

# Oracle Fusion Middleware und WebLogic Server in der Cloud

Autor: Frank Munz, munz & more

*Clouds sind da. Sie werden täglich genutzt, ermöglichen neue Geschäftsmodelle und bieten weitaus mehr als nur virtualisierte Rechner. Dieser Artikel gibt Ihnen einen Überblick, wie einfach es sein kann, Ihre Middleware in der Cloud zu betreiben, welche Cloud-Dienste als Alternativen zu Middleware Features in Betracht kommen und wie diese die Architektur beeinflussen.*

## **Cloud-Definition statt Vaporware**

Lassen Sie uns kurz definieren, was Cloud Computing ist. Sie haben sicher schon den einen oder anderen Artikel gelesen, in dem Google Mail oder eine andere über das Internet bereitgestellte Webanwendung als Cloud Computing verkauft wird. Gar nicht so selten müssen Clouds auch als Synonym für etwas herhalten, was wir vor 15 Jahren schlicht „online“ genannt hätten. Cloud Computing, das den Namen verdient, zeichnet sich hingegen durch die folgenden drei Kriterien aus:

- 1) Ressourcen werden als Dienste genutzt. Zu diesen Ressourcen gehören beispielsweise Server, hochverfügbare Dateisysteme, Load Balancer, aber ebenso Dienste wie ein Queueing System, eine Datenbank oder eine WebLogic-Umgebung. Dienst bedeutet, dass es eine technische Schnittstelle gibt, über die eine Ressource angefordert werden kann. Üblicherweise setzt eine webbasierte Managementkonsole auf dieser Programmschnittstelle auf.
- 2) Clouds bieten schnelle Elastizität, dadurch können Ressourcen in der Cloud in kurzer Zeit vergrößert oder verkleinert werden.
- 3) Sie bezahlen nur das, was sie wirklich nutzen, bzw. es wird das verrechnet, was sie nutzen (Pay-on-Demand). Dass Anbieter wie Amazon neuen Kunden eine Micro-Instanz für ein Jahr umsonst bereitstellen, ist hierzu kein Widerspruch – Sie können sich nämlich sicher sein, dass Ihnen alle zusätzlichen Ressourcen außerhalb des freien Einstiegsangebots stundengenau in Rechnung gestellt werden.

## **Werner Vogels und Larry Ellison: Pay-on-Demand oder Hoheit über die Server?**

Werner Vogels, der CTO des größten öffentlichen Cloud-Anbieters Amazon Web Services (AWS), erwähnt immer wieder gerne, dass eine Cloud, bei der zunächst eine Investition in Hardware notwendig ist, in Wirklichkeit gar keine Cloud ist. Larry Ellison kontert darauf, dass Sie doch sicher nicht Ihre Kronjuwelen (= ihre Daten) irgendwo ins Internet stellen wollen.

Für die weiteren Betrachtungen sehen wir das eher pragmatisch: In der obigen Definition steht nichts davon, dass die Beschaffung von Hardware im Widerspruch zu Cloud Computing steht. Natürlich sind private Clouds auch Clouds. Warum sollte auf ein und dieselbe Technologie, wenn sie auf der einen Seite einer Firmenfirewall steht, der Begriff Cloud zutreffen, wenn sie auf der andere Seite steht aber nicht?

Vermutlich werden Sie nicht als erstes Projekt Ihre Finanzbuchhaltung in eine öffentliche Cloud migrieren wollen. Davon abgesehen sind Ihre Daten zunächst einmal in einer öffentlichen Cloud so sicher wie auf jedem anderem Server, auf den aus dem Internet zugegriffen wird. Vielleicht möchten Sie für Ihr aktuelles Oracle WebLogic Projekt möglichst schnell Ihren ersten Scrum Sprint unter Last testen, bevor dann, meistens 3 Monate später, Ihre bestellten Server für die eigentliche Lasttestumgebung eintreffen? Auch hierzu eignet sich eine öffentliche Cloud.

