heben. Spätestens danach sollte es möglich sein, mit den Ergebnissen anhand der geschätzten Nutzerlast die nötige Skalierung der Produktivumgebung durchzuführen. Weiter kann dann anhand der Ergebnisse die Einhaltung vereinbarter Service Level Agreements geprüft werden und somit optimalerweise eine Freigabe erteilt werden.

Kontinuierliche Überwachung der Produktion

Mit der erhaltenen Freigabe sollten Last- und Performance-Tests nicht enden. Gerade im Zuge sich weiter verbreitender Continuous Integration und Continuous Delivery, aber auch klassischer Produkt-Updates, sollte sich die Durchführung des Quality Gate wiederholen. Zusätzlich sollte eine kontinuierliche Überwachung der Produktion instanziiert sein. Diese überwacht durch technisches Monitoring die Verfügbarkeit und Reaktionszeiten der Anwendung und damit eine fortwährende Einhaltung der SLAs.

Zu guter Letzt muss sichergestellt sein, dass die Nutzung der Anwendung mithilfe fachlichen Monitorings qualitativ ausgewertet werden kann. Um etwa bei einer eigenentwickelten Oracle-ADF-Anwendung die durch ADF-Task-Flows abgebildeten, fachlichen Anwendungsteile zu identifizieren, bietet sich das Oracle-Produkt „Real User Experience Insight" an, um das Konzept der Task-Flows zu verstehen und interpretieren zu können. In der Praxis hat der Autor jedoch erleben müssen, wie in der eingesetzten Version (korrespondierend zu ADF 11.1.1.6) dieses Werkzeug durch einen bekannten Bug in Exceptions lief und damit die Anwendung aushebelte. Somit bleibt zumindest für Eigenentwicklungen zu berichten, dass fachliches Monitoring oftmals nicht trivial ist, da vielen Standardwerkzeugen der Einblick in AJAX-Requests oder ADF Task Flows fehlt und somit meist eigenes fachliches Logging benötigt wird.

Fazit

Performance Monitoring sollte evolutionär betrachtet werden, um die Mitarbeiter mitzunehmen. Eine Bereitstellung umfangreicher Werkzeuge für ein wenig erfahrenes Team trägt nicht weit. Durch frühe und einfache erste Betrachtungen des Themas „Performance" wird Transparenz geschaffen, Know-how aufgebaut sowie für das Thema sensibilisiert.

Last- und Performance-Tests sind ein wertvolles Quality Gate. Annahmen zum Lastverhalten und somit die Skalierung von Produktions-Umgebungen sind abzusichern. Auch das komplexe Zusammenspiel diverser Systemkomponenten und deren passende Konfiguration sollte sich unter Last beweisen, da hier oftmals Nachbesserungsbedarf besteht.

Dabei sollte die Vergleichbarkeit zur Produktionsumgebung maximiert werden und damit die Belastbarkeit der Ergebnisse. Diese Tests sollten in einem iterativen Prozess verankert sein, um die gelieferte, hohe Qualität zu sichern und beizubehalten. Neue, Performance-relevante Anwendungsteile sollten dementsprechend immer Last- und Performance-Tests durchlaufen. Die Einhaltung der SLAs sollte auf Produktions-Umgebungen kontinuierlich überwacht werden.

Christian Kunzmann
ck@enpit.de

DevOps und Microservices beschleunigen Applikationsbereitstellung

Markus Eisele, Red Hat GmbH

DevOps verzahnt IT-Entwicklung und Betrieb miteinander und ermöglicht so die schnelle und schrittweise Bereitstellung von Software. Das Konzept bildet auch einen wichtigen Erfolgsfaktor für die Entwicklung und den Betrieb flexibler Microservices-Architekturen. Umgekehrt vereinfachen Microservices die Implementierung von DevOps.

