



PaaS-Marktübersicht: Java aus der Cloud

Weltweit programmieren etwa 9 Millionen Software-Entwickler in Java – ein gigantischer Kundenstamm. Aber was taugen die aktuellen Java-Umgebungen in der Cloud?

VON EBERHARD WOLFF

Cloud Computing verspricht, die IT-Landschaft grundlegend zu verändern. Denn der Ansatz bietet on-demand die Nutzung von IT-Diensten über Selbstbedienungsportale an. Sie stehen also je nach Bedarf zur Verfügung und jeder Anwender kann sie selbst buchen. Typischerweise rufen Kunden die Dienste übers Internet ab. So kann man bei IaaS-Angeboten (Infrastructure as a Service) wie Amazon EC2 virtuelle Rechner anlegen und die Instanzen sind innert weniger Minuten einsatzbereit.

Die Vorteile des Cloud-Ansatzes bestehen einerseits in einem flexibleren Kostenmodell. Nur die jeweils aktuell benötigten Ressourcen verursachen Kosten. Weil bei Bedarf schnell zusätzliche Ressourcen zur Verfügung stehen, ist der Umgang mit Lastspitzen sehr effizient möglich. Ausserdem liegen die Investitionskosten für eine Cloud-Lösung sehr niedrig, weil die Anschaffung eigener Server entfällt. Und Cloud-Lösungen müssen nicht langwierige organisatorische Prozesse durchlaufen, sondern können durch Self-Service-Portale genutzt werden.

Kunden nehmen dadurch schneller Anwendungen in Betrieb, was in Konsequenz zu einer besseren Time-to-Market führt.

Eberhard Wolff (Twitter: @ewolff) arbeitet als Architecture & Technology Manager für die adesso AG. Seine Schwerpunkte liegen auf Cloud-Technologien, Enterprise Java u. a. mit Spring und Software-Architekturen → www.adesso.de

GRETCHENFRAGE: IAAS ODER PAAS?

Eine IaaS-Cloud wie Amazon EC2 offeriert nur «nackte» Rechner und Storage. Soll darauf eine selbst geschriebene Anwendung laufen, muss vorab eine Infrastruktur – zum Beispiel ein Application Server – installiert werden. Als Alternative dazu bieten sich PaaS-Clouds (Platform as a Service) an. Auf einer solchen Plattform ist die für den Betrieb von Anwendungen nötige Infrastruktur bereits vorhanden. Ergo können Kunden ihre Anwendungen direkt in der Cloud installieren. Ausserdem enthält eine PaaS einschlägige Lösungen für den Betrieb und das Monitoring der installierten Anwendungen. Das Skalieren bei Lastspitzen erfolgt automatisch.

PaaS-Lösungen bieten – oft proprietäre – Services mit fertigen Funktionalitäten, die

Programmierern bei der Entwicklung von Applikationen helfen. Beispielsweise enthält die Amazon-Cloud-Plattform einen Service, über den der Kunde per Kreditkarte bezahlen kann. Durch die Nutzung solcher Dienste begibt man sich allerdings in Abhängigkeit von «seiner» PaaS, da sich Anwendungen nicht so einfach auf alternative Plattformen, die solche Services nicht anbieten, migrieren lassen.

Im Markt der Enterprise-Anwendungen stellen Java-Plattformen die wichtigste Ablaufumgebung dar. Es ist noch nicht lange her, da konnte man PaaS-Plattformen für Java an ein, zwei Fingern abzählen. Aber in den letzten Monaten ist das Angebot gewachsen.

WER IST DER BESTE ANBIETER?

Auf dem Java-PaaS-Markt haben sich mittlerweile einige grosse Player positioniert. Dabei fallen die Erfolgskonzepte sehr unterschiedlich aus.

■ **Google** gilt als Anbieter der ersten Stunde, aber die Plattform unterstützt typische Java-Technologien nur sehr eingeschränkt, sodass heute wohl kaum jemand darauf zurückgreifen wird. Es sei denn, proprietäre Dienste wie beispielsweise Googles User-Verwaltung spielen bei der Entscheidung eine Rolle.

■ **Amazon Beanstalk** fügt seiner populären Public-Cloud-Plattform ein Java-PaaS hinzu, sodass die erwiesenermassen leistungsfähige Amazon-Infrastruktur nun wesentlich einfacher auch von Java-Applikationen genutzt werden kann.

