Überblick

Cloud Computing
- Begriffsklärung
- Grundeigenschaften
- Technologie und Algorithmen
- Formen
- Vergleich: Timesharing-Systeme

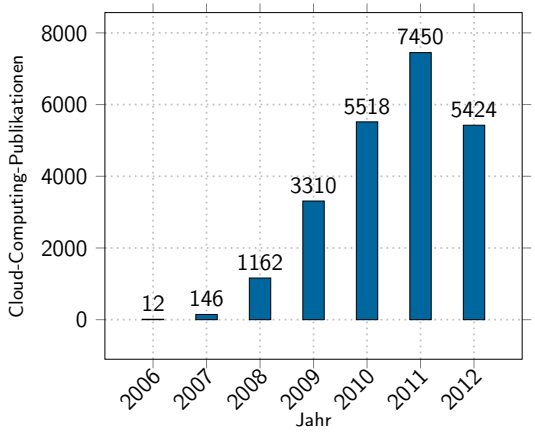
Cloud-Computing-Systeme
- Amazon Web Services
- Twitter

Cloud Computing: Der Anfang. . .

»We call it cloud computing (. . .) «

Eric Schmidt (Google)
Search Engine Strategies Conference, San Jose, 9. August 2006

Cloud Computing: . . . und seine Folgen.



- Gravierende Auswirkungen des Modeworts „Cloud Computing"
  - Forschung
  - Wirtschaft
- Cloud Computing
  - Fokus auf Technik
  - Cloud ≙ Internet, Cloud Computing ≙ Internet + **?**

Cloud Computing: Zeitpunkt, Grundeigenschaften

- Wieso entstand Cloud Computing zu dieser Zeit? Wieso nicht früher?
- **Infrastruktur**, Hard- und Software-Technologie ✓
  - Commodity-Hardware
  - Virtualisierung
- **Systemsoftware**, Verteilte Systeme und deren Algorithmen ✓
  - Parallele, verteilte Datenverarbeitung
  - Schlüssel-Wert-Datenbank (Key-Value-Store)
  - Verteilter Koordinierungsdienst
  - Verteilte, dezentrale Datenhaltung
- **Dienstleistungsprinzip**, Geschäftsmodell („. . . *as-a-Service*") ✓
  - *Service-Oriented Architecture* (SOA)
  - Infrastructure-as-a-Service
  - Platform-as-a-Service
  - Software-as-a-Service

Cloud Computing: Zeitpunkt, Grundeigenschaften

- Erfüllbarkeit der Grundeigenschaften von Cloud-Computing-Systemen
- **Skalierbarkeit**, unter Wahrung von:
  - Konsistenz (*Consistency*)
  - Verfügbarkeit (*Availability*)
  - Partitionstoleranz (*Partition tolerance*)

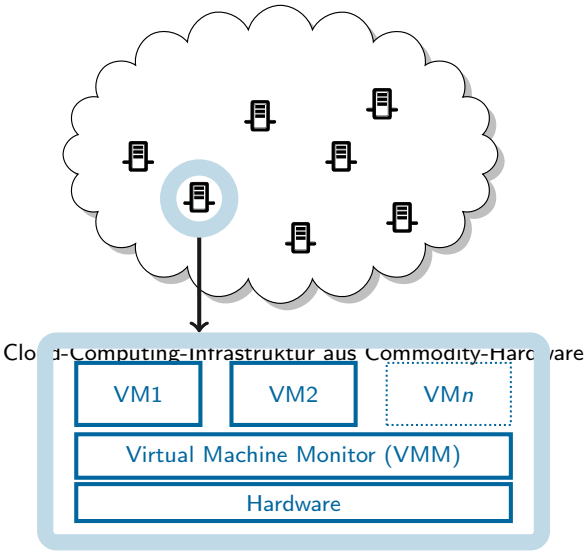
  → CAP-Theorem
- **On-Demand**, zum Ermöglichen von:
  - dynamischer Zuordnung von Ressourcen
  - Abrechnung nach tatsächlichem Verbrauch
- **Robustheit**, zur Vermeidung von:
  - Inkonsistenzen im Datenbestand
  - (unkontrollierter) Fehlerausbreitung im System

