Michael Brokmann
(D1BROK@de.ibm.com)
leitet innerhalb der Cloud-Organisation der IBM Deutschland GmbH das Cloud-Architekturteam. Er hat über 20 Jahre Erfahrung in der IT-Branche, überwiegend in technischen Führungspositionen bei IBM. Vor der Übernahme seiner aktuellen Position war er der verantwortliche IBM Client Technical Advisor für die Generali Insurance Group. Er verbrachte mehr als 10 Jahre in der IBM Software Group als Softwarearchitekt mit starkem Fokus auf SOA-und Systems-Management. Er ist aktiv in die IBM Cloud Computing Reference Architecture einbezogen und dort u. a für den Hybrid Cloud Workstream verantwortlich. Seit einigen Jahren ist er auch aktiv an Vorlesungsreihen zum Thema Cloud Computing an der Universität Stuttgart beteiligt.

Robert Michel
(michel@de.ibm.com)
ist IT-Architekt innerhalb der Software Group der IBM Deutschland GmbH. Seine Lösungsschwerpunkte liegen im Bereich Private Cloud- und DevOps-Architekturen. Er kann auf über 15 Jahre Erfahrung im Bereich System- und Service-Management zurückblicken. Projekterfahrungen reichen dabei von klassischen System-Managementthemen über Service-Management gemäß ITIL, Software und Application Lifecycle-Managementlösungen bis hin zur Umsetzung von Cloud-Architekturen. Neben seiner praxisnahen Erfahrung verfügt er auch über fundierte theoretische Kenntnisse u. a. durch Zertifizierungen in dem Bereich ITIL (V3 Expert).

DevOps und Cloud – Eckpfeiler einer „Continuous Delivery Pipeline“

DevOps und Cloud gehören aktuell wohl zu den meistverwendeten Schlagworten in der IT-Branche. DevOps steht für „Development und Operations“ und bezeichnet primär die Zusammenarbeit von Entwicklung und Betrieb, um Software schneller, häufiger und risikoärmer in laufende Betriebsprozesse übernehmen zu können. Dabei sind Entwicklungsverantwortliche, Testverantwortliche, Betriebsverantwortliche, Architekten, Entwickler, Systemadministratoren alle gleichermaßen betroffen. Die Umsetzung der DevOps-Prinzipien lässt sich nur gemeinsam lösen. Dieser Artikel richtet sich an diejenigen Leser, die sich sowohl einen Überblick über die Prinzipien und Werte von DevOps verschaffen wollen, als auch an konkreten Beispielen der Umsetzungen interessiert sind. Was ist DevOps, welche Herausforderungen im Bereich der Software Delivery werden adressiert, warum wird dieses doch recht „alte Phänomen“ nun wieder aktuell, was hat das Ganze mit Cloud zu tun und welche Lösungen und Konzepte stellt die IBM für diese Herausforderungen zur Verfügung? Diese Fragen werden im Folgenden genauer betrachtet. Ein konkretes Kundenbeispiel kann bei Interesse ebenfalls abgerufen werden.

DevOps versucht, die Lücke zwischen agiler Entwicklung und ITIL-gesteuertem Betrieb zu schließen, ohne bereits getätigte Investitionen überflüssig zu machen. Dies geschieht immer mittels einer Kombination aus Prozessanpassung und Einführung geeigneter Werkzeuge (vgl. Abbildung 1).

Der hauptsächliche Fokus von DevOps richtet sich auf die Aktivitäten zwischen der Programmierung und dem Betrieb, also Build, Deploy und Test. Es wird dabei mithilfe geeigneter Werkzeuge und Cloud-Technologien eine möglichst vollständige Automatisierung dieser Phase umgesetzt.

DevOps – was ist das und warum wird dadurch alles besser?