■ Das Start-up **CloudBees** verfolgt mit seinen auf Entwickler abgestimmten Services einen ganz anderen Ansatz, kann aber als Ablaufumgebung – zumindest in der zurzeit kostenlosen Version – kaum überzeugen. Hier darf man auf die künftigen Angebote gespannt sein.

■ Schliesslich sticht mit **VMwares Cloud Foundry** eine Open-Source-Lösung ins Auge, die ein interessantes Konzept verfolgt und dabei gleichzeitig offen und flexibel bleibt.

■ Und schliesslich hat Salesforce mit **Heroku** auch einer der PaaS-Pioniere eine Java-Story, die ausserdem durch den Salesforce.com-Bezug sehr viel Potenzial hat. Es bleibt also spannend!

DER KLASSIKER: GOOGLE APP ENGINE

Google hat bereits seit 2008 ein Java-PaaS am Markt: die Google App Engine (GAE). Neben Java unterstützt die GAE auch Python. Die Kosten für den Betrieb von Anwendungen hängen von drei Parametern ab: Bandbreite, Storage und CPU-Zeit. Zum Testen gibt es ein freies Kontingent. Google offeriert ausserdem eine interessante Finanzierungsvariante: Kunden können ein fixes Budget für ihre Anwendung festlegen, um so die Kosten zu deckeln. Für Applikationen, die lediglich innerhalb der Firma verwendet werden, gibt es die Google App Engine for Business. Dabei erfolgt die Abrechnung pro Anwendung und Benutzer, ausserdem unterscheiden sich die Service Level Agreements (SLA) der Standard- und der Business-Version.

GAE treibt den PaaS-Ansatz recht weit: Der Kunde hat keinen Zugriff auf das zugrundeliegende Betriebssystem. Anwendungen können auf einem oder mehreren Servern laufen, ohne dass der Entwickler davon Kenntnis hat. Die Lastverteilung und Skalierung der Anwendung sind transparent. Um dies zu ermöglichen, verfolgt GAE ein sehr eingeschränktes Programmiermodell, das die Funktionalität von Java nicht voll ausschöpft. Die auf GAE nutzbaren Java-Klassen stehen auf einer White List. So ist es beispielsweise nicht möglich, neue Threads zu starten; der Zugriff auf das File-System ist nicht erlaubt und ein Request darf nicht mehr als 30 Sekunden Bearbeitungszeit erfordern. Durch diese Einschränkungen ist die Nutzung einiger Java-Frameworks gar nicht möglich oder nur sehr aufwendig zu realisieren. Eine Liste informiert über Frameworks, die unter GAE laufen.

Als Datenspeicher bietet Google eine NoSQL-Datenbank an. Eine relationale Datenbank ist zwar angekündigt, soll dann aber nur im Rahmen von Google App Engine for Business nutzbar sein. In der Folge muss die Persistenzarchitektur einer Anwendung, die auf GAE laufen soll, entsprechend angepasst werden. Die Portierung auf eine andere Infrastruktur wird dadurch natürlich erschwert.

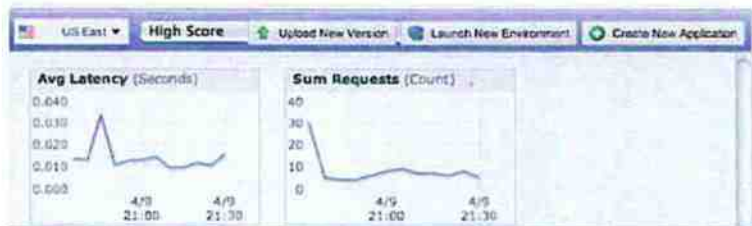
Das GAE-Angebot beinhaltet zahlreiche zusätzliche Services wie einen Cache, XMPP-basiertes Instant Messaging, Bildverarbeitung oder die Speicherung von Blobs. Ausserdem können GAE-Entwickler via Web Services APIs auf andere Google-Services zugreifen. Auch

eine Integration in die Google-User-Verwaltung fehlt nicht. Für Programmierer stehen darüber hinaus Plug-Ins für Eclipse zur Verfügung, sodass ein Deployment von Eclipse direkt auf GAE keine Probleme bereitet. Mithilfe des GAE SDKs (Software Development Kit) lassen sich Anwendungen für Testzwecke auch lokal installieren. Ein einfaches Monitoring hilft, die Performance und den Ressourcenverbrauch der Anwendungen im Auge zu behalten.

