GitHub

Sieben DevOps Tipps für schnelleres App Development

Bei DevOps dreht sich vieles um die Geschwindigkeit: Schnelleres Application Development, schnellere Updates und Continuous Development sowie schnelleres Shipping. All das führt zu verkürzten System Development Lifecycles. Eine global angelegte DevOps-Studie ergab, dass sich leistungsstarke Entwicklungsteams 96 Mal schneller von Downtime erholen. Diese Teams haben außerdem eine 5 Mal niedrigere Change Failure Rate und können Code bis zu 46 Mal häufiger deployen. Erfolgreiche DevOps-Teams können Code somit innerhalb von Stunden und nicht Wochen deployen und recoveren.

Es ist daher kein Wunder, dass Unternehmen unterschiedlichster Größe und in allen vertikalen Märkten und globalen Regionen nach der DevOps-Methodik arbeiten. Eine weitere Umfrage ergab, dass fast drei viertel der weltweit befragten Unternehmen die DevOps-Methodik bereits eingeführt haben. Allerdings stehen dem Potenzial von DevOps auch echte Herausforderungen gegenüber. Dieselbe Studie ergab auch, dass die Beseitigung von Prozessengpässen zur Beschleunigung von Releases an die Spitze der Liste der Herausforderungen bei DevOps rückte. Die Optimierung der Zusammenarbeit und die Auswahl des richtigen Pakets von Automatisierungs- und anderen Tools sind weitere zentrale Herausforderungen.

Dieser Beitrag führt die sieben wichtigsten DevOps-Tipps auf, denen leistungsstarke Teams folgen, um maximalen Nutzen aus ihren DevOps-Bemühungen und -Investitionen zu ziehen.

1 DevOps-Kultur spielt eine wichtige Rolle, besonders wenn sie neu geschaffen wird.

Der Autor des jährlichen “State of DevOps-Berichts”, der nun im siebten Jahr erscheint, schreibt: „Für uns war es immer am wichtigsten, die Teams zu befähigen, beste Arbeit zu leisten, indem sie die kulturelle Kluft zwischen Entwicklung (Dev) und IT-Betrieb (Ops) überwinden.“ Eine umfassende Analyse der historischen Daten aus diesem Bericht zeigt, dass erfolgreiches DevOps meistens mit kleinen Initiativen beginnt. Dann benötigen Entwickler eine schlanke Plattform, über die sie frühe Erfolge und bewährte Praktiken sicher mit anderen Teams teilen können. Dieser Austausch wird dann auf mehrere Teams innerhalb einer Abteilung und schließlich auch auf andere Abteilungen ausgeweitet. Anders ausgedrückt: Man sollte es nicht gleich mit seinen DevOps-Bemühungen übertreiben.

2 Sicherheit von Anfang an tief in DevOps einbetten.

Das bedeutet, Sicherheitskonfigurationen so weit wie möglich zu automatisieren. Da sich DevOps-Organisationen von ihren elementaren Anfängen her entwickeln, wird die Sicherheitsrichtlinie zu einem wesentlichen Bestandteil des IT-Betriebs statt nur als Nachweis zu dienen, dass Audit-Anforderungen eingehalten werden. GitHub, die weltweit führende Software-Entwicklungsplattform kann sowohl in unternehmenseigener Infrastruktur als auch als SaaS-Lösung betrieben werden.

3 Möglichst weitgehende Automatisierung.

Spricht man über Automatisierung in DevOps, geht es in der Regel um die Automatisierung von Systemkonfiguration, Workflows und Systembereitstellung. Diese Infrastrukturautomatisierung dient vor allem dazu, die Arbeit der Entwickler im Einklang mit den Fähigkeiten des IT-Betriebs zu halten, die direkt an die Möglichkeit zur Bereitstellung von Software geknüpft sind. Automatisierung dient einem weiteren Kernelement, nämlich der Schaffung eines breiter verfügbaren Self Service in späteren Entwicklungsstadien, was natürlich mehr Effizienz bewirkt. Mithilfe einer Reihe von Vorlagen ermöglicht GitHub den Benutzern die automatische Konfiguration von Workflows, um den Status von Projektboard-Items mit den damit verbundenen Issues und Pull-Requests synchron zu halten. Benutzer können Workflows anhand von Triggern automatisieren, um zeitaufwendige manuelle Aufgaben bei der Verwaltung eines Projektboards zu eliminieren. Zudem können Benutzer ein GitHub-Projektboard kopieren, um es mit seinen Anpassungen für ähnliche Projekte wiederzuverwenden.