Systeme aus Commodity-Hardware

Systeme aus Commodity-Hardware

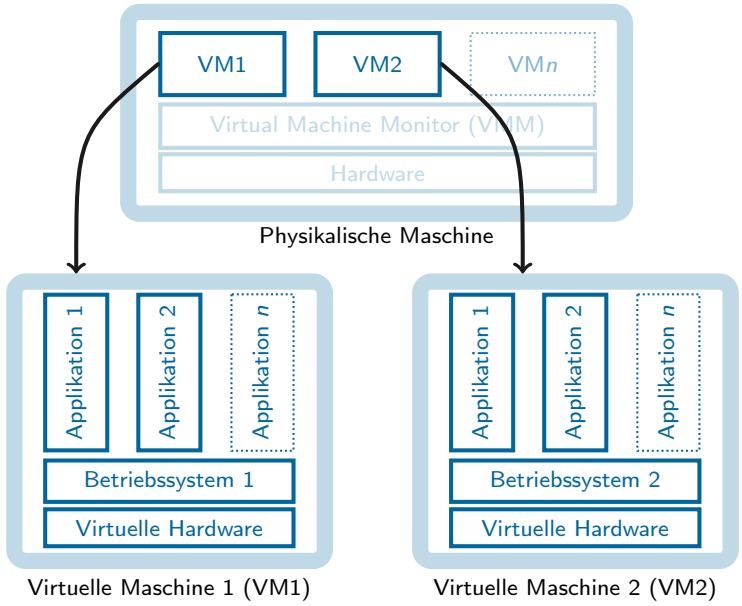
- Charakteristik
  - hoher Grad der Verteiltheit
  - „unbegrenzte" Speicherkapazität
- Herausforderungen für verteilte Systeme
  - **Ausfälle sind** nicht die Ausnahme sondern **der Regelfall**
  - System muss trotz vieler, regelmäßig ausfallender Knoten die geforderten Kriterien (→ CAP-Theorem) gewährleisten
- Offene Standards
  - Verwendung heterogener Knoten fordert Verwendung offener Standards
  - Adressiert Problem der Anbieterabhängigkeit („*Vendor Lock-In*")
  - Gilt sowohl für Soft- als auch Hardware

Commodity-Hardware, Virtualisierung



Physikalische Maschine

Virtualisierung



Systemkomponenten einer Cloud-Computing-Infrastruktur

- System-Eigenschaften
  - Massiver Grad an Verteiltheit (→ *„Who owns the most servers"*)
  - Dynamische Anpassung verfügbarer Rechen- und Speicherressourcen
  - Häufiger Ausfall einzelner Komponenten des Gesamtsystems
- Typische Anwendungsszenarien in einer Cloud-Umgebung
  - Abarbeitung sehr großer Probleme
  - Verarbeitung riesiger Datenmengen
  - Beispiele: Suchindex-Erstellung, Graph-Operationen in sozialen Netzen
- Algorithmische Abbildung
  - Betriebsumgebung verlangt angepasste Systeme und Algorithmen
  - Aufteilung großer Aufgaben in **kleinere**, **verteilbare** und damit **parallel** zu verarbeitende Teilaufgaben (→ teile und herrsche)
  - Neue Konzepte zwingend notwendig um Cloud-Computing-Infrastruktur überhaupt effizient zu nutzen

Verteilte Datenverarbeitung: MapReduce, Hadoop

- MapReduce
  - Framework zur parallelen Bearbeitung großer Datenmengen
  - *„MapReduce: Simplified Data Processing on Large Clusters"* (OSDI'04)
  - Wichtige, einfache Datenstruktur: Schlüssel-Wert-Paare (Key-Value)
- Ablauf
  - Map- und Reduce-Phase(n)
  - Anwendungs-Programmierer implementiert Map- und Reduce-Funktionen
  - Jede Phase hat Schlüssel-Wert-Paare als Ein- und Ausgabe
  - Mehrmalige Ausführung von Jobs zur Reduktion von Verzögerungen
- Apache Hadoop (→ `http://hadoop.apache.org`), u.a.:
  - Hadoop MapReduce
  - Hadoop Distributed File System (HDFS)

Jeffrey Dean and Sanjay Ghemawat
MapReduce: Simplified Data Processing on Large Clusters
In *Proceedings of 6th Symposium on Operating Systems Design and Implementation (OSDI'04), pp.137–149*, 2004.

