

Architektur für skalierbare Webanwendungen

Vladislav Faerman

Hochschule für angewandte Wissenschaften München
Fakultät für Informatik und Mathematik

07. Juni 2017

Agenda

Einleitung

Motivation

Theorie

Skalierbarkeit

Wartbarkeit

Architektur

Konzept

Relevante Technologien

Prototyp live

Projekte, die rapide populär werden und ein exponentielles Anwenderwachstum erleben, sind sehr oft nicht im Stande mit rasant steigender Anzahl von Anfragen und großen Datenmengen effizient umzugehen.

Für dieses ernsthaftes Problem kommen zwei wichtige Punkte in Betracht:

- Skalierbarkeit und
- Wartbarkeit

Agenda

Einleitung
Motivation

Theorie
Skalierbarkeit
Wartbarkeit

Architektur
Konzept
Relevante Technologien
Prototyp live

Skalierbarkeit

Skalierbarkeit

Die Fähigkeit eines Systems, aufgrund der wachsenden Anforderungen, entweder die Leistung der vorhandenen Ressourcen zu verbessern oder zusätzlich die neuen Ressourcen hinzufügen.

Bei der Skalierbarkeit sind zwei Arten zu unterscheiden, eine *vertikale* und eine *horizontale* Skalierbarkeit.

Skalierbarkeit

Vertikale Skalierbarkeit

Anstreben einer qualitativen Steigerung der Leistungsfähigkeit der bereits eingesetzten Ressourcen.

Skalierbarkeit

Horizontale Skalierbarkeit

Im Gegensatz zur vertikalen Skalierbarkeit wird bei der horizontalen Skalierbarkeit die Last auf zusätzliche Rechner verteilt.

Das CAP-Prinzip

Im Jahr 2000 präsentierte Eric A. Brewer das CAP-Theorem - ein Ergebnis seiner Forschungen zu verteilten Systemen:

■ Consistency (Konsistenz)

- ▶ Korrektheit der in der Datenbank gespeicherten Daten, auch in allen replizierenden Knoten in einem Cluster

■ Availability (Hochverfügbarkeit)

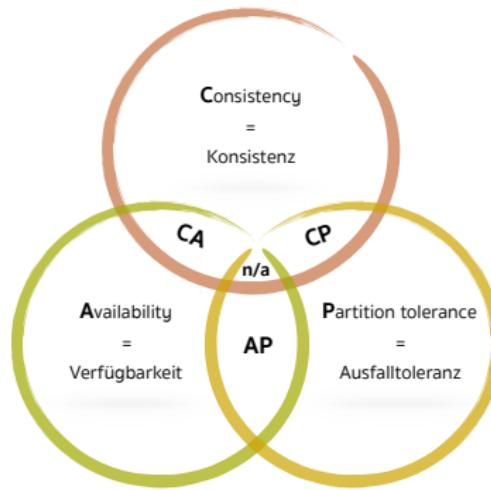
- ▶ Alle Anfragen an das System werden mit akzeptabler Reaktionszeit stets beantwortet

■ Partition Tolerance (Partitionstoleranz)

- ▶ Die Ausfalltoleranz eines Knotens bzw. eines Servers aus einem Cluster

Skalierbarkeit

Das CAP-Prinzip



Die drei Anforderungen sind laut Brewer gleichzeitig nicht zu erfüllen.

Das CAP-Prinzip

Die Anforderungen in Paaren klassifizieren bestimmte Datenbanktechnologien.

■ CA (Consistency und Availability)

- ▶ für die klassischen relationalen Datenbankmanagementsysteme (RDBMS)

■ CP (Consistency und Partition tolerance)

- ▶ z. B. für Banking-Anwendungen

■ AP (Availability und Partition tolerance)

- ▶ die Social-Media-Sites wie Twitter oder Facebook

Die SOLID-Prinzipien

Fünf Prinzipien des objektorientierten Designs:

- Single responsibility principle
- Open/closed principle
- Liskov substitution principle
- Interface-segregation principle
- Dependency inversion principle

Bei korrekter Anwendung dieser Prinzipien erfolgt eine höhere Wartbarkeit am Softwareprodukt.

Dependency Injection (DI)

Dependency Injection ist der nächste Schritt nach Dependency Inversion.