4 Die Vorteile einer offenen Plattform nutzen, um Transparenz zu schaffen.

Zusammenarbeit, ein Haupttreiber für Effizienz bei DevOps, liegt im Wesen von Open Source, wo Software in einem sehr kollaborativen, kollegialen und öffentlichen Prozess entwickelt wird. Der Umsatz durch Open Source ist in den letzten Jahren weltweit drastisch gestiegen, da Unternehmen aller Art Open Source intensiv in ihre DevOps-Methodik integriert haben. Webbasierte Softwareplattformen ermöglichen es den Benutzern, die Aktionen einer sehr großen Anzahl anderer Entwickler zu verfolgen, unabhängig davon, wo sie sich befinden oder für wen sie arbeiten. Diese Offenheit erzeugt die Möglichkeit optimaler Transparenz, wodurch sich die Zusammenarbeit und das Lernen der Softwareentwicklung radikal verbessern kann. Eine Studie zeigt, dass die Nutzer in einer solchen sozialen Entwicklungsumgebung umfangreiche und gegenseitig bereichernde Verbindungen und Hinweise aus dieser vernetzten Aktivität gewinnen. Es können Rückschlüsse auf die technischen Ziele und Visionen eines anderen bei der Bearbeitung von Code gezogen werden oder eine bessere Einschätzung, welche Projekte langfristig die besten Erfolgsaussichten haben gewonnen werden. Die Benutzer kombinieren diese Erkenntnisse dann zu effektiveren Strategien zur Koordination der Arbeit und zur Weiterentwicklung der eigenen technischen Kompetenzen. Mit mehr als 2,1 Millionen Organisationen und 40 Millionen Entwicklern weltweit bildet die Community der GitHub-Nutzer die weltweit größte Open- Source-Community. Sie tauschen Code aus, arbeiten in allen Phasen der DevOps-Methodik eng zusammen und entwickeln schließlich Software mit optimaler Effizienz und Zweckmäßigkeit, und zwar im Rahmen einer intuitiven und sehr benutzerfreundlichen Plattform.

5 Integration mit einem möglichst umfangreichen Toolset.

Laut der neuesten Auflage einer jährlichen Umfrage zu DevOps-Implementierungen in Unternehmen sind die beiden wichtigsten Herausforderungen für effektivere DevOps-Prozesse mitarbeiterbezogen (Beseitigung von Prozessengpässen und effizientere Zusammenarbeit der Entwickler). Dicht darauf folgt in dieser Liste jedoch eine dritte Herausforderung, nämlich die Auswahl und Implementierung von Tools für Anwendungsmonitoring, Verwaltung und Automatisierung. Die Fähigkeit, DevOps-Bemühungen mit einer möglichst großen Vielzahl von Tools zu integrieren, ist ein Eckpfeiler für den Erfolg von DevOps. Ob für Continuous Integration, Code-Review oder Code- Editing, es ist wichtig, dass Entwickler für jeden Schritt des Entwicklungsprozesses die richtigen Werkzeuge finden und diese dann vor Ort einsetzen können. GitHub bietet Zugang zu Hunderten von Tools, die Entwicklungsteams dabei helfen sollen, besser zu kommunizieren, ihre Arbeit zu automatisieren und bessere Software zu entwickeln. Gelegentlich wird nur ein speziell entwickeltes Werkzeug für eine bestimmte Aufgabe benötigt. GitHub ermöglicht es Anwendern, solche maßgeschneiderten Tools mit umfangreicherem Datenzugriff per GitHub GraphQL API zu erstellen; dies ist die gleiche API, die GitHub selbst für die internen Microservices verwendet.