`http://www.usenix.org/events/osdi04/tech/dean.html`

Skalierbare Schlüssel-Wert-Datenbanken

- Schlüssel-Wert-Datenbanken (Key-Value-Stores)
  - Key-Value-Stores haben keine relationalen Datenbeziehungen und besitzen kein Datenbankschema
  - Auslagerung großer Teile der Logik (Datenbank → Applikation)
  - Extrem effizienter Datenzugriff auf Schlüssel-Wert-Paare
  - Beispiele: Amazon DynamoDB, Apache Cassandra, Google BigTable
- Abgrenzung zu SQL-Datenbanken
  - Aufweichen von Garantien ermöglicht Skalierbarkeit
  - BASE (Basically Available, Soft state, **Eventual consistency**) → NoSQL
  - ACID (Atomicity, Consistency, Isolation, Durability) → SQL
- Apache CouchDB (Cluster of Unreliable Commodity Hardware)
  - Schemafreie Schlüssel-Wert-Datenbank
  - Datenverwaltung in JSON (Java Script Object Notation)
  - Peer-to-Peer für verteilte Datenverwaltung

→ Demo CouchDB

Verteilte Dateisysteme, Verteilte Koordination

- Verteilte Dateisysteme
  - Unkonventionelle Schnittstellen, *sehr* verschieden im Vergleich mit herkömmlichen (verteilten) Dateisystemen → kein POSIX
  - Replikation gewährleistet Fehlertoleranz beim Ausfall von Knoten
  - Beispiele: Amazon S3, Google Filesystem
- Verteilte Koordination
  - Programmier-Schnittstelle ähnelt der von Dateisystemen
  - Umsetzung von Sperrmechanismen (Locking), Anführerwahl
  - Fehlertoleranz durch aktive Replikation
  - Beispiele: Google Chubby, Yahoo ZooKeeper

Patrick Hunt, Mahadev Konar, Flavio P. Junqueira and Benjamin Reed
ZooKeeper: Wait-free Coordination for Internet-scale Systems
In *Proceedings of the 2010 USENIX Annual Technical Conference (ATC'10)*, 2010.
http://www.usenix.org/events/atc10/tech/full_papers/Hunt.pdf

Cloud-Computing-Modelle: „*Everything as a Service*"

- Infrastruktur: „*Infrastructure-as-a-Service*" (IaaS)
  - Anmieten von Ressourcen (virtuelle Maschinen, Speicherkapazität)
  - Beispiele: Amazon Elastic Cloud Computing (kurz: Amazon EC2), Amazon S3, Microsoft Azure, Google Compute Engine
- Plattform: „*Platform-as-a-Service*" (PaaS)
  - Anmieten von Laufzeitumgebung um Applikationen auszuführen
  - Beispiele: AWS Elastic Beanstalk, Google App Engine
- Software/Anwendung: „*Software-as-a-Service*" (SaaS)
  - Statt starrer Lizenzierung von Anwendungen wird nur für den Benutzungszeitraum abgerechnet
  - Beispiele: Google Gmail, Auslagerung ganzer Geschäftsprozesse
- Gemeinsamkeiten
  - Abrechnung nach tatsächlichem Verbrauch („*pay what you use*")
  - Garantierte Skalierbarkeit, keine langfristige Bindung („*on demand*")
  - Prozessauslagerung auf entfernte Systeme
    → Problematik „*Vendor Lock-In*"

Ist das alles neu? Exkurs: Timesharing-Systeme

- „*How to Consider a Computer*" (1959)
  - Erste wissenschaftliche Arbeit zum Thema Timesharing von Bob Bemer (IBM)
  - „The Father of ASCII": hat u.a. Backslash und Escape beigetragen
  - Weitblick: Erste Veröffentlichung zur Jahr-2000-Problematik im Jahre 1971 („*Time and the Computer*")
- „*Utility Business Model*" (1961)
  - Rede von John McCarthy (Stanford University) am Massachusetts Institute of Technology (MIT)
  - Rechenleistung und Spezialanwendungen sollen verkauft werden wie andere Ressourcen — wie Wasser und Strom
- Timesharing-System mit Virtualisierung (1967)
  - IBM Cambridge Scientific Center, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL): IBM CP-40
  - Native Virtualisierung