Eine öffentliche Cloud ist aber nicht nur auf das Testen beschränkt. Sie ist vielmehr ein vollständig programmierbares, virtuelles Rechenzentrum, bei dem nur die tatsächlich genutzten Ressourcen bezahlt werden. Daraus ergeben sich neue Geschäftsmodelle für Unternehmen, die ohne diese Technologie nicht existieren würden – aber dank der Cloud gutes Geld verdienen. Durch die Elastizität der Cloud und das automatische Skalieren werden nur die tatsächlich benötigten Ressourcen angefordert. Durch das Pay-on-Demand-Prinzip ergibt sich daraus ein einfaches Kostenmodell.

Die Vorteile, solche Cloud-Anwendungen auf der Basis von Oracle Fusion Middleware (OFM) zu entwickeln, liegen auf der Hand: Sofern Sie OFM auf einer beliebigen Cloud betreiben können, läuft dort auch Ihre Anwendung. Aber passt

Fusion Middleware mit der Cloud überhaupt zusammen und worauf sollten Sie dabei achten?

## Oracle Middleware in öffentlichen Clouds am Beispiel von AWS

Als konkretes Beispiel soll zunächst Amazon Web Services (AWS), die größte und umfangreichste öffentliche Cloud dienen. Abbildung 1 zeigt eine mögliche Referenzarchitektur für den Einsatz von Oracle Middleware in der Cloud. Diese Referenzarchitektur enthält exemplarisch die wichtigsten Cloud-Dienste, die im Folgenden näher erläutert werden.

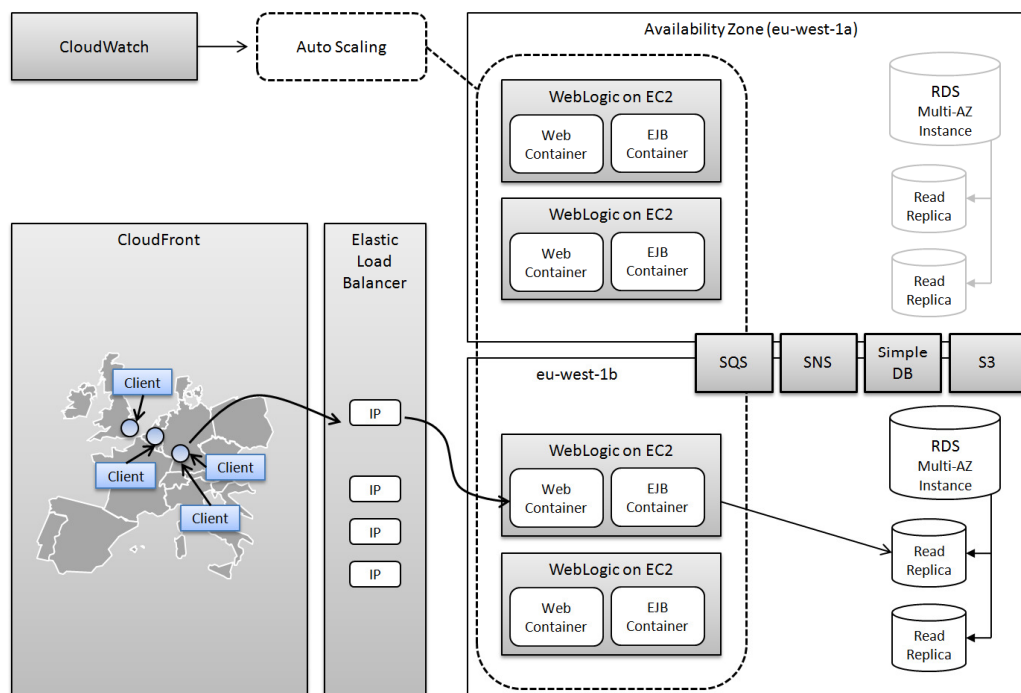


Abbildung 1: Referenzarchitektur Oracle WebLogic in der AWS Cloud Region Europa

Für den Einsatz von Oracle Fusion Middleware bei AWS gibt es eine Reihe von Ansätzen:

- 1) Sie starten mit einem Amazon Machine Image (AMI), das ein von OFM unterstütztes Betriebssystem enthält. Diese AMIs werden von Amazon selbst erstellt und enthalten nur das Betriebssystem. Das Image wird über die webbasierte AWS Managementkonsole gestartet. Mit dem laufenden Image installieren Sie OFM und richten die notwendigen Domänen ein.

