

# Middleware – Cloud Computing – Übung

## Verteilte Dateisysteme & Container: Hadoop Distributed File System

---

Wintersemester 2021/22

Michael Eischer, Laura Lawniczak, Tobias Distler

Friedrich-Alexander-Universität Erlangen-Nürnberg  
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)

[www4.cs.fau.de](http://www4.cs.fau.de)



Lehrstuhl für Verteilte Systeme  
und Betriebssysteme



FRIEDRICH-ALEXANDER  
UNIVERSITÄT  
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Verteilte Dateisysteme

Dateisysteme

Hadoop Distributed File System (HDFS)

## **Verteilte Dateisysteme**

---

### **Dateisysteme**

## Lokale Dateisysteme

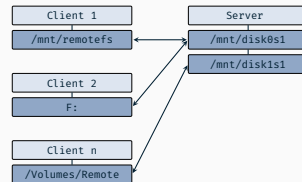
- Adressierung von Daten auf physikalischen Datenträgern
- *Beispiele:* FAT32, Ext4, Btrfs

## Netzwerk-Dateisysteme

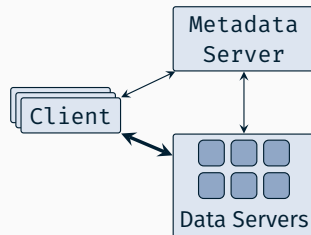
- Zugriff auf entfernte, persistente Daten über Rechnergrenzen
- *Beispiele:* Andrew File System (AFS), Network File System (NFS)

### ⚡ Probleme

- Fehleranfälligkeit (z.B. Ausfall von Netzwerkverbindungen)
- Flaschenhalsbildung durch Ungleichgewicht (Anzahl Clients vs. Server)



- Trennung von Belangen (engl. *separation of concerns*)
  - Indizierung
  - Datenverwaltung
- Replikation der Daten für höhere Ausfallsicherheit  
→ Einhaltung von *Service-Level-Agreement*, kurz: SLA
- Auflösung von Konflikten zwischen Clients
- Beispiele:
  - Ceph
  - Google File System
  - **Hadoop Distributed File System**



## **Verteilte Dateisysteme**

---

**Hadoop Distributed File System (HDFS)**

- Teil des Apache Hadoop Frameworks für skalierbare, verteilte Datenverarbeitung



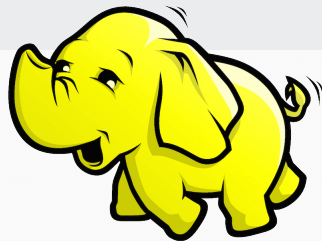
Quelle der Illustration: <https://blog.codecentric.de/2013/08/einfuehrung-in-hadoop-die-wichtigsten-komponenten-von-hadoop-teil-3-von-5/>

## Konzepte

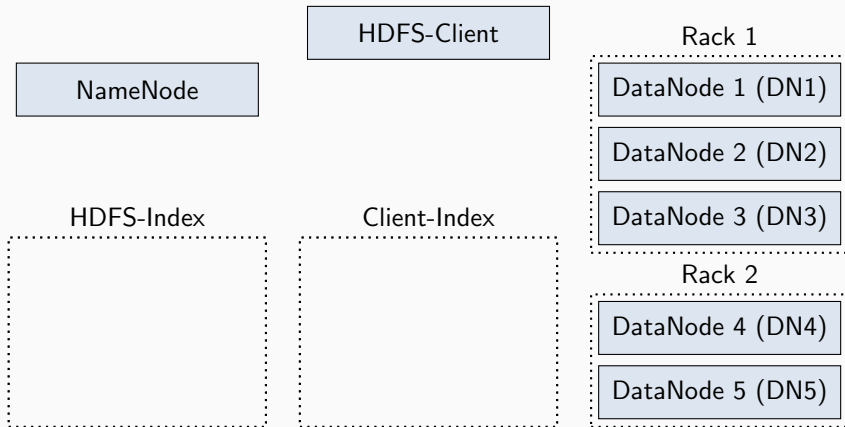
- Write-once, read-many (WORM)
- Replikation
- Datenlokalität („rack-aware“)