Trends wie Big Data, Mobile, Cloud Computing und das Internet der Dinge stellen Unternehmen aller Branchen vor große Herausforderungen. Sie müssen ihre Geschäftsprozesse schnell und flexibel an diese Veränderungen und neue Anforderungen anpassen. Hier ist speziell die IT-Abteilung gefordert. Fachabteilungen erwarten, dass sie Anwendungen in hoher Qualität sehr schnell entwickelt und neue Funktionen sowie Updates zügig ausliefert. Kurzum: IT-Entwicklung und -Betrieb müssen agil sein. Um dies zu erreichen, gibt es zwei wichtige Optionen: Zum einen den Aufbau einer flexiblen Anwendungs-Architektur (unter anderem mit Microservices), zum anderen innerhalb der IT-Abteilung eine engere Zusammenarbeit von Entwicklung und Betrieb (DevOps), um Reibungsverluste zu minimieren und die Bereitstellung von Anwendungen zu beschleunigen. Beide Ansätze ergänzen sich gegenseitig.

Microservices als Ansatz für agile Anwendungen

Monolithisch strukturierte Anwendungen stoßen in puncto Agilität durchaus auch an ihre Grenzen. Ändern Entwickler nur einen kleinen Teil der Anwendung, muss die gesamte Applikation zumeist unter großem Aufwand neu getestet werden. Ziel einer Microservices-Architektur ist, dies deutlich flexibler zu gestalten.

Im Gegensatz zu monolithischen Architekturen bestehen Microservices aus lose

gekoppelten, voneinander unabhängigen Diensten beziehungsweise Services mit einer in sich abgeschlossenen, fachlichen Funktionalität. Dank Versionierung auf den Schnittstellen kann vielfach ausgeschlossen werden, dass Änderungen an einem der Services Einfluss auf die Funktionsweisen oder Eigenschaften eines anderen Service haben. Zudem lassen sich Updates mit Erweiterungen oder Verbesserungen gezielter und häufiger vornehmen, ohne die gesamte Anwendung aktualisieren zu müssen. Weitere Vorteile: Bereits aus dem Umfeld von SOA ist bekannt, dass sich kleinere Services mit definierten Schnittstellen deutlich besser austauschen lassen. Auch die Skalierbarkeit ist erheblich besser als in monolithischen Architekturen, da sich Dienste unabhängig und bedarfsgerecht skalieren lassen.

In letzter Zeit werden vor allem Browser-basierte Anwendungen häufiger als Microservice konzipiert. Eine E-Commerce-Anwendung wird dann nicht als monolithisches Ganzes konzipiert und entwickelt, sondern in einzelnen Modulen, zum Beispiel mit eigenen Komponenten für die Produktsuche, Bestellung oder Produktbewertungen. Im Verbund ergeben die Module die Gesamtfunktionalität einer Anwendung und enthalten neben den fachlichen Funktionen auch die relevanten Oberflächen-Anteile.

Die verteilte Architektur der Microservices bedingt aber, dass sich die Entwickler auch mit dem Betrieb der Anwendungen auseinandersetzen müssen, um Fehler bei der Verknüpfung und Koordination der einzelnen Services zu vermeiden. Hier hilft DevOps weiter.

DevOps verknüpft Entwicklung und Betrieb

Der Begriff setzt sich zusammen aus „Dev" für die Software-Entwicklung („Development") und „Ops" für den IT-Betrieb („Operations"). Ziel von DevOps ist eine enge Zusammenarbeit von Entwicklung und Betrieb durch den Aufbau von interdisziplinären Teams. Es gibt also keine direkte Trennung der beiden Bereiche mehr. Vielmehr ist jedes Team-Mitglied für seine Komponente verantwortlich, zum Beispiel hinsichtlich Konzeption, Entwicklung, Test oder Inbetriebnahme. Hintergrund: Klassisch betrachtet, verfolgen Betrieb und Software-Entwicklung unterschiedliche Ziele. Während der IT-Betrieb die Priorität auf höchste Verfügbarkeit und stabile sowie kostengünstige Betriebsprozesse legt, geht es den Entwicklern darum, neue Anwendungen oder Funktionen möglichst schnell bereitzustellen und flexibel auf kurzfristig geänderte Spezifikationen zu reagieren. Probleme führen hier schnell zu Spannungen und gegenseitigen Vorwürfen. Das Ziel von DevOps ist es, diese Reibungsverluste zu verhindern, bestenfalls zu eliminieren.