Vorteil: Dieser Ansatz bietet Ihnen die größtmögliche Flexibilität. Die Auswahl des zugrunde liegenden Betriebssystems und die Version der installierten OFM-Komponenten liegen in Ihrer Hand. Am Anfang ist es am einfachsten, das Image komplett zu konfigurieren und ein neues Abbild zu erstellen, das die vollständig konfigurierte Software enthält.

Wenn Sie später unterschiedliche Images benötigen, können Sie zu einem skriptbasierten Ansatz mit wiederverwendbaren Konfigurationsmodulen wechseln, um den Verwaltungsaufwand zu minimieren. Der Einsatz eines professionellen Cloud Managementtools wie RightScale, das eine skriptbasierte Konfiguration unterstützt und eine Skriptbibliothek (allerdings nur für Open-Source-Komponenten) und Multicloud-Unterstützung bereits mitbringt, ist in größeren Umgebungen vorteilhaft [CCBuch, RightS].

Nachteil: Diese Vorgehensweise ist mit größerem Aufwand verbunden, obwohl sie wegen der wiederverwendbaren Skripten längerfristig den Wartungsaufwand minimiert.

- 2) Oracle selbst bietet eine Reihe von AWS Images an, die vorinstallierte Oracle Produkte wie WebLogic Server enthalten, um Ihnen den Einstieg zu erleichtern [OraAMI].

Vorteil: Sie sparen sich die Installation des Oracle Produkts und können sich darauf verlassen, dass auch alle Voraussetzungen, wie Betriebssystem Patches usw. erfüllt sind.

Nachteil: Es ist zu beachten, dass nicht alle Oracle Produkte als AMIs angeboten werden und die verfügbaren AMIs üblicherweise nur die neueste Version der Oracle Produkte enthalten. Auch ist nicht jedes in einem AMI angebotene Oracle Produkt in allen Amazon Regionen verfügbar. Sie dürfen daher nicht blind darauf vertrauen, dass diese Images immer verfügbar sind und sollten sich daher eine Kopie von dem von Ihnen benötigten Image anlegen.

- 3) Seit Anfang 2011 ist es eingeschränkt möglich, auch Templates von Oracles eigener Virtualisierungsplattform OVM, so genannte Oracle VM Templates, in der Amazon Cloud zu starten. In allen AMIs ist ein 64-bit Oracle Enterprise Linux installiert [OraEC2].

Nachteil: Diese Templates liegen als AWS AMIs vor, momentan allerdings nur

für die AWS Region US-East. Die Auswahl der möglichen AWS Elastic Compute Cloud Instanzen (EC2) ist ebenfalls eingeschränkt.

## **Technische Besonderheiten in der Cloud**

Da die Server in der Cloud virtualisiert sind, müssen eine Reihe technischer Besonderheiten beachtet werden.

### *WebLogic Clustering*

Beim Clustering von WebLogic ist darauf zu achten, dass die meisten öffentlichen Clouds kein Multicast zulassen (dazu gehört neben Amazon zum Beispiel auch der zweitgrößte Anbieter Rackspace) [RackS]. Ein WebLogic Cluster muss daher als Unicast Cluster definiert werden, da sonst die notwendigen Cluster Heartbeats und die Verteilung des JNDI-Baums im Cluster nicht funktionieren. Unicast bildet die 1:n Kommunikation des IP Multicast auf n-mal 1:1 TCP-Verbindungen ab. Bei der aktuellen WebLogic Version 11g ist Unicast die Standardeinstellung.

### *Verfügbarkeit von JMS*

Java Message Service (JMS) ist ein Dienst, der an genau einen WebLogic Server gebunden ist. Im Gegensatz etwa zu einer Webanwendung, die auf mehrere WebLogic Managed Server deployed werden kann, kann ein JMS-Server nur genau einem WebLogic Server zugewiesen werden. Das Einrichten eines WebLogic Clusters erhöht die Verfügbarkeit eines JMS-Server nicht. Um die Verfügbarkeit von JMS zu garantieren, muss bei WebLogic entweder Whole-Server oder Service Migration konfiguriert werden.

### *Whole Server Migration*

Whole Server Migration (WSM) ist ein gängiges Mittel, um die Verfügbarkeit eines JMS-Servers zu erhöhen. WSM startet mithilfe des Nodemanagers einen ausgefallenen Managed Server auf einer anderen Maschine erneut.

