

Cloud- Anwendungen

MEHR WISSEN IN KOMPAKTER FORM:

Weitere Architektur-Spicker

gibt es als kostenfreies PDF unter

www.architektur-spicker.de

5
NR.

Dieser Spicker zeigt, wie Sie Anwendungen bauen, die das Potential einer Cloud-Umgebung voll ausschöpfen.

IN DIESER AUSGABE

- Welche Technologien und Fähigkeiten sind entscheidend?
- Wie vermeiden Sie Fehler?
- Wie bleiben Sie beweglich?



Worum geht's? (Herausforderungen/Ziele)

- ➔ Ihre Organisation entwickelt zukünftig für die Cloud. Welche grundlegenden Entscheidungen stehen an?
- ➔ Sie entwickeln neue Cloud-Anwendungen. Worauf achten Sie bei Architekturentwurf und Technologieauswahl?
- ➔ Sie migrieren eine bestehende Anwendung in die Cloud. Wie gehen Sie vor?
- ➔ Es gibt Bedenken bezüglich Cloud-Lösungen. Wie entkräften Sie diese? Wo ist was dran?



Cloud-Computing: Was und warum?

Beim Cloud-Computing stellt ein Anbieter seinen Kunden IT-Dienste und Ressourcen (z. B. Speicher- oder Rechenkapazität) über das Internet zur Verfügung.

Charakteristisch dabei:

- Selbstbedienung – die Kunden buchen die Ressourcen selbst, beispielsweise über ein Web-Frontend („Self Service“)
- Genutzte Kapazitäten können nach Bedarf (unbeschränkt) wachsen und schrumpfen („Elastizität“)
- Die Zahlung erfolgt nach Verbrauch, dabei keine oder geringe Fixkosten und keine langfristige Anbieterbindung („Pay per use“)

Beispielhafte Motivationen für den Einsatz

- Höhere Verfügbarkeit und Zuverlässigkeit erreichen
- IT-Kosten am tatsächlichen Bedarf ausrichten (z. B. klein starten und bei Erfolg wachsen)
- Lastspitzen in der Produktion auffangen (z. B. Monatsabschluss, Weihnachtsgeschäft)
- Test- und Entwicklungsumgebungen schneller bereitstellen



Liefermodelle im Cloud-Computing

Die Liefermodelle unterscheiden sich vorrangig darin, wer wem etwas anbietet.

Potentielle Stärken:

- geringe Start-/Fixkosten
- unbegrenztes Wachstum
- globale Verfügbarkeit
- leicht an Innovationen teilhaben können

Public Cloud:

Jedermann zugängliches Angebot eines Dritten. Nutzung der Ressourcen als ein Kunde unter vielen.

Private Cloud:

Angebot des eigenen Unternehmens im eigenen Rechenzentrum. Nutzung der Ressourcen gemeinsam mit anderen Projekten/Teams der Organisation.

Hybrid Cloud:

Mischform aus Public und Private, um die Stärken der Modelle verbinden zu können.

Potentielle Stärken:

- Kontrolle, insbesondere über die Daten
- Nähe zu eigenen (internen) Benutzern
- Nähe zu eigenen Bestands-/Legacy-Systemen

Das Ausschöpfen der Potentiale für das eigene Vorhaben hängt insbesondere vom konkreten Cloud-Anbieter ab. Vendor Lock-in zählt zu den Top Cloud-Risiken.



Servicemodelle im Cloud-Computing

Die Service-Modelle (aaS = „as a Service“) unterscheiden sich in den angebotenen Ressourcen und darin, wie stark Anbieter darunter liegende Details verstecken.

	stellt bereit ...	Beispiele	Abstraktion von Implementierungs-details	Nutzungsszenarien für Individualentwicklung
SaaS Software as a Service	Funktionalität in Form einer Anwendung	<ul style="list-style-type: none"> • GMail • Salesforce • Cisco WebEx • Atlassian Confluence/ JIRA (auch public) 	hoch	<ul style="list-style-type: none"> • Seinen Kunden Softwareprodukte in diesem Modell anbieten • Angebote Dritter im eigenen Entwicklungsprozess (z. B. GitHub) oder als Backing Service nutzen
PaaS Platform as a Service	einen Rahmen für Anwendungen des Kunden/Nutzers	<ul style="list-style-type: none"> • Heroku • Elastic Beanstalk • CloudFoundry • OpenShift 		Eigene Anwendungen in diesem Technologie-Stack entwickeln und in der PaaS-Umgebung betreiben
IaaS Infrastructure as a Service	Speicher- und Rechenkapazität, Netzwerk	<ul style="list-style-type: none"> • Amazon Web Services • Microsoft Azure • Google Compute Engine • OpenStack • VMware vCloud 	groß	Eigene Infrastruktur/Zielumgebung durch Cloud-Lösung ersetzen

XaaS ist als Abkürzung für „Anything as a Service“ gebräuchlich.
 FaaS („Function as a Service“) ist die Spezialform ein PaaS, in der der Kunde Funktionen hochlädt und ausführen lässt, ohne eine Umgebung dafür zu definieren („serverless“).
 Die Bezahlung erfolgt nach Anzahl Aufrufen und/oder Rechenzeit.

