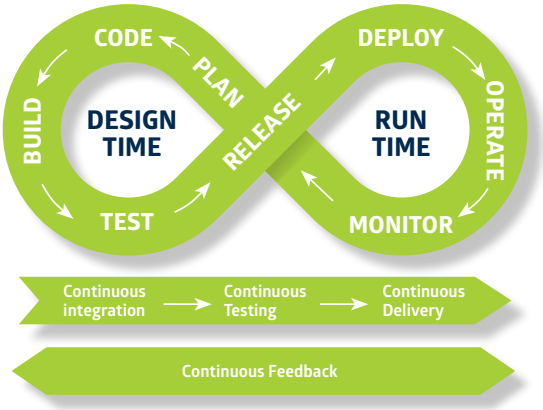
5 Techniken, mit denen DevOps produktiver werden

Der DevOps-Ansatz hat sich durchgesetzt. Doch wieviel Charme das Entwicklungs-Paradigma ausstrahlt, hängt vor allem davon ab, wie es umgesetzt wird. Sinn ergibt er vor allem in Verbindung mit containerisierten Anwendungen. Aber die haben so ihre Tücken. Hier sind fünf Tipps, wie Du auch mit einem kleinen DevOps-Team sicher, produktiv und zielgenau Anwendungen entwickeln kannst, die Dein Unternehmen erfolgreich macht.

DevOps bezeichnet eine Kultur, die auf der produktiven Zusammenarbeit von Development und Operations basiert. Grundlage sind gegenseitiges Vertrauen und die Bereitschaft zum Wissensaustausch mit dem Ziel einer reibungslosen Kooperation. Mittel zum Zweck sind gemeinsame Werkzeuge und Prozesse. Das übergeordnete Ziel heißt Effizienz durch Automatisierung.

Früher stellte die Entwicklung („Development") ein neues Release zur Verfügung, das Operations dann in Betrieb nehmen sollte. Tauchte ein Fehler auf, schoben sich die beiden Parteien gegenseitig die Verantwortung zu. Das Go-Live wurde dadurch hinausgeschoben oder völlig unmöglich. In einer DevOps-Architektur hingegen gibt es keine festen Release-Termine. Design- und Run-Time sind in einer Endlosschleife miteinander verbunden, die von der Planung über die Codierung, den Test, das Deployment und den Betrieb inklusive Monitoring wieder zurück in den Design-Prozess führt. Auf diese Weise wird alles, was in der Entwicklung passiert, umgehend im Betrieb geprüft. Daraus ergeben sich ein ständiges Feedback und ein immerwährender Integrations-, Test- und Delivery-Prozess. DevOps nutzen dafür meist eine Art Fertigungsstraße, auch CI/CD-Pipeline (Continuous Integration/Continuous Delivery) genannt.