Für die Funktion der WSM müssen eine Reihe von Voraussetzungen erfüllt werden, unter anderem müssen auf der Zielmaschine neben der WebLogic-Installation auch alle notwendigen Dateien der Domäne (am einfachsten also das komplette Domänenverzeichnis) und der Nodemanager verfügbar sein.

Vorteil: Der Vorteil der WSM besteht darin, dass der komplette Managed Server mit den gesamten Konfigurationen wie JDBC, Workmanager und JMS auf einer anderen Maschine gestartet wird und sich dadurch die Konfigurationen der einzelnen Managed Server in einer Domäne nicht verändern.

WSM in der Cloud: Bei der WSM weist der Nodemanager die Floating IP der ausgefallenen Ausgangsmaschine einer neuen Zielmaschine zu. In öffentlichen Clouds wie Amazon oder Rackspace funktioniert dieser Mechanismus allerdings nicht. Bei AWS gibt es keine Floating IPs. Bei Rackspace gibt es zwar die Möglichkeit, zusätzliche IPs gegen eine geringe Gebühr zu beantragen, allerdings können diese nur vom Betriebspersonal der Cloud einer anderen Maschine zugewiesen werden. Kurz gesagt: Die WSM funktioniert in öffentlichen Clouds nicht.

### *Service Migration*

Bei der Service Migration wird JMS als Service betrachtet. Dieser Service wird auch ohne Nodemanager auf einen anderen Managed Server verschoben, sollte der Ausgangsserver ausfallen. Im Unterschied zur WSM wird allerdings kein neuer WebLogic Server gestartet.

Nachteil: Die Konfiguration des Zielservers in der Domäne ändert sich. Einem bestehenden Server wird der ausgefallene JMS-Dienst zugeordnet, daher muss sichergestellt sein, dass es nicht zu Last- oder Ressourcenproblemen kommt.

Service Migration in der Cloud: Es gibt keine Einschränkung für die Service Migration in Clouds. Die Service Migration kann in Clouds als Alternative für die WSM verwendet werden.

Beim Beachten der oben beschriebenen Eigenheiten ist der Betrieb von OFM basierten Anwendungen ohne Probleme in der Cloud möglich.

## **Echtes Cloud Computing mit Middleware?**

Lassen Sie uns nochmals einen kurzen Blick auf die Kriterien einer Cloud am Anfang des Artikels werfen, um zu sehen, wie diese nun tatsächlich implementiert werden können.

*Nutzen von Ressourcen als Dienste:*

Im Rahmen des hier vorgestellten Artikels greifen wir auf die Infrastrukturdienste von AWS über die vom Cloudanbieter bereitgestellten Schnittstellen zu. Wir nutzen WebLogic in der Cloud – durch weitere Automatisierung ist es auch, möglich eine dedizierte WebLogic Cloud aufzubauen.

#### *Elastizität:*

Das schnelle horizontale Skalieren von Anwendungen lässt sich durch das Autoscaling von AWS erreichen. Hierzu ist es notwendig, eine Launch-Konfiguration festzulegen, die definiert, mit welchen Parametern ein neues Image beim Überschreiten einer Lastschwelle gestartet wird. Die maximale und minimale Größe der Umgebung wird durch eine Autoscaling-Gruppe definiert und lastabhängige Schwellenwerte, die das Vergrößern oder Verkleinern der Gruppe auslösen, werden durch Autoscaling Trigger definiert.

Seitens der Middleware muss zusätzlich sichergestellt werden, dass die neu gestarteten Images auch Teil einer OFM-Domäne sind, was aber durch ein Startskript mit dem WebLogic Scripting Tool (WLST) erreicht werden kann, das die Instanz beim Hochfahren des AMI in die Domäne einfügt [CCBuch].

#### *Pay-on-Demand:*

Sämtliche Dienste der Amazon Cloud werden stundenweise in Rechnung gestellt, sofern Ihre Nutzung über einem bei einigen Ressourcen vorhandenen Gratisvolumen liegt. Das Oracle Lizenzmodell erlaubt die Mitnahme einer vorhandenen OFM-Lizenz in die Amazon Cloud (oder den Kauf einer neuen Lizenz speziell für die Cloud).