- die Klassen haben die Abhängigkeiten nur zu den Interfaces
- konkrete Implementierungen werden durch die externe Komponente zur Laufzeit eingefügt (injected).
- **Ziel:** Anwendungsobjekte voneinander entkoppelt zu halten: **lose Kopplung**.

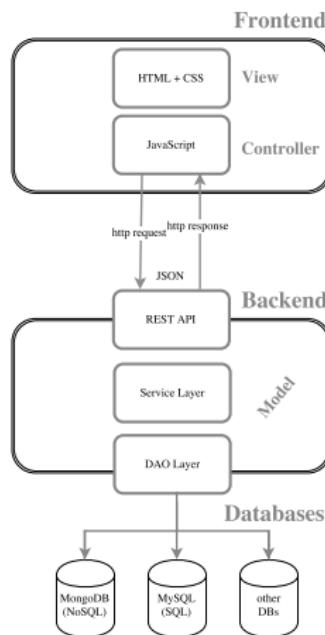
Diese ermöglicht dem Entwickler die Konzentration auf die Entwicklung einzelner Komponenten unabhängig voneinander.

Agenda

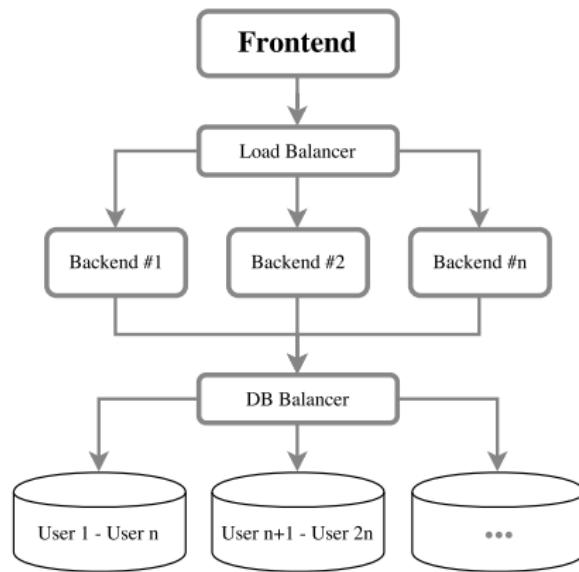
Einleitung
Motivation
Theorie
Skalierbarkeit
Wartbarkeit

Architektur
Konzept
Relevante Technologien
Prototyp live

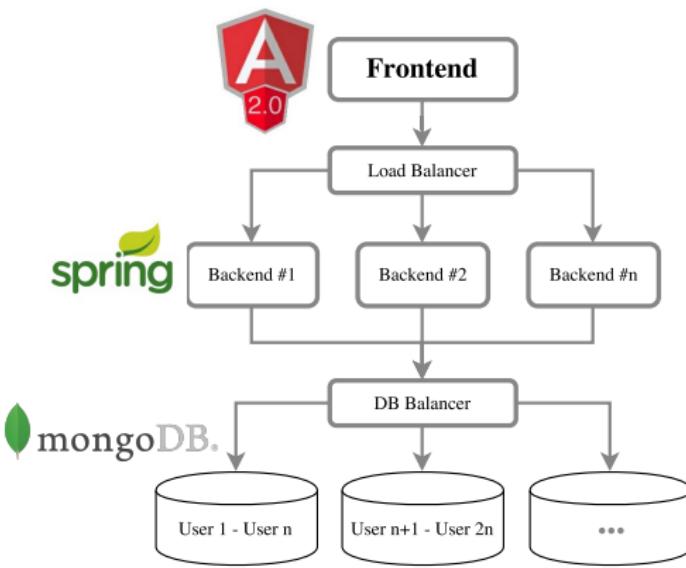
Typische Architektur einer Webanwendung



Konzept



Umsetzbarkeit



Agenda

Einleitung
Motivation
Theorie
Skalierbarkeit
Wartbarkeit

Architektur
Konzept
Relevante Technologien
Prototyp live

NoSQL-Datenbanken

NoSQL erlaubt eine unstrukturierte Haltung von Daten.