1 CLOUD NATIVE ENTWICKLUNG

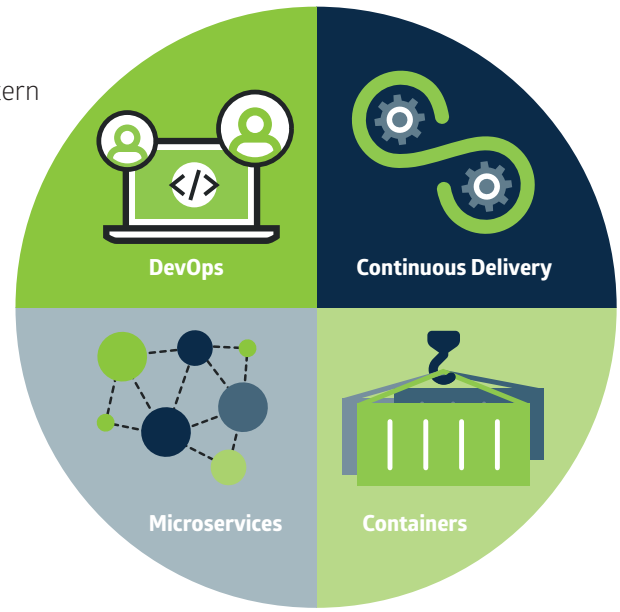
DevOps bedeutet nicht nur eine neue Entwicklungskultur, sondern in aller Regel auch eine zeitgemäße, also Cloud-basierte Architektur. Die Anwendungen werden nicht mehr für bestimmte Hardware und Betriebssysteme entwickelt, sondern als Services, die in unterschiedlichen physischen Umgebungen laufen können. Die Grundlage dafür sind sogenannte Container. Durch diese Containerisierung von Anwendungen, die in Microservices aufgeteilt und in leichtgewichtige Container verpackt werden, können eine Vielzahl von Server- und Betriebssystemen bereitgestellt und orchestriert werden: „Cloud-Native-Technologien ermöglichen Unternehmen, skalierbare Anwendungen in modernen, dynamischen Umgebungen wie öffentlichen, privaten und hybriden Clouds zu entwickeln und auszuliefern". Im Fachjargon spricht man von „Cloud Native Development".

Im Gegensatz zu traditionellen Anwendungen, die mit dem Fokus auf eigene Installationsroutinen ausgeliefert wurden, sind Cloud-Native-Anwendungen dafür geschaffen, sowohl on premises als auch in der Public-, der Private- oder einer Hybrid Cloud betrieben zu werden. Dazu gehört auch, dass sie sich mit anderen Cloud-Diensten integrieren lassen. Auf diese Weise erstellte Applikationen sind keine monolithischen Blöcke. Sie setzen sich aus entkoppelten Microservices zusammen, die über APIs miteinander kommunizieren. Änderungen beschränken sich auf den jeweiligen Service. Der Rest der Anwendung bleibt davon unberührt.

Um das Deployment dieser relativ komplexen Anwendungen zu vereinfachen, werden die Microservices in „Containern" zusammengefasst. Der modulare Aufbau führt dazu, dass die Anwendungen flexibel, skalierbar und nicht an ein bestimmtes Base-Image gebunden sind. Sie lassen sich in dynamischen Umgebungen implementieren und betreiben.

Die Vorteile einer Cloud-Native-Entwicklung sind:

- mehr Agilität und Geschwindigkeit
- kürzere Release-Zyklen
- Unabhängigkeit von bestimmten (Cloud-)Anbietern
- hohe Skalierbarkeit und Verfügbarkeit
- größere Prozess- und Kosteneffizienz



2 DEVOPS-AUTOMATISIERUNG

Das klingt zunächst einmal gut, hat aber auch ein paar Ecken und Kanten. So muss das DevOps-Team vieles bewältigen, das per se mit manueller Arbeit zu tun hat. Dazu zählen Test- und Deployment-Abläufe sowie die Verwaltung der Container. Wo sind welche Services enthalten? Wie lassen sich diese zu einer Lösung verbinden? Welche Umgebungen sind involviert? Diese Informationen zusammenzubringen ist eine Aufgabe, die sich händisch kaum bewältigen lässt.

Um die DevOps-Teams zu entlasten, ist deshalb eine möglichst weitgehende Automatisierung notwendig. Softwaretests und Deployment-Prozesse sollten ohne manuelle Eingriffe möglich sein. Die Idee des Continuous Delivery setzt voraus, dass sich der Code mit automatisierten Routinen testen und in ein Repository, zum Beispiel das Open-Source-System „Git", hochladen lässt, wo er stets in der aktuellsten Version verfügbar ist. Nur eine hochgradig automatisierte und optimierte CI/CD-Pipeline verkürzt tatsächlich die Zyklen für Releases, Updates und Bugfixes.