Die organisatorischen Veränderungen im Zuge des Einsatzes interdisziplinärer Teams sind vielfach auch mit einem kulturellen Wandel innerhalb des Unternehmens verbunden. Die Mitglieder aus beiden Abteilungen treffen sich regelmäßig, teilen ihr Wissen und pflegen eine offene Feedback-Kultur, vor allem aber arbeiten sie tatsächlich gemeinsam an einem bestimmten Thema. Basis dafür sind die Festlegung von gemeinsamen Zielen und Werten sowie die permanente Bereitschaft zum Lernen. Die Entwickler machen sich mit den Anforderungen und Prozessen des Betriebs vertraut, die Administratoren haben Zugriff auf den Source-Code oder die Build-Tools. Auch Metriken und Planungen sollten offen für alle sein. Das Ergebnis dieser engen Zusammenarbeit sind interdisziplinäre Teams, die gemeinsam Verantwortung für einen kompletten Teil einer Anwendung tragen – von der Konzeption über die Entwicklung und den Test bis zum Betrieb.

Agile Methoden

Da DevOps das flexible Vorgehen der agilen Softwareentwicklung unterstützt, ist es möglich, Anwendungen bereits zu testen und zu nutzen, wenn deren Entwicklung noch nicht abgeschlossen ist. DevOps wendet agile Methoden auch im IT-Betrieb an und verbindet Standardmodelle für Softwareentwicklung und IT-Betrieb miteinander. Die Verzahnung führt zu kürzeren Release-Zyklen und minimiert das Risiko ungetesteter Code-Elemente, weil die eingesetzten Werkzeuge und Verfahren über den gesamten Software-Erstellungsprozess hinweg identisch sind.

Die Vorteile von DevOps belegt beispielsweise der Software-Hersteller Puppet Labs in Portland, Oregon, im „2015 State of DevOps Report". Demnach stellen Unternehmen mithilfe von DevOps ihre Anwendungen um bis zu dreißigmal schneller bereit als Firmen, die traditionell arbeiten – und sie verzeichnen 60 Prozent weniger Störungen oder Ausfälle bei Anwendungen. Laut Puppet Labs beheben sie Ausfälle im Schnitt auch 168-mal schneller als Firmen, die Entwicklung und IT-Betrieb nicht eng verzahnen.

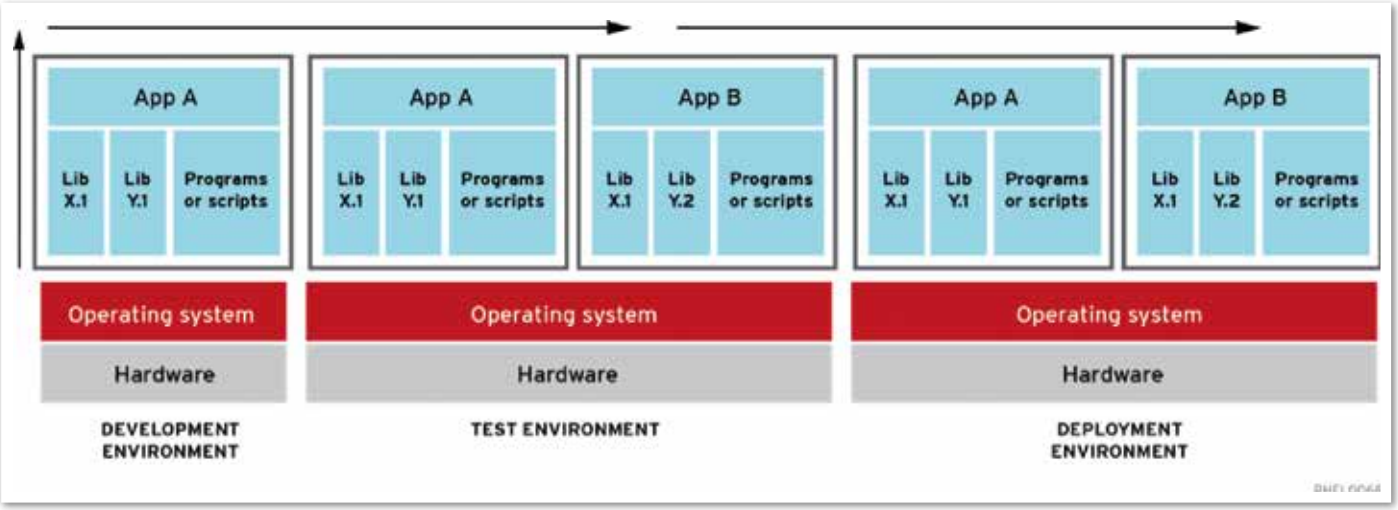

Abbildung 1: Container bieten eine konsistente Umgebung für Anwendungen, da sie diese mit allen Komponenten kapseln, die sie benötigen, – das vermeidet komplizierte Konflikte mit anderen Anwendungen. Zudem lassen sich die Applikationen einfacher portieren.