Mögliche **NoSQL**-Kategorien:

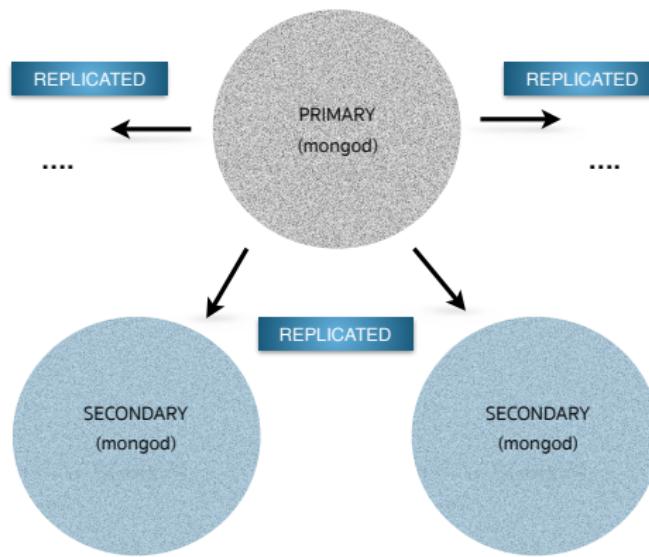
- Key-Value-Datenbank
- spaltenorientierte Datenbank
- Graphen-Datenbank
- dokumentenorientierte Datenbank

MongoDB

MongoDB ist eine von vielen NoSQL-Datenbanken, die als eine schemalose, dokumentenorientierte Open-Source-Datenbank bekannt ist.

Wichtige Konzepte sind **Ausfallsicherheit** und **horizontale Skalierung**.

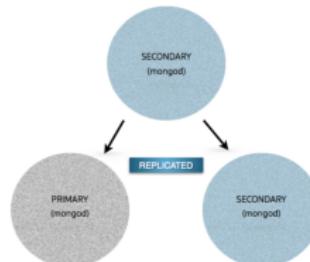
MongoDB: Replikation (Replication)



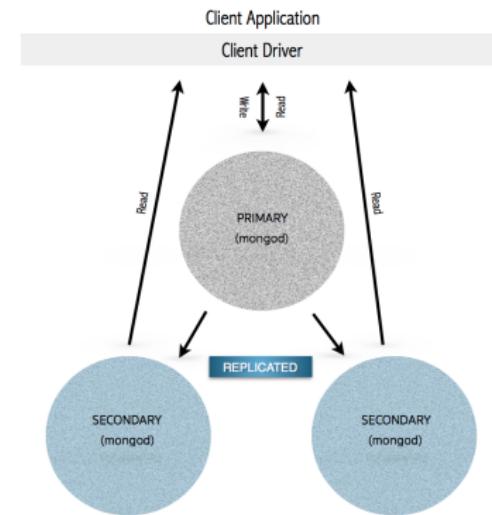
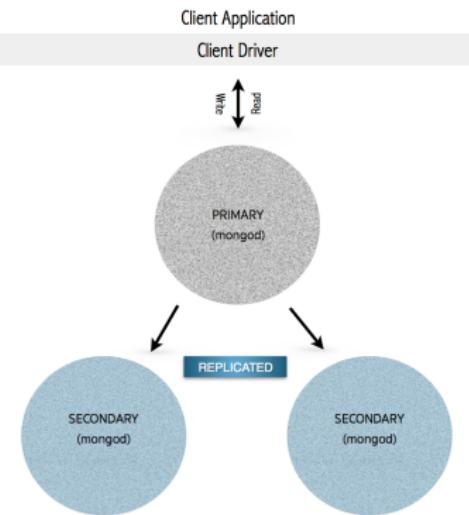
Szenario für eine Replikationsgruppe mit drei Servern in einer Shard