Aber keine Bange! Es gibt auf dem Markt eine Reihe von Orchestrierungs-Tools. Damit lassen sich Container-basierte Umgebungen und Anwendungen automatisiert einrichten, betreiben, warten, verwalten und skalieren. Sie versetzen die DevOps-Teams also in die Lage, CI/CD-Pipelines vollständig zu automatisieren. Zu den Features dieser Softwarewerkzeuge gehören automatisches Rollout und Rollback, selbständige Skalierung, „Auto-Healing", Secrets- und Konfigurations-Management sowie Service Discovery.

3 ORCHESTRIERUNG MIT KUBERNETES

Einige dieser Tools sind proprietär, binden die Kundschaft also an die Umgebungen und Release-Zyklen eines Herstellers. Deshalb entscheiden sich viele Unternehmen lieber für eine Open-Source-Lösung. Eine quelloffene Lösung für die Orchestrierung von Container-basierten Umgebungen und Anwendungen ist das inzwischen etablierte Projekt Kubernetes.

Entwickelt wurde K8s, wie Kubernetes auch genannt wird, von Google. Die erste Version erblickte 2015 das Licht der IT-Öffentlichkeit. Mittlerweile hat Google das Projekt an die Cloud Native Computing Foundation (CNCF) übergeben, unter deren Ägide nun die Weiterentwicklung der Software steht.

Eine Kubernetes-Umgebung basiert auf einer hierarchischen Architektur. Deren kleinste Einheit ist ein Pod, der einen oder auch mehrere Container enthalten kann. Die Pods laufen als Prozesse auf einem Node, unterschiedliche Nodes lassen sich zu Clustern zusammenfassen. Der K8s-Master steuert die Nodes mitsamt den Pods und Containern, weist ihnen Ressourcen zu und erteilt ihnen Aufgaben. Wichtige Prozesse dieses „Master" betreffen den API-Server, den Key-Value-Store („etcd"), den Controller-Manager und den Scheduler.

K8s wird von vielen Cloud-Computing-Plattformen unterstützt, so zum Beispiel Amazon Web Services (AWS), Microsoft Azure, Google Cloud oder Oracle Cloud Infrastructure. Gleichzeitig arbeitet das Orchestrierungswerkzeug mit unterschiedlichen Container-basierten Virtualisierungssystemen und Container-Engines zusammen.

Weil die Container-Steuerung unabhängig von der jeweiligen Cloud-Umgebung ist, lassen sich containerisierte Anwendungen mit Kubernetes zwischen unterschiedlichen Cloud-Umgebungen verschieben. Zudem ist der komplette Betrieb der Container automatisch skalierbar. So geschieht die Lastenverteilung genauso automatisiert wie die Zuteilung der benötigten Ressourcen, also Speicherbedarf und Rechenzeit, was besonders in hochverfügbaren Systemen wichtig ist. Zudem stellt K8s für zustandslose Container persistenten Storage bereit.

Zusammengefasst hat die Orchestrierung mit Kubernetes folgende Vorteile:

- Container lassen sich schneller ausrollen.
- Bereitstellung und Skalierung der Container geschieht ohne händische Prozesse.
- Die so erstellten Systeme sind hochverfügbar und sehr stabil.
- Es werden automatisch Redundanzen vorgehalten.
- Ressourcen für die Container-Anwendungen werden schnell und automatisiert bereitgestellt.
- Updates lassen sich ohne Beeinträchtigung der Service-Verfügbarkeit einspielen.
- Der Betrieb der Container geschieht unabhängig von der jeweiligen Plattform.
- Zugleich sind Container problemlos zwischen verschiedenen Umgebungen transferierbar.

Fazit: Kubernetes vereinfacht die Software-Bereitstellung in DevOps-Umgebungen, weil es die Handhabung der Infrastruktur automatisiert. So werden die DevOps produktiver, und sie können sich besser auf ihre Kernaufgaben konzentrieren. Schließlich geht es nicht darum, Infrastrukturen zu managen, sondern Anwendungen zu entwickeln, die das Unternehmen voranbringen.