Was ist DevOps? DevOps ist keine technische Lösung, kein Produkt und auch kein

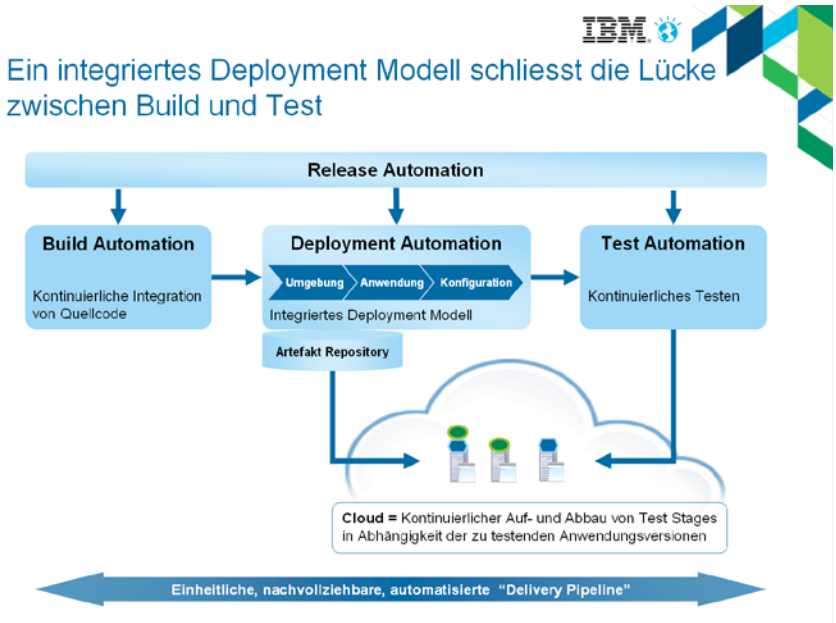

Abb. 1: Release Automation

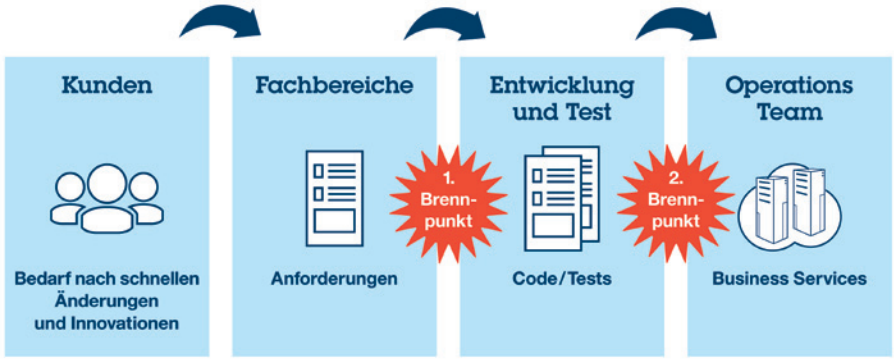

Abb. 2: Brennpunkte im Rahmen der Software Delivery

fertiger Methodenbaukasten, sondern ein Satz von Prinzipien, der die Kommunikation, Zusammenarbeit und Integration zwischen IT-Entwicklungs- und Betriebsorganisationen adressiert. Zu diesen Grundwerten gehören u. a.:

- bereichsübergreifende Zusammenarbeit,
- Entwicklung und Test gegen produktionsnahe Systeme,
- häufiges und regelmäßiges Deployment als wiederholbarer und zuverlässiger Prozess
- kontinuierliche Überwachung und Validierung von operationalen Qualitätsfaktoren.

Erreichen will man hierdurch eine schnellere Umsetzung der Anforderungen aus den Fachbereichen, eine Reduzierung von Risiken und Kosten sowie eine Qualitätsverbesserung der produzierten Software.

Abbildung 2 zeigt die beiden wesentlichen Brennpunkte, die sich im Rahmen der Software Delivery innerhalb als auch im Übergang zum Betriebsbereich zeigen.