Aufgrund des Lizenzmodells von Oracle ist es momentan aber nicht möglich, WebLogic beispielsweise nur für 21 Tage zu lizenzieren und im Rahmen einer kurzen Werbekampagne auch nur für diesen Zeitraum zu bezahlen.

Stand heute ermöglicht also die technische Infrastruktur des Cloudanbieters eine exakte, stundenweise Pay-on-Demand-Abrechnung für alle Ressourcen – das Oracle Lizenzmodell allerdings nicht. Eine Ausnahme hierzu, bzw. einen ersten Schritt in die richtige Richtung, stellt der unten beschriebene relationale Datenbankdienst (RDS) von Amazon dar. RDS garantiert die stundenweise Bereitstellung und Abrechnung einer MySQL-Instanz in der AWS Umgebung. Die alternative Bereitstellung einer

Oracle Datenbankinstanz in der Cloud, ebenfalls mit stundenweiser Abrechnung, ist angekündigt [OraRDS].

## **Middleware: Architektur und Cloud-Dienste**

Der Ausfall des Amazon Rechenzentrums im April 2011 hat gezeigt, dass es zwar aus Sicht der Anwendung keinen wesentlichen Unterschied macht, ob diese in der Cloud läuft oder nicht, dass aber auf der Ebene der Systemarchitektur die Maxime „Plan for Failure“ berücksichtigt werden muss.

In der Cloud stehen Dienste zur Verfügung, deren Einsatz die OFM-Architektur beeinflusst. In Abbildung 1 wird eine Architektur mit zwei Availability-Zonen gezeigt. Availability-Zonen sind voneinander unabhängige Rechenzentren innerhalb einer Region. Außerhalb der Cloud ist der Betrieb mehrerer, redundanter Rechenzentren mit immensen Kosten verbunden und für kleine und mittlere Unternehmen nicht bezahlbar. In der AWS Cloud können Sie selbst entscheiden und beim Anfordern einer neuen Serverinstanz die Region (USA Ost oder West, Europa, Singapur, Japan) und innerhalb der Region jeweils eine von mehreren Availability-Zonen angeben. Availability-Zonen stellen also die Grundlage für hochverfügbare Architekturen in der Cloud dar.

**Simple Queueing Service (SQS)** ist ein Cloud-Dienst für die zuverlässige Übermittlung von Nachrichten und kann als Alternative zu JMS Queues betrachtet werden. Der Dienst wird von Amazon verwaltet und ist in jeder Region der AWS Cloud verfügbar. Für die Verwendung von SQS muss keine EC2-Instanz gestartet werden. Die maximal 65 kB großen Nachrichten werden hochverfügbar persistiert. SQS Queues können beliebig viele Nachrichten speichern und garantieren eine At-Least-Once-Semantik.

Ähnlich wie SQS kann der AWS Dienst **Simple Notification Service (SNS)** als Alternative zu JMS Topics betrachtet werden. Im Gegensatz zu SQS besteht jedoch bereits eine Integration von SNS in die webbasierte AWS Managementkonsole. Die maximal 8 kB großen SNS Notifications können über unterschiedliche Transportprotokolle wie http, email (auch im JSON-Format) und SQS zugestellt werden. SNS garantiert eine Best-Effort-Semantik.

Im Vergleich zu WebLogic JMS Queues bietet SQS/SNS allerdings weniger Features: kein Auto-Acknowledge, keine Quotas, keine Flusskontrolle (siehe Tabelle 1). Durch die inhärente Verfügbarkeit von SQS/SNS spart man jedoch die oben beschriebene aufwändige Konfiguration der Verfügbarkeit des JMS-Servers sowie das aktive Management des Speicherplatzes für die Persistierung der Nachrichten.