Mögliche Einflußnahme, Gestaltungsspielraum

Ein Mischpult für Ihre Cloud-Anwendungen

Steuern Sie Ihre Anwendungs-/Plattformarchitektur so aus, dass Sie auf Einflüsse reagieren können, wichtige Fähigkeiten erlangen und so die gewünschten Auswirkungen erzielen!

Input

Neben typischen Architektur-relevanten Anforderungen wie Rahmenbedingungen, Qualitätszielen und Risiken fordern variable Größen und unvorhersehbare Ereignisse Ihre Cloud-Anwendungen heraus.

LAST

Anzahl Anfragen, Nutzer ...
Insbesondere auch Schwankungen und Lastspitzen

req/s

MARKTDRUCK

Neue Anforderungen und Kundenwünsche, motiviert z. B. durch den Wettbewerb

bar

STÖRUNGEN

- Teilausfall Infrastruktur
- Ausfall Fremdsystem

Beim Festlegen von Zielumgebung, Makro-Architektur und Vorgehen sowie der Erstellung neuer Anwendungen in diesem Rahmen treffen Sie viele Entscheidungen. Ein „Hochregeln“ folgender Stellschrauben erhöht Flexibilität und Komplexität gleichermaßen.

GRANULARITÄT

fein / grob

Micro-services / Self-contained Systems / Deployment-Monolith

VIRTUALISIERUNG

starke / keine

serverless / Container, z. B. Docker / Virtuelle Maschinen

AUTOMATISIERUNG

voll / manuell

Continuous Deployment / Continuous Integration / Unit Tests

STANDARDISIERUNG

gering / hoch

polyglott / Vorschläge/ Good Practices / Richtlinien Blaupause

Wie umfangreich sind einzeln deploybare Teile (Module, Subsysteme, ...) einer Anwendung?
 Wie weit wird von Hardware und Laufzeitumgebung (z. B. OS, Middleware) abstrahiert?
 Wie stark sind Schritte innerhalb des Entwicklungsprozesses automatisiert?
 Wie weit sind der Technologie-Stack und übergreifende Konzepte (z. B. Persistenz) vereinheitlicht?

Betrieb und Weiterentwicklung der Anwendungen gestalten sich mehr oder weniger wirksam, wirtschaftlich und dynamikrobust.

GESCHÄFTSNUTZEN

Konversationsrate

ENTWICKLUNGSGESCHWINDIGKEIT

BPM

REAKTIONSFÄHIGKEIT

Visk.

KOSTEN

0 1 2 4 7 1

EUR

Output Monitor



Migration einer bestehenden Anwendung in die Cloud

Eine exemplarische, schrittweise Migration in die Cloud. Jeweils mit zentralen Entscheidungen und potentiell gewonnenen Erkenntnissen und Vorteilen. Ziel: Risiken und Hindernisse früh zu identifizieren. Mitunter sind Rückschritte erforderlich.

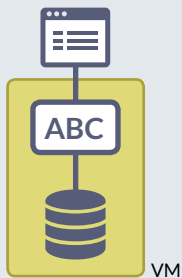
Ausgangspunkt des Beispiels:
Klassische 3-Schicht-Applikation
gehostet auf Server im eigenen
Rechenzentrum



- **Client: Browser-basierte Oberfläche**
HTML, CSS, JavaScript
- **Präsentations- und Geschäftslogik (fachliche Subsysteme A, B und C) auf dem Server**
(z. B. Java, .NET, PHP ...)
- **Relationale Datenbank**
(z. B. MySQL, Oracle, DB2 ...)

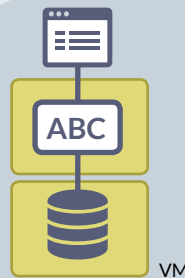
Schritt 1

Logik und Persistenz unverändert auf einer Cloud-VM betreiben.



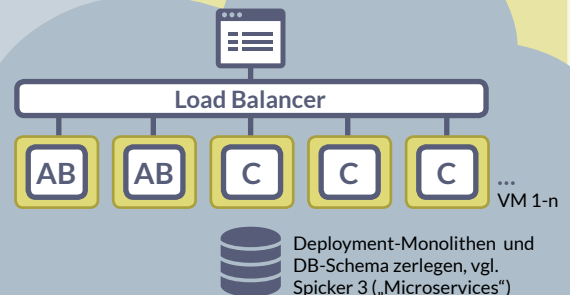
Schritt 2

Logik und Persistenz innerhalb der Cloud trennen.



Schritt 3

Logik horizontal skalieren/vertikal schneiden



Mögliche Implementierung am Beispiel Amazon Web Services (AWS)

Elastic Compute Cloud (Amazon EC2),
AWS Direct Connect, ...

Zentrale Entscheidungen + Maßnahmen

- Public vs. Private entscheiden
- Cloud-Anbieter/Plattform auswählen
- Security konzipieren (z. B. Auth.)
- (Fremdsysteme bereitstellen, z. B. VPN)
- (Deployment automatisieren)

Wichtige Erkenntnisse für Cloud-Betrieb

- Grundsätzliche Probleme mit der Applikation in der Cloud?
- Antwortzeiten und Latenzen für Clients?