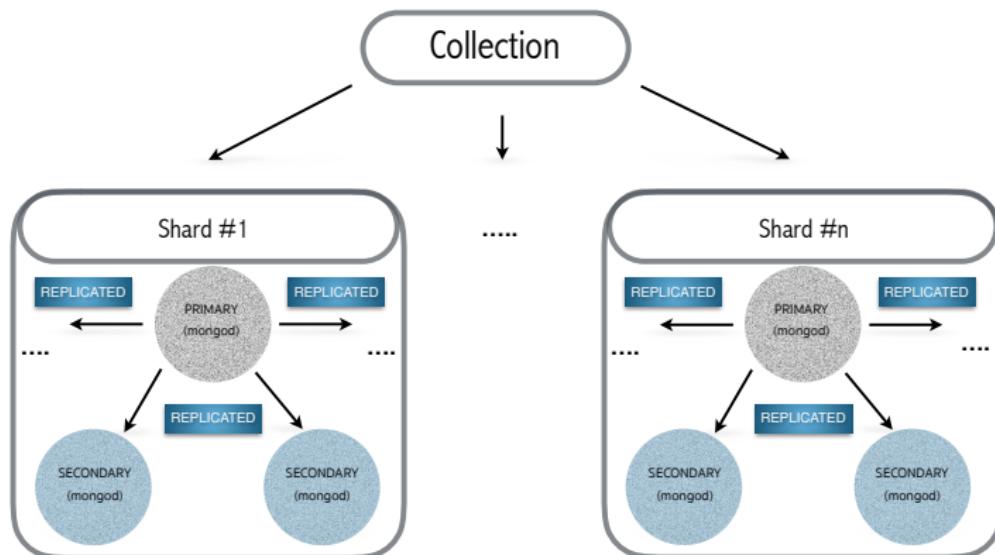
(a) Initialer Zustand des Replica Sets

(b) Ausfall des **Primary**-Knotens(c) Neuer **Primary** wurde gewählt

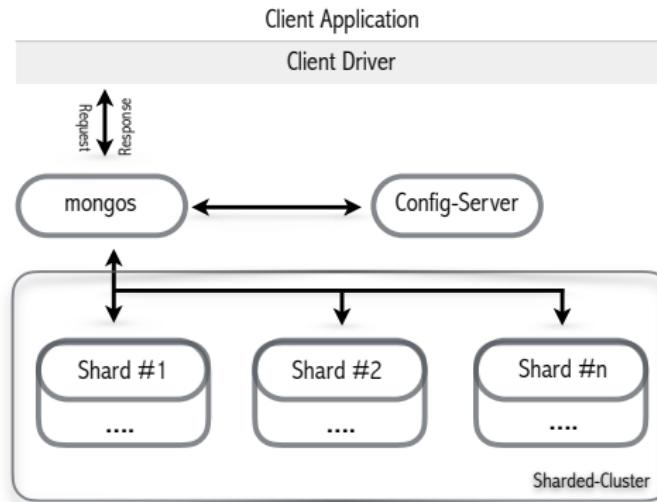
Freischaltung der Lesezugriffe



MongoDB: Horizontale Skalierung (Sharding)



MongoDB: Sharded Cluster

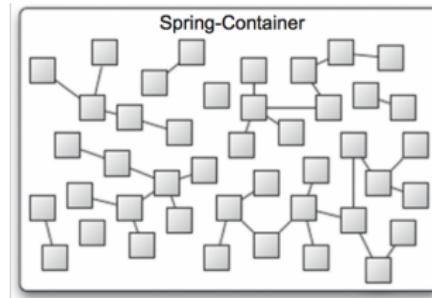


Spring Framework (1)

- Open Source Java Framework
- fundamentale Mission:
 - ▶ die Entwicklung mit Java/Java EE zu vereinfachen
- zwei wichtige Kernstrategien:
 - ▶ lose Kopplung durch „Injizieren“ von Abhängigkeiten und Interface-Orientierung
 - ▶ Reduzierung von Boilerplate-Code durch Vorlagen

Spring Framework (2)

- Verwaltung von einfachen Java Objekten, sogenannte Plain-Old-Java-Objects (POJO) durch Spring Framework als **Spring-Beans**.



- Spring nutzt POJOs bzw. Java-Beans, indem sie per DI zusammengesetzt werden.

AngularJS 2 - JavaScript Framework

AngularJS dient zur Entwicklung von so genannten Single-page-Anwendungen.

In einer Single-page Webanwendung werden die HTML-Seite mit dem ganzen Inhalt nur einmal geladen und die Teile davon dynamisch nachgeladen oder upgedated.

Entwickelt in TypeScript.

Agenda

Einleitung
Motivation
Theorie
Skalierbarkeit
Wartbarkeit

Architektur
Konzept
Relevante Technologien
Prototyp live

!!! Live !!!