Die Anforderungen an Qualität und insbesondere Geschwindigkeit aus den Fachbereichen an die IT-Entwicklungsabteilungen haben sich in den letzten Jahren erheblich gewandelt. Traditionelle Entwicklungsmethoden, wie beispielsweise RUP (Rational Unified Process), halten diesen Anforderungen nicht mehr stand. Eine Konsequenz daraus ist die zunehmende Adaption agiler Entwicklungspraktiken (in Abbildung 2 die Konsequenz des ersten Brennpunktes) im Gegensatz zu statischen Projektplanungen.

Diese Änderungen innerhalb des Software-Entwicklungsbereiches reichen jedoch nicht aus, um eine höhere Dynamik und Geschwindigkeit im Ergebnis zu erzielen. Agile Entwicklungsprozesse erzeugen kontinuierlich neue Versionen, die zum Test und Deployment anstehen. Solange die Aufgaben der Betriebsabteilung jedoch nicht der gleichen Dynamik unterliegen, ergibt sich ein Engpass im Deployment und Testing neuer Anwendungsversionen (der 2. Brennpunkt in Abbildung 2).

Somit wächst im Betrieb und in der Systembereitstellung die Forderung nach einem höheren Grad an Automation für die Bereitstellung und Konfiguration der Test- und Produktionsumgebungen. Die Bewältigung dieser Herausforderung bedingt eine Zusammenarbeit zwischen Entwicklung und Betrieb – und zwar nicht wie bisher in Form sequentieller und getrennt voneinander ablaufender Arbeitsschritte, sondern in einer frühzeitig im Entwicklungsprozess stattfindenden gemeinsamen Planung des „nächsten Releases“.

Dazu wird der in der Entwicklung begonnene Gedanken einer *kontinuierlichen Integration* der Entwicklungsergebnisse erweitert in das Praktizieren eines *kontinuierliches Deployments* und eines *kontinuierlichen Testens* der Anwendungen. Das bedeutet, dass mit jeder neuen Version einer Anwendung, die in der Entwicklung entsteht, ein automatisierter Test- und Deployment-Prozess initiiert wird.

Dieses Prinzip einer durchgängigen Ende-zu-Ende-Automation wird auch als „Delivery Pipeline“ bezeichnet (vgl. Abbildung 3).

Die Umsetzung dieser „Delivery Pipeline“ bedingt jedoch Veränderungen sowohl in Prozessen als auch in der Technologie. Die Bereitstellung der Infrastruktur und die Konfiguration der Infrastruktur werden von einer der Entwicklung nachgelagerten Tätigkeit zu einer von vornherein im Entwicklungsprozess berücksichtigten Aktivität.

Sämtliche Schritte zur Bereitstellung der Infrastruktur und Konfiguration der Anwendung werden in einem Modell während der Design- und Entwicklungsphase definiert und als Teil des Release zusammen mit der Anwendung verwaltet. Somit ist zu jedem Zeitpunkt der Entwicklung exakt definiert, aus welchen Elementen sich eine bestimmte Version einer Anwendung bezüglich Anwendungskomponenten und Infrastruktur zusammensetzt.

Genauso wie jede Veränderung am Source Code einer Anwendung zu einer neuen Version führt, gilt dies nun auch für Veränderungen relevanter Konfigurationsinformationen, die die Infrastruktur betreffen.

Das Stichwort lautet „Infrastructure as Code“ – Infrastrukturaspekte werden auf „Augenhöhe“ mit Anwendungsquellcode behandelt, nicht mehr als nachgelagerte Notwendigkeit. Das bedeutet, dass Infrastrukturen nicht mehr unabhängig vom Entwicklungsprozess und statisch über Releases hinweg zur Verfügung gestellt werden, sondern als Teil eines Gesamt-Releases mit jeder neuen, zu testenden Version einer Anwendung.