## Architektur

- HDFS-Client
- NameNode → Namensraum (Index, Metadaten)
- DataNode → Blockreplikate (Blockdaten + Metadaten)

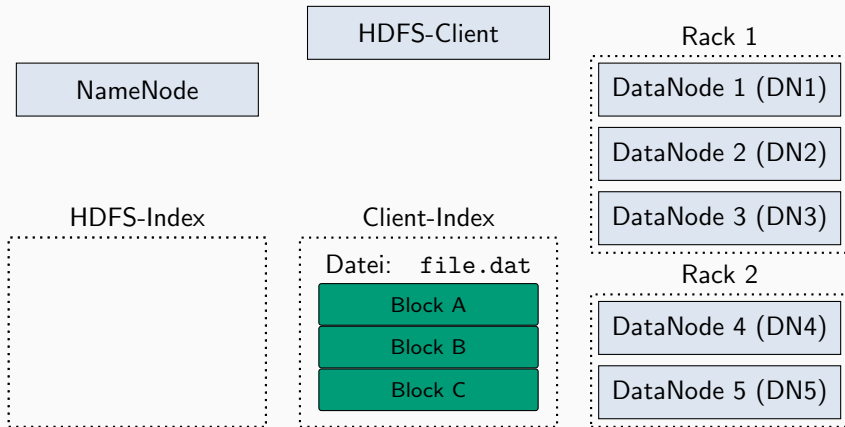


# Hadoop Distributed File System (HDFS)

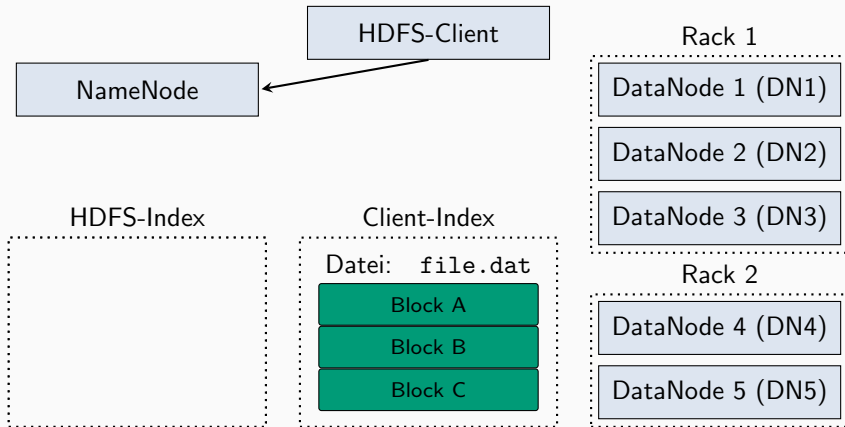


## ■ System-Konfiguration

- 1x HDFS-Client
- 1x NameNode
- 5x DataNodes (Rack 1: DN1–3, Rack 2: DN4–5)

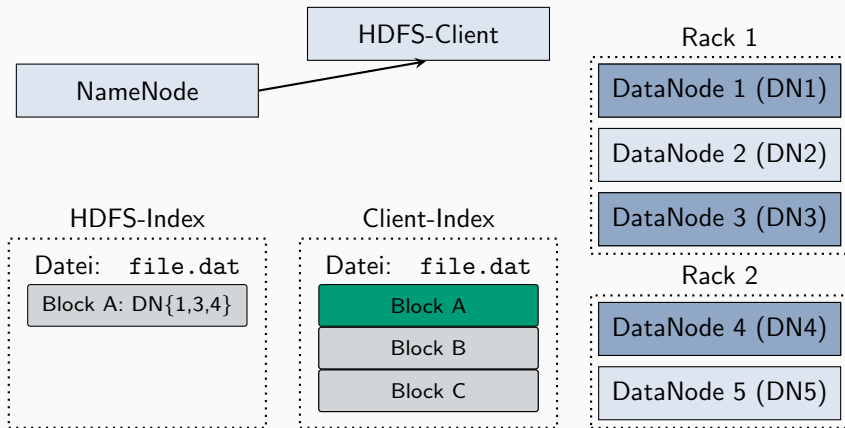


- HDFS-Client legt die aus drei Blöcken (A, B und C) bestehende Datei `file.dat` im HDFS an