Google App Engine

- ⊕ Automatische Skalierung der Anwendung
- ⊕ Lange am Markt, deshalb bewährt
- ⊖ Wesentliche Einschränkungen gegenüber dem Standard-Java-Programmiermodell
- ⊖ Relationale Datenbank angekündigt, aber zurzeit nicht verfügbar
- ⊖ Aus den Einschränkungen ergeben sich Abhängigkeiten (Vendor's Lock-In)
- ⊖ Keine Eingriffs- oder Tuning-Möglichkeiten auf Ebene des Webservers oder Betriebssystems

DER VORREITER: AMAZON BEANSTALK



Monitoring einer Anwendung mit Amazon Beanstalk

Amazon ist schon seit 2006 mit verschiedenen Services in der Cloud präsent. Dazu zählen virtuelle Rechner (Elastic Compute Cloud, EC2), Storage (Simple Storage Service, S3), aber auch komplexere Dienste wie schlüsselfertige MySQL-kompatible Datenbanken (Relational Database Service, RDS).

Mit Beanstalk (Bohnenranke) stellt Amazon ein Angebot vor, das die Installation und den Betrieb von Java-Anwendungen deutlich vereinfacht. Die Anwendungen werden dabei auf EC2 installiert. Auf dem virtuellen Rechner läuft Linux, ergänzt durch eine OpenJDK-Java-Implementierung und den Java-Webserver Tomcat. Um Lastspitzen abzufedern, verwendet Amazon einen Load Balancer sowie seinen Auto Scaling Service. Er startet bei hoher Last automatisch zusätzliche EC2-Instanzen. Als Auslöser für den automatischen Start kommen unterschiedliche Indikatoren ins Spiel, etwa die überlange Latenzzeit eines Requests oder die CPU-Last auf einzelnen Rechnern. Die Skalierung, also die Abfederung von Überlasten, ist jedoch recht grobgranular konzipiert: Erreicht ein Rechner seine Lastgrenze, muss gleich eine neue Instanz gestartet werden, während sich beispielsweise bei GAE mehrere Kunden eine Rechnerinstanz teilen können.

Die unterschiedlichen Versionen einer Anwendung lassen sich auf S3 speichern, sodass sie bei Bedarf wiederhergestellt werden können. Ein Eclipse-Plug-In macht den Zugriff auf einen Beanstalk-Server ähnlich einfach wie den Zugriff auf einen lokalen Tomcat. Ein Kommandozeilenwerkzeug ist ebenfalls enthalten.

Ausserdem bietet Beanstalk Raum für eigene Anforderungen: So lassen sich etwa die

Parameter des Tomcat auch auf einer laufenden Maschine ändern, selbst Rechner-Images können modifiziert werden.

Der Einstieg in die Plattform fällt leicht: Beanstalk stellt beim ersten Login die Installation einer Beispielanwendung zur Auswahl, und innerhalb weniger Minuten läuft das System. Als Ablage für die Daten dienen die Relational Data Services. Einem klassischen Enterprise-Java-Programmiermodell steht daher nichts im Wege. Entwickler können aber auch einfachere NoSQL-Lösungen wie Amazon SimpleDB verwenden. Zudem offerieren Drittfirmen auf EC2 mittlerweile zahlreiche, hilfreiche Zusatzdienste.

Amazon Beanstalk selbst ist kostenlos. Die verwendeten EC2-Instanzen und weitere Amazon-Dienste wie S3 belasten allerdings das Budget. Neukunden dürfen ein grosszügig bemessenes Service-Kontingent für die Dauer eines Jahres kostenlos nutzen, bevor sie in den produktiven Einsatz gehen.