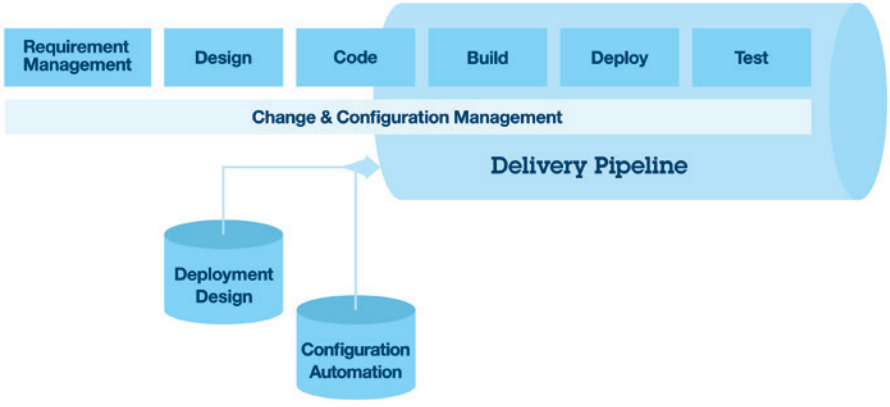

Abb. 3: Delivery Pipeline

DevOps & Cloud – eine sinnvolle Symbiose?

Welche Rolle spielt Cloud Computing nun im Kontext der DevOps „Delivery Pipeline“?

Die Nutzung von Cloud Computing und damit verbundenen „aaS“-Prinzipien ist keine zwangsweise Voraussetzung, um DevOps in die Praxis zu überführen, stellt jedoch einen sinnvollen und empfehlenswerten Aspekt bei der Umsetzung der „Delivery Pipeline“ dar.

Cloud Computing bedingt Standardisierung, Standardisierung erlaubt Automation. Mittels Cloud-Lösungen lassen sich Infrastrukturen und Plattformen nach einem definierten Modell dynamisch und reproduzierbar aufbauen und ebenso wieder abbauen. Grundlage dafür sind Modellbeschreibungen definierter Cloud Services, die sich von Zielumgebung zu Zielumgebung (Stages) unterscheiden können. Diese Cloud Services bilden somit einen Teil der für das Gesamt-Release der Anwendung notwendigen Gesamtdefinition.

Für die Infrastrukturbereitstellung bietet Cloud Computing die notwendige Agilität: Liegt ein neuer Build eines Anwendungs-Releases vor, so wird dieser automatisch deployed und getestet. Getestet wird auf dem Modell der Infrastruktur, welches als Teil des Gesamt-Releases definiert wurde.

Somit ergibt sich ein Zusammenspiel zwischen Anwendungs- und Infrastrukturdefinitionen:

Für jede Stage / Entwicklungs- und Teststufe erfolgt die Definition eines Infrastrukturmodells in Form definierter Cloud Services und gegebenenfalls zusätzlicher anwendungsnaher Konfigurationsinformationen, welches einem Anwendungsmodell zugeordnet wird. Die Kombination beider Informationsmodelle wird zentral in der Deployment Automation verwaltet.

Auf diese Art und Weise ist zu jeder Zeit bekannt, welcher kombinierte Stand aus anwendungs- und infrastrukturnaher Automation notwendig ist, um das Gesamt-Release vollständig automatisiert auszurollen.

Das Zusammenlegen beider „Informationsmodelle“ bedingt eine Zusammenarbeit von Rollen aus Entwicklung und Betrieb im frühen Entwicklungszyklus. Nur so kann ein einheitliches Modell sichergestellt werden, auf dem eine automatisierte „Delivery Pipeline“ sichergestellt werden kann.

Der definierte Stand der Infrastruktur kann in der Cloud auf Basis eines Cloud Service-Modells in sehr kurzer Zeit zur Verfügung gestellt werden. Ist das nicht exakt, was das DevOps-Prinzip des „Testens gegen produktionsartige Systeme“ fordert? Dem Entwickler geht es um Produktivität, er will seinen Code schnell und häufig deployen und testen – und genau hierbei hilft die Integration von Cloud-Technologie innerhalb der Continuous Delivery Pipeline.