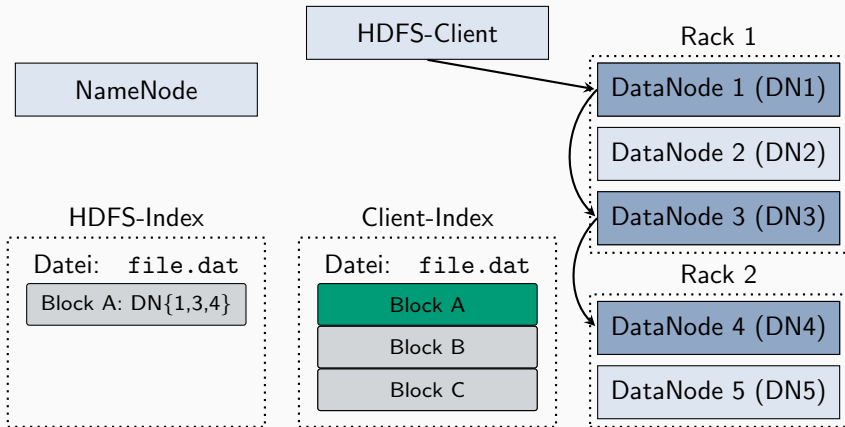
1. HDFS-Client → NameNode:

Anforderung eines sog. Lease (dt. *Miete*) für das Schreiben der Datei `file.dat`

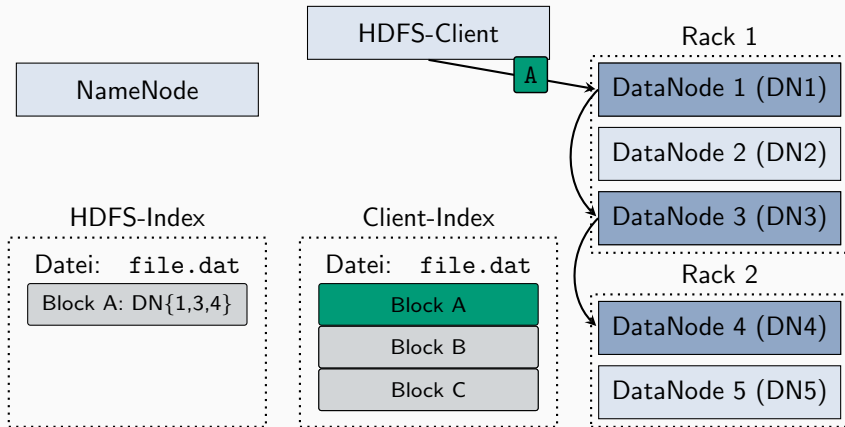


## 2. NameNode → HDFS-Client:

Erteilung des Lease, Erzeugung einer Block-ID für den ersten Block (Block A),  
Zuteilung der Replikate (DN1, DN3 und DN4)