Das Angebot steckt noch in der Beta-Phase. Da Beanstalk lediglich Dienste aus dem Amazon-Cloud-Angebot kombiniert und deren Nutzung vereinfacht, hält sich das Risiko für Beta-Kunden aber in Grenzen. Beanstalk steht zurzeit nicht in allen Amazon-Rechenzentren, sondern nur in der US-East-Region zur Verfügung. →

Amazon Beanstalk

- ⊕ Bewährte Cloud-Infrastruktur als Basis
- ⊕ Übliches Enterprise-Java-Programmiermodell (Tomcat/relationale Datenbank)
- ⊕ Daher geringe Lock-In-Gefahr auf Code-Ebene
- ⊕ Zahlreiche weitere Services (RDS, SimpleDB, Payment etc.)
- ⊖ Zurzeit noch in Beta und nur in US-East verfügbar
- ⊖ Grobgranular: Skalierung nur durch zusätzliche Rechner

DER EINSTEIGER: CLOUDBEES

Owner: Created Mar 10, 2011 Status hibernate ID ewolff/adessorodemo
Location: http://bees.chun.chun.ec2.amazonaws.com/

Development Operations Logs Configuration

Cloud Environment

Bees Hive
Free tools to run your applications

Memory Usage
Select the maximum amount of server memory the application can use
 256 MB Memory
Use a 256 MB slice of the server memory

Redundancy and Scale

Das Start-up CloudBees hat sich auf den Java-Cloud-Markt spezialisiert und offeriert unter dem Branding DEV@Cloud einige attraktive Dienste. Dazu zählt etwa der «Continuous Integration Server», der sich den Source-Code nach jeder Änderung aus der Versionskontrolle holt, ihn kompiliert und automatisierte Tests durchführt. Dies passt sehr gut zum Beschaffungsmodell Cloud: Er benötigt zwar viele Ressourcen, aber nur dann, wenn Entwickler tatsächlich den Code modifizieren, und auch dann nur für kurze Zeit. Ausserdem sind die Umgebungen für Continuous Integration standardisiert, was die Auslagerung in die Cloud erleichtert. Der neu kompilierte Code kann in einem von CloudBees angebotenen Maven Repository abgelegt und von anderen Projekten weiterverwendet werden. DEV@Cloud bietet einen kostenlosen Service mit reduzierter Infrastruktur und verschiedene kostenpflichtige Dienste.

Als Ablaufumgebung bietet CloudBees die zurzeit noch kostenlose RUN@Cloud an. Dahinter steckt im Wesentlichen ein Tomcat-Webserver. Was nichts kostet, bietet auch wenig Komfort, dies bestätigt auch RUN@Cloud. Die Ablaufumgebung lässt sich praktisch nicht anpassen und bietet auch kaum Skalierungsmöglichkeiten. Der Nutzer hat lediglich die Option, zwischen einer und mehreren Instanzen zu wählen. Ein einfaches Monitoring und ein Kommandozeilenwerkzeug ist im Dienst inbegriffen.

Ein grosser Vorteil des CloudBees-Ansatzes: Mehrere Kunden können sich einen virtuellen Rechner teilen, mehrere Tomcat-Instanzen laufen auf einem (virtuellen) Rechner. Dadurch ist die Lösung kosteneffizienter als Beanstalk.

Als Datenbank spendiert CloudBees MySQL-Instanzen. Auch hier ist der Dienst einfach ge-

strickt. Der Zugriff auf eine MySQL-Instanz ist ohne Einschränkung aus dem gesamten Internet erlaubt und es besteht keine Möglichkeit, Backups anzulegen. Die CloudBees-Infrastruktur läuft jedoch auf Amazon EC2, sodass sich die Kombination mit Diensten wie RDS anbietet.

Man sollte bei der Bewertung von CloudBees berücksichtigen, dass RUN@Cloud zurzeit kostenlos ist. Wenn der bezahlpflichtige Dienst zur Verfügung steht, hat CloudBees sicherlich viele der angesprochenen Probleme gelöst.

CloudBees

- ⊕ Attraktiv für Entwickler wegen Continuous Integration und Repository Server
- ⊕ RUN@Cloud kostenlos
- ⊕ EC2-Infrastruktur, daher leicht mit Amazon-Angeboten kombinierbar

- ⊖ RUN@Cloud zurzeit in Beta
- ⊖ Kein echtes Skalierungskonzept
- ⊖ Keine SLAs
- ⊖ Datenbanklösung eher rudimentär
- ⊖ Tomcat-Installation kaum anpassbar