Zusammengefasst bedeutet dies, dass die Nutzung von Cloud Services und Infrastrukturen einen idealen Gegenpol zu agilen Praktiken in der Anwendungsentwicklung darstellen. Die Anforderung nach kontinuierlichen Testverfahren und Deployments bedingt eine hochdynamische und standardisierte Methode, Infrastrukturen in kurzer Zeit bereitzustellen, zu konfigurieren und wieder abzubauen. Cloud-Infrastrukturen stellen ein geeignetes Mittel dar, sodass die Fragestellung nach einer sinnvollen Symbiose eindeutig mit einem „Ja“ beantwortet werden kann.

Bleibt die Frage: Was für eine Cloud wird für eine solche Aufgabenstellung benötigt?

Betrachten wir dazu die Trennung der Zuständigkeiten im Kontext des gesamten Release Stacks.

Während die Cloud für die Bereitstellung der Infrastruktur und Middleware-Komponenten zuständig ist, setzt das Anwendungs-Deployment auf der bereitgestellten Plattform auf. Das Anwendungs-Deployment übernimmt Konfigurationsaufgaben (z. B. Resource-Konfigurationen) sowie das Deployment der Artifakte. IBM bietet für letztere Zwecke die Lösung IBM UrbanCode Deploy (UCD) während auf Seiten der Cloud verschiedenste Angebote je nach Deployment-Modell (Private / Public Cloud) und unterstütztem Service-Modell (IaaS / PaaS / SaaS) zur Verfügung stehen.

Deployment Automation ist die Domäne, in der sich das Anwendungs-Deployment und die Infrastruktur-Provisionierung unterstützt durch Cloud-Technologie treffen. Anwendungen werden dabei entweder auf laufende Infrastrukturen (in der Cloud) deployed oder es wird implizit mit jedem Anwendungs-Deplyoment die Infrastruktur provisioniert, auf der das eigentliche Deployment aufsetzt. IBM UrbanCode Deploy kann dabei als „Spinne im Netz“ agieren und Cloud-Provisionierungsvorgänge sowie automatisiertes Testen im Zuge eines Gesamtprozesses einbinden.

Im Hinblick auf die Integration von Cloud-Lösungen wird dabei in IBM UrbanCode Deploy die Funktion eines Application Blueprints verwendet. Metdaten der Cloud Pattern-Definitionen (JSON) werden dabei in das Deployment Automation-Werkzeug importiert, sodass eine Übersicht der einzelnen Infrastrukturkomponenten und ihrer Eigenschaften angezeigt wird.

Den Infrastrukturkomponenten können nun die Komponenten der Anwendungen – die Deployment Units – zugeordnet werden, sodass ein Gesamtbild des Release Stacks entsteht. Auf diese Art und Weise lässt sich modellgetrieben eine gesamte Stage von der Infrastruktuktur aufwärts inklusive der fertigen Anwendung in wenigen Minuten in der Cloud zur Verfügung stellen.

Die Entscheidung, ob es sich um eine Private Cloud („On Premise“) oder um eine Public Cloud („Off Premise“) handelt, ist dabei für das Verfahren zweitrangig. Wichtiger ist die Kategorisierung der Cloud-Lösungen in die Art der Services, die bereitgestellt werden.

SaaS-Lösungen spielen in diesem Betrachtungskontext eine untergeordnete Rolle, da fertige Anwendungen aus der Cloud bezogen werden. Somit bleiben IaaS- und PaaS-Lösungen in der relevanten Betrachtung.

Je höher im Stack eine fertige Plattform zur Verfügung gestellt werden kann, umso einfacher werden die Aktivitäten für das gesamte Release-Rollout. Aus Sicht des Deployment Automation Toolings sollte die Cloud möglichst alles an Plattform Services bereitstellen, was notwendig ist, um relevante Konfigurationen vornehmen und die Artefakte deployen zu können.

Generell kann man also sagen, dass PaaS-Lösungen geeigneter sind als reine IaaS-Lösungen. Letztere lassen sich auch einsetzen, bedingen jedoch entsprechende Nachbereitungsaktivitäten, um aus isolierten Infrastrukturressourcen eine für das Anwendungs-Deployment fertige Plattform zur Verfügung zu stellen.