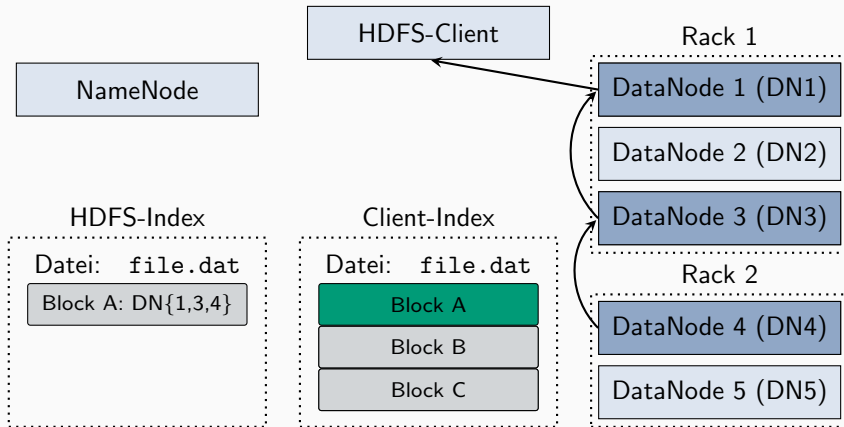


3. „Daten-Pipeline“ zur Vorbereitung der Schreiboperationen von Block A:  
HDFS-Client — DN1 — DN3 — DN4



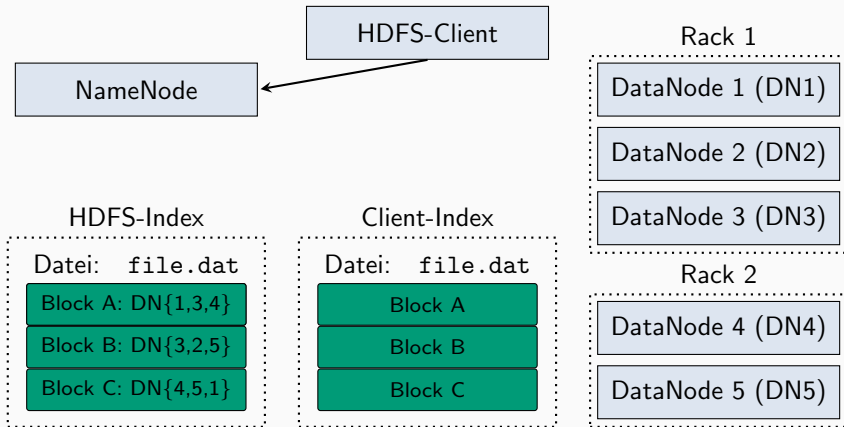
#### 4. Durchführung der Schreiboperationen:

HDFS-Client sendet Block A an DN1, DN1 sendet empfangenen Block A an DN3, DN3 sendet empfangenen Block A an DN4



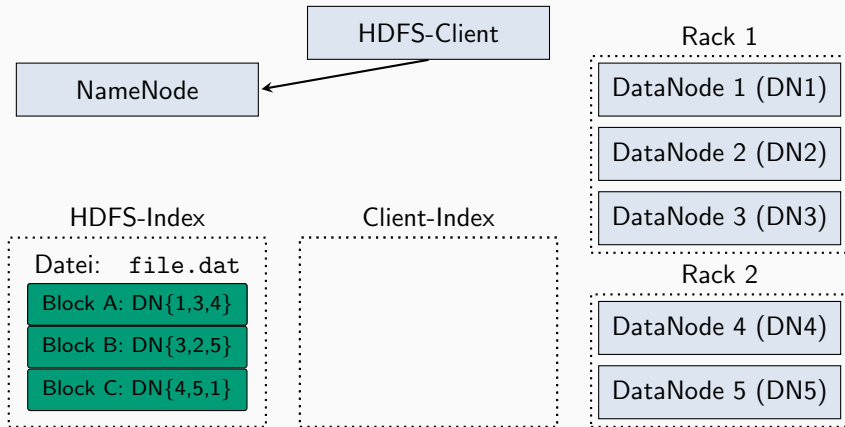
##### 5. Bestätigung der Schreiboperationen:

Jede DataNode bestätigt das erfolgreiche Schreiben von Block A entlang der Pipeline (Abbau)