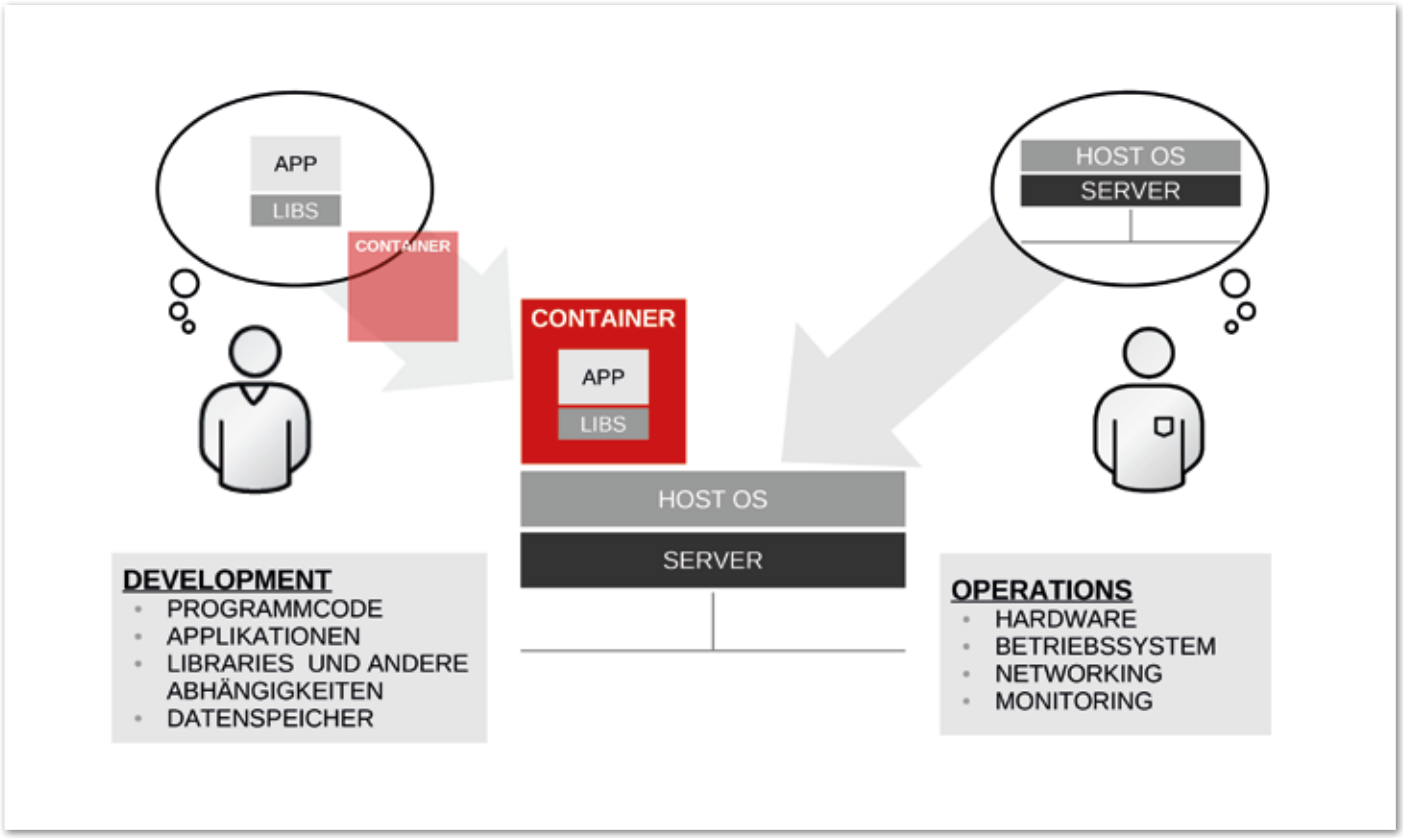

Abbildung 2: Container ermöglichen eine kontinuierliche Auslieferung von Applikationen und fördern den Einsatz und die Verbreitung von DevOps-Ansätzen