Grundsätzlich lässt sich jede Art von IaaS/PaaS Cloud in IBM UrbanCode Deploy integrieren, sofern sie Schnittstellen bereitstellt, die einen hohen Grad an Automation zulassen. Nachfolgend gehen wir in einer Case Study darauf ein, wie wir die globale IBM SoftLayer Cloud als IaaS-Lösung mit der Deployment Automation-

Lösung IBM UrbanCode Deploy integriert haben.

Case Study eines Pilotprojektes

An dieser Stelle wollen wir die beschriebene Theorie durch ein aktuelles Kundenprojekte aus der Automobilindustrie unterfüttern. Extrem aggressive Wachstumspläne auf globaler Basis sowie eine schnellere Time-to-Market-Strategie im Zuge mobiler Dienstleistungen bedingen insbesondere schnellere Entwicklungs- und häufigere Releasezyklen (im Minimum wöchentlich) auf der Seite der mobilen Backend Services, um die vom Fachbereich geforderten Innovationsspünge umsetzen zu können.

Wie die erforderliche Geschwindigkeit insbesondere in der dynamischen Bereitstellung von erforderlichen Testumgebungen auf Basis von externen Cloud Services auf I/PaaS im Kontext einer Continuous Delivery Pipeline bereitgestellt wurde, können sie unter http://ibm.co/SLCDUC1 und http://ibm.co/SLCDUC2 in voller Länge erfahren. Eine Zusammenfassung dieses Kundenpilotprojektes auf deutsch finden Sie unter: ftp://ftp.software.ibm.com/software/emea/de/rational/Pilotprojekt.pdf.

Herausforderung des DevOps-Konzeptes war – neben der üblichen organisatorischen Trennung von Anwendungsentwicklung und Betrieb – die Splittung der beiden Bereiche über Unternehemensgrenzen hinweg. Während die Anwendungsentwicklung in der Verantwortung des Kunden verbleibt, liegt die Betriebsverantwortung in den Händen des Cloud Providers, der IBM.

Zusammenfassung

Die Umsetzung von DevOps-Prinzipien bedingt eine bereichsübergreifende Zusammenarbeit zwischen Entwicklungs- und Betriebsorganisation und verfolgt das Ziel einer Kostensenkung und Risikominimierung, indem Fachanforderungen schneller in die Produktion gebracht werden können. Der Übergang („Staging“) in die Produktion erfolgt reproduzierbar, automatisiert und auditierbar in kürzeren Test-Zyklen mit dem Ergebnis einer letztendlich höheren Softwarequalität.

DevOps bedingen eine Ende-zu-Ende-Automationsverkettung des Gesamtprozesses, in dem verschiedene Teilaspekte verknüpft werden. Dies wird auch als „Delivery Pipeline“ bezeichnet und erstreckt sich u. a. über die Stufen des Build-, Deploy- und Testmanagements. Das Deployment Management kann dabei idealerweise durch Cloud-Technologie unterstützt werden, um die Agilität in der Infrastrukturbereitstellung zu gewährleisten.

Die Kompetenz der IBM sowie der Mehrwert der IBM-Lösung liegt in der gesamtheitlichen Betrachtung und Ab-deckung der „Delivery Pipeline“. Die IBM bietet ein durchgängiges aber gleichzeitig auch offenes Lösungskonzept zur Einbindung vorhandener Lösungselemente an. ■

Weitere Informationen

[IBM DO] IBM DevOps: http://www.ibm.com/ibm/devops/us/en/

[IBM DOD] DevOps for Dummies (deutsche Version): https://www14.software.ibm.com/webapp/iwm/web/signup.do?source=swg-DE_WEB-ORG_Rational&S_PKG=ov18162

[IBM RP] IBM Rational Proof of Technology Workshops: http://www.ibm.com/de/events/pot-rational/