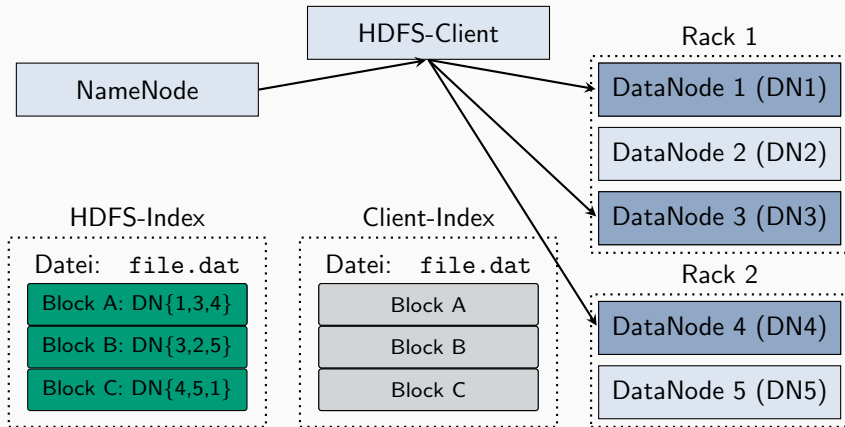
#### 6. HDFS-Client → DataNodes:

Analog werden die restlichen Blöcke der Datei vom HDFS-Client an die DataNodes verschickt;  
HDFS-Client benachrichtigt NameNode von erfolgreicher Schreiboperation



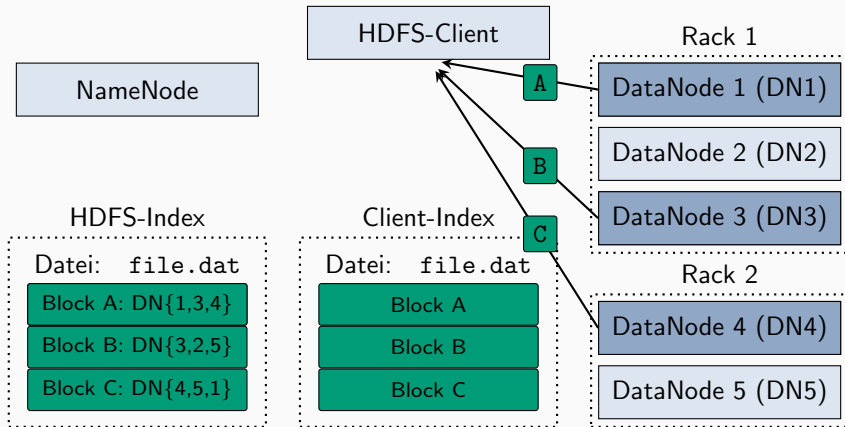
1. HDFS-Client → NameNode:

Anforderung der DataNodes-Liste: Alle DataNodes, die Blöcke der zu lesenden Datei file.dat speichern



2. NameNode → HDFS-Client, HDFS-Client → DataNodes:

Client erhält DataNodes-Liste und wählt den ersten DataNode für jeden der Datenblöcke

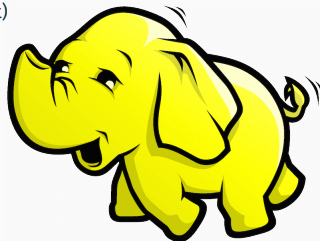


### 3. DataNodes → HDFS-Client:

HDFS-Client liest die Blöcke sequentiell, DataNodes senden die angeforderten Blöcke an den HDFS-Client

## ■ (Weitere) HDFS-Details

- Herzschlag-Nachrichten (engl. heartbeat) von DataNodes zum NameNode
  - Alle drei Sekunden (Default) ein Herzschlag
  - Replikationsfaktor sicherstellen
  - Grundlast bei sehr großen Clustern
- Block-Report: NameNode generiert Metadaten aus den Block-Reports
  - Umfangreicher Bericht über alle Blöcke alle 60 Minuten (Default)
  - Löschen ungenutzter Blöcke
- NameNode
  - *Die Sollbruchstelle des Systems?*



## ■ Literatur



Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler

### **The Hadoop distributed file system**

*Proceedings of the 26th IEEE Symposium on Mass Storage Systems and Technologies (MSST '10), pages 1–10, 2010.*