	<b>SQS Queues</b>	<b>WebLogic JMS Queues</b>
<b>Max. Queue-Länge</b>	Unbegrenzt	Bestimmt durch JVM Heap und Persistent Store
<b>Beste Quality of Service</b>	At-Least-Once	Exactly-Once mit Transaktionen
<b>Konfigurierbare Retries</b>	Nein	Ja
<b>Persistenz</b>	Immer	Optional
<b>Skalierbarkeit</b>	Inhärent	Distributed Queues
<b>Verfügbarkeit</b>	Inhärent	Whole-Server oder JMS Service Migration
<b>Nachrichtenreihenfolge</b>	Nicht garantiert	Kann erzwungen werden
<b>Konfigurierbare Quotas</b>	Nein	Ja
<b>Flusskontrolle</b>	Nein	Ja
<b>Auto-Acknowledge</b>	Nein	Ja
<b>Time to Live</b>	1 h bis 14 d	1 ms bis ca. 2 Mio. Jahre
<b>Max. Nachrichtengröße</b>	64 KB	Unbegrenzt, Default 10.000 KB
<b>Kompression</b>	Nein	Ja
<b>Kosten</b>	Kostenloses Einstiegsangebot, danach pro Request und Datenvolumen	Keine zusätzlichen Kosten zu WLS-Lizenz

Tabelle 1: Vergleich AWS SQS Queues und WebLogic JMS Queues

Der **Relational Database Service (RDS)** von AWS erlaubt das Anlegen von MySQL-Instanzen auf Knopfdruck – ähnlich wie Sie es von klassischen Webhostern mit einem Datenbankpaket kennen. Im Gegensatz zu klassischen Webhostern können Sie für Ihre Datenbank auch zusätzliche Read-Replica-Instanzen anfordern, um die

Leserate zu erhöhen oder Multi-Availability-Zone-Instanzen einzurichten, die ein automatisches Failover im Falle des Ausfalls einer Zone garantieren. Im Rahmen der RDS-Konfiguration werden die Größe der Datenbank und des Datenbankservers sowie das Zeitfenster für das Anfertigen von Sicherheitskopien und der Instandhaltung festgelegt. Das Bereitstellen von Oracle Datenbanken statt MySQL mittels RDS ist von Amazon und Oracle angekündigt. Genügt also RDS Ihren Anforderungen, dann steht Ihnen eine Datenbank bereits nach einem einfachen Dienstaufzuruf zur Verfügung.

**Elastic Load Balancing (ELB)** ist Amazons Antwort in der Cloud auf hardwarebasierte Loadbalancer in klassischen Umgebungen. Nebenbei bemerkt ist ELB ein schönes Beispiel für die Idee Infrastructure as a Service (IaaS). ELB ersetzt den Kauf einer teuren Hardwarekomponente durch die stundenweise Verwendung und Bezahlung eines Cloud-Dienstes.

Es ist auch ohne Weiteres möglich, einen softwarebasierten Loadbalancer wie HAProxy in einer dedizierten Cloud-Instanz zu betreiben, allerdings haben Messungen gezeigt, dass aufgrund der Virtualisierung die I/O-Rate bei mehr als hunderttausend Paketen pro Sekunde für eine Cloud-Instanz begrenzt ist.

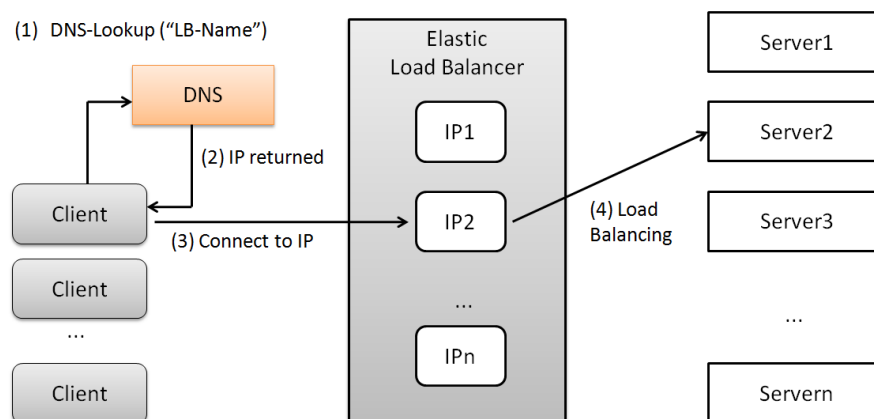
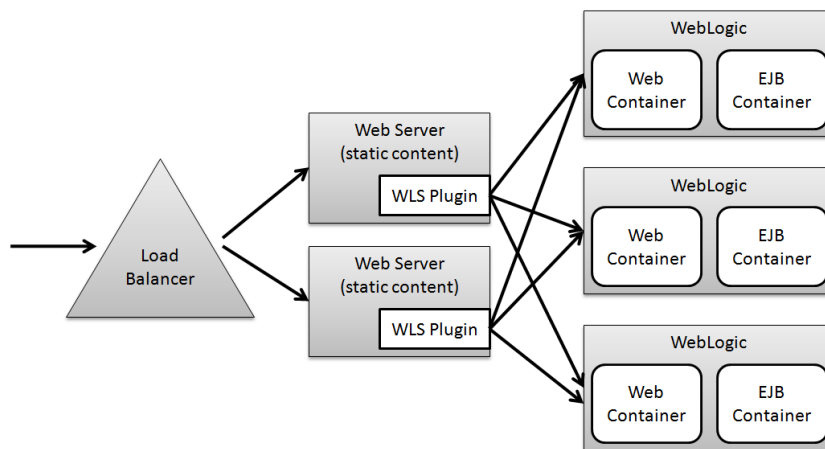