Fazit (ausgeschöpfte Cloud-Potentiale)

Eher Experiment und Zwischenschritt. Sie sammeln erste Erfahrungen und identifizieren Showstopper. Sie können bereits leicht identische Umgebungen bereitstellen, z. B. für eine CI-Pipeline. Der Aufwand, die DB so betreiben zu können, ist mitunter verschenkt (> Schritt 2).

Relational Database Service (Amazon RDS),
Amazon CloudFront, ...

Zentrale Entscheidungen + Maßnahmen

- DB-Option des Cloud-Anbieters wählen
- Daten exemplarisch migrieren
- Features der DB-Option nutzbar machen (Backup, Recovery, Clustering ...)
- Realistische Lasttests durchführen
- (weitere Plattform-Services nutzen)

Wichtige Erkenntnisse für Cloud-Betrieb

- Performance nach Aufteilung und bei Nutzung der Plattform-Services?
- Angedachtes Sizing (CPU, RAM, ...) angemessen?
- Managed Services bringen erhoffte Vorteile und Ersparnisse?

Fazit (ausgeschöpfte Cloud-Potentiale)

Sie profitieren vom Know-How des Anbieters bzgl. der Managed Services (z. B. DB-Backup). Eine einzelne Instanz für die Logik stellt oft ein hohes Risiko dar (> Schritt 3).

Elastic Load Balancing (ELB),
Auto Scaling Groups (EC2), ...

Zentrale Entscheidungen + Maßnahmen

- Session Management ggf. überarbeiten
- Geschäftslogik mehrmals instanzieren
- Load Balancing + Autoscaling einführen
- (funktional zerlegen, Teile unabhängig skalieren, Kommunikation konzipieren)
- (Kommunikation zwischen Anwendungsteilen konzipieren)

Wichtige Erkenntnisse für Cloud-Betrieb

- Logik (ABC) zustandlos?
- Unverändert horizontal skalierbar?
- unabhängige Teile, in die sich die Anwendung zerlegen lässt?

Fazit (ausgeschöpfte Cloud-Potentiale)

Ausfallsicherheit und nutzungsabhängige Kosten erst jetzt erreicht. Die Zerlegung in Vertikalen (A, B, C) ist kein Muss und ... und auch bei einer Weiterentwicklung adressierbar (z. B. Application Strangulation).



Die Top-Probleme beim Migrieren

Beim Migrieren bestehender Anwendung in eine Cloud treten die folgenden Hindernisse und Risiken besonders häufig auf:

- Anbindung an eigene Systeme, die (noch) nicht in die Cloud migriert werden, aber für die Anwendung erforderlich sind.
- Flexible Reaktion auf wachsendem Ressourcenbedarf nur mit hohem Kostenaufwand möglich.
- Netzwerk-Bandbreite reicht nicht aus um die bestehenden Clients effizient zu bedienen.
- Lizenzen für Fremdsoftware (z. B. Middleware) können nicht in die Cloud überführt werden oder führen dort zu hohen Mehrkosten.
- Technische Rahmenbedingungen (z. B. ungewöhnliches OS oder spezielle Hardware) sind nicht in der Cloud abbildbar.
- Die spätere Migration auf eine andere Cloud-Plattform wäre mit einem ähnlichen Aufwand verbunden („Vendor Lock-in“).

Als Maßnahmen zur Minderung bieten sich oftmals Abstraktion und Prototyping an – eine regelmäßige Reflektion ist in jedem Falle angezeigt (vgl. Spicker 4: Architektur-Reviews).

Zentrale Aspekte und Prinzipien für Cloud-Anwendungen

Die Graphik zeigt wichtige Themen und Facetten rund um die Anwendungsentwicklung in der Cloud und setzt sie in **Beziehung** zueinander.

Angelehnt an die Prinzipien der Twelve-Factor-App und Diskussionen darum.



Legende: **Unterstützende Methoden, Technologie-Beispiele**

TFA BTT Nummer des Aspekts in den „Twelve Factor Apps“ (TFA), bzw. „Beyond the TFA“ (BTT), siehe weitere Informationen.

Weitere Informationen



Bücher (Auswahl)

- ➔ Christoph Fehling et al.: „Cloud Computing Patterns: Fundamentals to Design, Build, and Manage Cloud Applications“, Springer 2014
- ➔ Kevin Hoffman: „Beyond the Twelve-Factor App. Exploring the DNA of Highly Scalable, Resilient Cloud Applications“, O'Reilly Media 2016



Online-Ressourcen

- ➔ Cloud Native Landscape Project, visuelle Kategorisierung von Technologien und Werkzeugen: <https://github.com/cncf/landscape>
- ➔ Amazon Web Services in Plain English, prägnante Beschreibung der Services bei AWS: <https://www.expeditedssl.com/aws-in-plain-english>
- ➔ The Twelve-Factor App: <https://12factor.net>



Wir freuen uns auf Ihr Feedback: spicker@embarc.de

<http://architektur-spicker.de>