Vergleich: Timesharing, Cloud Computing

- Gemeinsamkeiten
  - Teilen von Ressourcen
  - Abrechnungsmodell
  - Aufteilung von Anwendungslogik und Benutzerschnittstelle
- Unterschiede
  - Transparenter Zugriff, ortsunabhängig
  - *Unendlichkeit* der Ressourcen
  - Grad der Verteilung, Netzstruktur

Überblick

Cloud Computing
- Begriffsklärung
- Grundeigenschaften
- Technologie und Algorithmen
- Formen
- Vergleich: Timesharing-Systeme

Cloud-Computing-Systeme
- Amazon Web Services
- Twitter

Amazon Web Services (AWS)

- Idee: Ungenutzte Ressourcen der Amazon-Rechenzentren gewinnbringend vermieten
- Dienste ermöglichen den Aufbau eigener, komplexer Systeme in einer Cloud-Infrastruktur (Auszug):
  - Elastic Compute Cloud (EC2) – Betrieb virtueller Maschinen
  - Simple Storage Service (S3) – Netzwerkbasierter Speicher-Dienst
  - Elastic Load Balancing – Lastverteilung für EC2
  - Elastic Map Reduce – MapReduce-Framework basierend auf EC2 und S3
  - DynamoDB – Key-Value-Store basierend auf Dynamo
- Die Abrechnung erfolgt nach tatsächlichem Verbrauch **und** Standort
  - Betriebsstunden, Speicherbedarf
  - Transfervolumen, Anzahl verarbeiteter Anfragen
  - Standorte in Nord- und Südamerika, Europa und Asien

  → AWS Preisübersicht: `https://aws.amazon.com/pricing`

Amazon Web Services (AWS)

Twitter

- Twitter und Cloud Computing
  - Als junges Start-Up-Unternehmen zunächst keine eigene Infrastruktur
    → ohne Cloud Computing würde Twitter nicht existieren
  - Nutzt Cloud-Dienste (z. B. Amazon S3) und Projekte wie ZooKeeper
- Zahlen zu Twitter
  - 24 Milliarden Suchanfragen pro Monat
    - Google: > 100 Milliarden
    - Yahoo: 9.4 Milliarden
    - Bing (Microsoft): 4.1 Milliarden
  - Über 400 Millionen Tweets pro Tag
  - Über 100 Millionen aktive Benutzer (pro Monat)
  - Etwa 900 Mitarbeiter
- Rekorde (Tweets-pro-Sekunde, TPS)
  - 25.088 TPS: Ausstrahlung *„Castle in the Sky"* in Japan (Dezember 2011)
  - 15.107 TPS: Wiederwahl Barack Obama (November 2012)
  - 6.049 TPS: Todesfall Steve Jobs (Oktober 2011)

Probleme komplexer Cloud-Computing-Systeme

- Probleme
  - Ständig steigende Systemkomplexität → stets neue Fehlerquellen
  - Fehler müssen so früh wie möglich erkannt werden
  - Entkopplung von Teilsystemen um Fehler-Ausbreitung zu verhindern
- Twitter-Ausfall am 21. Juni 2012

- Amazon-Web-Services-Ausfall (North Virginia) am 29. Juni 2012
  - Ereigniskette:
    Starke Unwetter → Stromausfall → Notstromversorgung fehlgeschlagen
  - Verfügbarkeit großer Firmen, die auf Amazon Web Services angewiesen sind, war stark eingeschränkt (z. B. Instagram, Netflix, Pinterest)

Cloud Computing: Chancen und Tücken

- Cloud Computing ist das Resultat paralleler, teilweise unabhängiger Entwicklung; nicht geplant, aber auch kein Zufall
- Heutige Cloud-Projekte sind erste Prototypen; nicht weniger, aber auch nicht mehr
- Cloud Computing bildet die Grundlage für Unternehmen ohne Infrastruktur; ansatzweise wie Twitter
- Nicht zu vernachlässigen: Risiken durch Abhängigkeiten von Softwarekomponenten und Firmen („*Vendor Lock-In*")
- Cloud-Computing-Vorlesung am Lehrstuhl 4
  Wintersemester 2013/2014: Middleware – Cloud Computing