Abbildung 2: Elastic Load Balancing

ELB ist, ähnlich wie SQS oder SNS, ein Cloud-Dienst, der nicht an eine EC2-Instanz gekoppelt ist. Bei der Konfiguration von ELB wird ein DNS-Name generiert, über den der Loadbalancer angesprochen wird. Bei erhöhter Last skaliert ELB intern automatisch und trägt zusätzliche IP-Adressen unter dem von den Clients verwendeten DNS-Namen ein (siehe Abbildung 2).

Vermutlich denken Sie bei Cloud Computing nicht als erstes an ein **Content Distribution Network (CDN)**, trotzdem bieten AWS und viele andere gängige Clouds diesen Dienst an.

Wie in Abbildung 3 gezeigt, werden bei einer klassischen Architektur den Applikationsservern oft Webserver vorangeschaltet. Bei Anwendungen mit einem hohen statischen Anteil (HTML, CSS, Bilder, Video, Musik etc.) können so die Anfragen nach statischen Inhalten schneller von einem der Webserver beantwortet werden, was die Last auf den Applikationsservern reduziert.



*Abbildung 3: Klassische WebLogic Architektur*

Mit dem CDN ist es nun möglich, diese Webserver durch eine skalierbare Alternative zu ersetzen. Amazons CloudFront kann statische Inhalte in 18 weltweit verteilten, so genannten Edge-Locations cachen, die breitbandig ans Internet angebunden sind (siehe Abbildung 4). Durch die zahlreichen Caches skaliert das CDN besser als ein einfacher Webserver, es verringert sich die Latenzzeit für die Clientzugriffe und es entfallen die Installation, der Betrieb und die Wartung der Webserver.

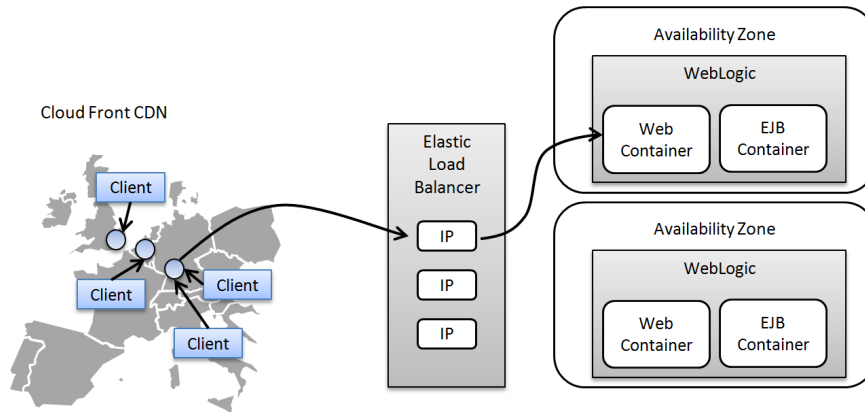


Abbildung 4: WebLogic in der Cloud mit CDN und ELB

Greift man beispielsweise von einem Client in München auf statische, für das CDN freigeschaltete Daten zu, dann reduziert sich die Anzahl der Zwischenstationen laut `traceroute` von 21 (Daten liegen im AWS Rechenzentrum in Dublin) auf 10 (Daten liegen im CloudFront CDN gecached in Frankfurt). In Abbildung 5 ist das vollständige Routing grafisch dargestellt.



