles um uns herum läuft. Über 40.000 Unternehmen – darunter mehr als 75 % der Fortune-100-Unternehmen – nutzen die Open-Source- und die kommerziellen Lösungen von Puppet, um DevOps-Praktiken zu etablieren, ihr Lagebewusstsein zu schärfen und zuversichtlich Softwareänderungen voranzutreiben. Puppet hat seinen Hauptsitz in Portland, Oregon, und ist ein Privatunternehmen mit über 500 Beschäftigten weltweit. Mehr dazu finden Sie unter **puppet.com**.

Mehr erfahren

Splunk, Splunk> und Turn Data Into Doing sind Marken und eingetragene Marken von Splunk Inc. in den Vereinigten Staaten und anderen Ländern. Alle anderen Markennamen, Produktnamen oder Marken gehören den entsprechenden Inhabern.
© 2022 Splunk Inc. Alle Rechte vorbehalten.

22-22512-Splunk-The 5 Foundational DevOps Practices-LS-116_GER

splunk>
turn data into doing®