4 RISIKO-BEWUSSTSEIN SCHÄRFEN

Kein Wunder, dass fast alle Unternehmen, die für die Cloud entwickeln, Kubernetes zumindest ernsthaft in Betracht ziehen! Einer aktuellen Studie des CNCF zufolge haben 96 Prozent der befragten Organisationen das Werkzeug entweder schon im Einsatz oder beabsichtigen das zumindest. Vor allem Betriebe, die mehrere Plattformen parallel betreiben, profitieren davon. Ohne eine Technik wie Kubernetes wäre das Deployment von Cloud-Native-Anwendungen deutlich komplexer: Für jede Umgebung müsste es neu definiert und implementiert werden.

Allerdings gilt es, auch die Kehrseite der Medaille zu betrachten. Kubernetes-Installationen sind wirklich komplex und erfordern Fachwissen auf Seiten der DevOps-Teams. Ganz abgesehen vom allgegenwärtigen Mangel an IT-Fachkräften sind die Träger:innen von Kubernetes-Knowhow besonders dünn gesät. Wer sich für den Einsatz von K8s entscheidet, sollte deshalb die Ausbildung der Mitarbeiter:innen gleich mitberücksichtigen. Und einen Dienstleister wählen, der über nachgewiesene Expertise verfügt.

Ein besonders kritischer Punkt ist das Multi-Tenant-Management. Jeder „Mieter“ (Tenant) einer Container-Architektur hat seine eigenen Anforderungen, beispielsweise in puncto Performance oder Sicherheit. Darüber hinaus muss sichergestellt sein, dass keiner von ihnen die Cluster-Ressourcen über Gebühr nutzen kann. Deshalb sind die Administratoren gefordert, für jeden Tenant ein eigenes Setup aufzubauen und getrennt zu verwalten.

Ein weiterer Knackpunkt ist die Nutzung der verfügbaren Ressourcen. Bei einem unzureichenden Design ist die Architektur möglicherweise schlecht auf unvorhergesehene Cloud-Migrationen vorbereitet. Um nicht in die Bredouille zu geraten, tendieren IT-Teams deshalb dazu, für jeden Service fest zugeordnete CPU-, Speicher- und Netz-Ressourcen vorzuhalten. So entstehen Silos mit schlummernden „Reserven“ für den Fall einer höheren Auslastung. Damit sinkt die Auslastung, und die Kosten steigen.

5 MANAGED KUBERNETES

Die IT-Verantwortlichen stehen heute unter dem Druck, geschäftskritische Software immer schneller und in immer heterogeneren Systemlandschaften bereitzustellen. DevOps und Cloud-Native-Entwicklung mit containerisierten Anwendungen erleichtern diese Aufgabe, schaffen aber neue Komplexität. Werkzeuge wie Kubernetes sind hilfreich, aber nur für Fachleute handhabbar – und die sind rar.

Hier helfen wir mit unserer Kubernetes-Lösung „MetaKube“ Deinen DevOps-Teams, ihre Kubernetes-Umgebung aufzubauen und zu automatisieren. Dabei übernehmen wir nicht nur die Einrichtung, sondern bieten auch Services, die über das reine Lifecycle-Management hinausgehen. Dazu gehören Load Balancer as a Service, Backup und Recovery, Monitoring der Cluster und Dashboards für den Überblick. MetaKube nimmt Dir und Deinem DevOps Team somit den komplizierten Teil des Erstellens, Betreibens und Updatens eines Kubernetes Clusters ab und ermöglichen einen Rundum-Sorglos-Betrieb – so kannst Du Dich voll und ganz auf Dein Kerngeschäft konzentrieren.

Weitere Infos findest Du hier: sys11.it/devops-kubernetes

SysEleven

produkt@syseleven.de | 49 (0)30 233 2012-20

SysEleven GmbH | Boxhagener Straße 80 | 10245 Berlin