Prozesse und Technologien

Wichtige Elemente von DevOps sind, neben einer Kultur der Zusammenarbeit, automatisierte Prozesse und moderne Technologien wie Container *(siehe Abbildung 1)*. Letztere kapseln und isolieren Anwendungen mit allen benötigten Komponenten in einem oder mehreren Paketen. Daher lassen sich Anwendungen schnell, einfach und vollständig konfiguriert bereitstellen. Zudem eignen sich Container durch die Kapselung sehr gut als Basis für die Bereitstellung von Microservices.

Ein zentrales Ziel von DevOps ist die Automatisierung möglichst vieler Arbeitsvorgänge, um die Release-Prozesse zu beschleunigen. Dazu muss das IT-Team aus Entwicklern und Betrieb zunächst alle einzelnen Arbeitsschritte bis ins kleinste Detail definieren und standardisieren. Das betrifft Entwicklung, Testen sowie Produktion und schließt auch Fragen des Betriebs mit ein. Durch die detaillierte Aufteilung der Prozesse ist es auch möglich, kleine Veränderungen schnell umzusetzen und die Frequenz der Releases zu erhöhen – und damit die zentrale Anforderung von Fachabteilungen hinsichtlich einer schnellen und flexiblen Bereitstellung von Applikationen zu erfüllen.

Eine wichtige Voraussetzung für eine erfolgreiche DevOps-Nutzung sind einheitliche Bewertungskriterien („Measurements") für die Prüfung der Applikation, ihrer zugehörigen Komponenten und der zugrunde liegenden Prozesse, um die Qualität der Software-Releases dauerhaft zu sichern. Dies erfolgt über Software-Tests, die auf einheitlichen, transparenten Metriken beruhen. Die DevOps-Teams können die Software-Tests und die Qualitätssicherung auch über Cloud-Plattformen beschleunigen und so kostenoptimiert Infrastrukturen simulieren, die der späteren Produktionsumgebung ähneln.

Von Bedeutung im DevOps-Umfeld sind nicht zuletzt auch Continuous-Integration- und Continuous-Delivery-Prozesse *(siehe Abbildung 2)*. Während sich Continuous Integration mit dem automatisierten Build-Prozess sowie Unit- und Funktionstests beschäftigt, geht Continuous Delivery weiter bis hin zum automatisierten Rollout. Da die einzelnen Arbeitsschritte der Software in kleineren Einheiten abgebildet sind, lassen sich Funktionserweiterungen direkt automatisiert testen. Entwickler erhalten damit ein schnelles Feedback über mögliche Fehler. Sind die Tests erfolgreich, lässt sich das neue Software-Paket kurzfristig auf dem Produktivsystem installieren. Die Automatisierung erlaubt damit die kontinuierliche Bereitstellung kleiner Funktionen.

Microservices und DevOps ergänzen sich

Generell können IT-Abteilungen mithilfe von Microservices-Architekturen besser auf die Anforderungen aus den Fachabteilungen reagieren und Anwendungen kurzfristiger realisieren. Sehr wichtig ist dabei die enge Verzahnung zwischen Entwicklung und IT-Betrieb, also die Einführung des DevOps-Modells. Ein weiterer Vorteil des Microservices-Ansatzes ist, dass auch kleinere Teams individuelle Dienste entwickeln und testen sowie Releases schnell bereitstellen können – damit wird auch die Implementierung von DevOps vereinfacht. Auf diese Weise ergänzen sich Microservices und DevOps gegenseitig. Das Duo ermöglicht in Verbindung mit Containern die Realisierung einer flexibleren und effizienteren Infrastruktur, liefert Applikationen, die diese Infrastruktur optimal nutzen, und etabliert Prozesse, mit denen diese Anwendungen schnell und mit hoher Qualität entwickelt und bereitgestellt werden können.

Markus Eisele
info@redhat.de