6 Mit Inner Source die Effizienz von Open Source maximieren.

Eine schnell wachsende Zahl von Unternehmen übernimmt und nutzt Inner Source, eine Entwicklungsmethodik, die Best Practices aus großen Open-Source-Projekten für kommerzielle Softwareentwicklung verwendet. Diese Art von hochkomplexen, entwicklungsübergreifenden Teambemühungen erfordert die Koordination von hunderten, wenn nicht sogar tausenden von Entwicklern und Teams. Damit erweist sich Inner Source als herausragende Möglichkeit zur Bewältigung der vielen Herausforderungen, die große DevOps-Projekte mit sich bringen, und als ein großartiges Instrument zur Effizienzsteigerung durch die Wiederverwendung von Code. Unternehmen können auf Open-Source-Best- Practices im Unternehmenskontext zugreifen und sie nutzen. Auf GitHub ist die größte Open-Source- Community der Welt zuhause. Entwickler und Teams auf der GitHub-Plattform können ein Projekt für jeden beliebigen Zweck im Einklang mit Open-Source-Lizenzen anzeigen, ändern und verteilen. Darüber hinaus bietet GitHub eine breite Palette von Selbsthilfe-Anleitungen zur Identifizierung von Entwicklern für ein Projekt, zum Aufbau offener Communities, zu Wartung und Verwaltung sowie Open/Inner-Source-Erfolgs-Metriken, um nur einige zu nennen.

7 Kontinuierliches Feedback, Integration, Änderungsmanagement und Implementierung fördern.

Die Kenntnis gängiger Praktiken und Industriestandards in DevOps ist ein wichtiger Aspekt - wichtiger ist jedoch, diese zu integrieren und tief in einen Prozessablauf einzubetten, der quer durch Daten- und Informationssilos von der Entwicklung über die Bereitstellung bis hin zum Betrieb verläuft. Zu diesem Zweck setzt Continuous Integration auf die Automatisierung von Tests, um sicherzustellen, dass beim Einfügen von neuem oder geändertem Code in eine Anwendung die Qualität auf der Anwendungsebene gewährleistet bleibt. Das Änderungsmanagement geht auf den Betrieb ein und gibt Aufschluss darüber, welche anderen Systeme betroffen sein könnten und welche Folgen oder Möglichkeiten eine Änderung auf einer allgemeineren Ebene haben könnte. Inkrementelle Bereitstellungsstrategien fallen oft operativen Einschränkungen zum Opfer, die die kontinuierliche Bereitstellung einschränken oder gar verhindern können. Dies ist zumeist eine Herausforderung für die Mitarbeiter, nicht der Technologie und das Gegenmittel ist die Einbeziehung aller Beteiligten in die Anwendungsentwicklung, den Betrieb und den Support. Diese Art von kontinuierlichem abteilungsinternem Input erhöht die Wahrscheinlichkeit erheblich, dass eine neue Anwendung auch die Bedürfnisse und Erwartungen der Benutzer erfüllt.

GitHub, dem mehr als 40 Millionen Entwickler und die Hälfte der Fortune-500-Unternehmen vertrauen, hilft DevOps-Teams jeder Größe bei der sicheren Zusammenarbeit und der schnelleren Bereitstellung eines besseren Kundenerlebnisses. Um Ihre kostenlose Testphase zu starten oder mehr über die On-Prem und SaaS- Lösungen von GitHub zu erfahren, besuchen Sie **https://github.com/enterprise**.

[1] "2017 State of DevOps Report," Puppet, 2017.

[2] "2018 State of DevOps Report: Practical guidance for your DevOps evolution," Puppet blog, September 2018.

[3] "Social coding in GitHub: Transparency and Collaboration in an Open Software Repository," jsntsay.com, 2012.