Abbildung 5: Routing mit CDN: München-Frankfurt. Ohne CDN: München-Dublin

Zum Schluss sei angemerkt, dass alle hier besprochenen Dienste cloudspezifisch sind und Sie sich dadurch fest an AWS binden. Selbst der zweitgrößte Cloud Provider Rackspace kann nur einen Bruchteil dieser AWS Dienste mit anderen Schnittstellen anbieten, was die Migration zu einem anderen Anbieter beim Disaster

Recovery eines ausgefallenen Cloudanbieters deutlich erschwert. Sollte es Ihnen nicht gerade wie Wikileaks gehen – die komplett von AWS verbannt wurden – besteht aber die Möglichkeit eines Disaster Recovery in einer anderen AWS Region. Zum Thema Disaster Recovery in der Cloud, das ja oft als Killerargument gegen öffentliche Clouds in Spiel gebracht wird, sei angemerkt, dass Wikileaks zwei Stunden nach dem Abschalten des AWS Accounts bei einem schwedischen Anbieter wieder erreichbar war.

### **Middleware in der privaten Cloud: Oracle Exalogic**

Mit der Open Source Software Eucalyptus ist es grundsätzlich möglich, auch in einer privaten Cloud dieselben Schnittstellen wie in Amazons AWS zu verwenden [Euca].

Oracles Cloud-Lösung für den Aufbau einer privaten Cloud ist die Exalogic V2.2, deren Auslieferung gerade begonnen hat. Bei der Exalogic wurden Hardware und Software aufeinander abgestimmt und für OFM optimiert (Reduktion der Anzahl der Buffer Copies, Vergrößerung der MTU auf 64 K sowie optimierte WebLogic Workmanager Einstellungen für 24 Hardware-Threads pro Compute Node). Die Exalogic ist als viertel, halbes oder ganzes Rack erhältlich. Der Listenpreis eines halben Exalogic 2.2 Racks mit 16 Compute Nodes und insgesamt 192 Xeon Cores, 1,5 TB DRAM und Quad Data Rate (QDR) InfiniBand mit 40 Gb/s liegt bei 675.000 US\$, weshalb die Maschine eher für größere Unternehmen mit hohen Anforderungen an Leistung und Verfügbarkeit interessant sein dürfte. InfiniBand ermöglicht zudem die direkte Anbindung der Exalogic mit Oracles Datenbankmaschine Exadata mit Latenzzeiten im einstelligen Mikrosekundenbereich. Die JDBC-Treiber und Datasources wurden optimiert, um die InfiniBand-Anbindung effizient nutzen zu können.

Die Akzeptanz von Oracles privater Cloud im Umfeld des Cloud Computing wird vor allem durch die Managementsoftware bestimmt sein. Diese muss sich zunächst in der Praxis an der Umsetzung der drei Kriterien aus der Definition für Cloud Computing messen lassen (Elastizität, Pay-on-Demand, Ressourcenanforderung als Dienst).

## Referenzen:

[OraAMI] <http://aws.amazon.com/amis/Oracle>

[OraRDS] <http://aws.typepad.com/aws/2011/01/coming-soon-oracle-database-11g-on-amazon-rds-1.html>

[OraEC2] <http://aws.amazon.com/solutions/global-solution-providers/oracle/faqs/#2>

[RightS] <http://www.rightscale.com/>

[RackS] <http://www.rackspace.com/cloud/>

[Euca] <http://open.eucalyptus.com/>

[Exalogic] <http://www.oracle.com/us/products/middleware/exalogic/index.html>

[CCBuch] Middleware and Cloud Computing, Frank Munz, ISBN 978-0980798005, <http://www.amazon.de/dp/0980798000/>

## **Kontakt:**

*Dr. Frank Munz*

*www.munzandmore.com*

Oracle on Amazon Web Services and Rackspace Cloud

# MIDDLEWARE AND CLOUD COMPUTING

Frank Munz



Independent, comprehensive and engaging.

<http://www.amazon.de/dp/0980798000/>