Karlsruher Institut für Technologie

AIFB

Comparison of Distribution Technologies in Different NoSQL Database Systems

Studienarbeit

Dominik Bruhn | 18.03.2011

INSTITUT FÜR ANGEWANDTE INFORMATIK UND FORMALE BESCHREIBUNGSVERFAHREN (AIFB), KIT

Agenda

Allgemeines

Dominik Bruhn

- Studiere Informatik am KIT mit Abschluss Diplom
- Studienarbeit betreut von Markus Klems

Studienarbeit

- Untersuchung von verschiedenen *NoSQL* Datenbank-Systemen
- Vergleich und Kategorisierung um Entscheidungen zu vereinfachen

Einleitung

Das Problem:

- Internet Unternehmen benötigen Zugriff auf **riesige Datenmengen**.
- Das **Skalieren** der Datenbank-Systeme ist schwierig.
- Der **”One-Size-Fits-All”** Ansatz der RDMS funktioniert nicht.
- RDMS **garantieren zu viel**.

Einleitung

Das Problem:

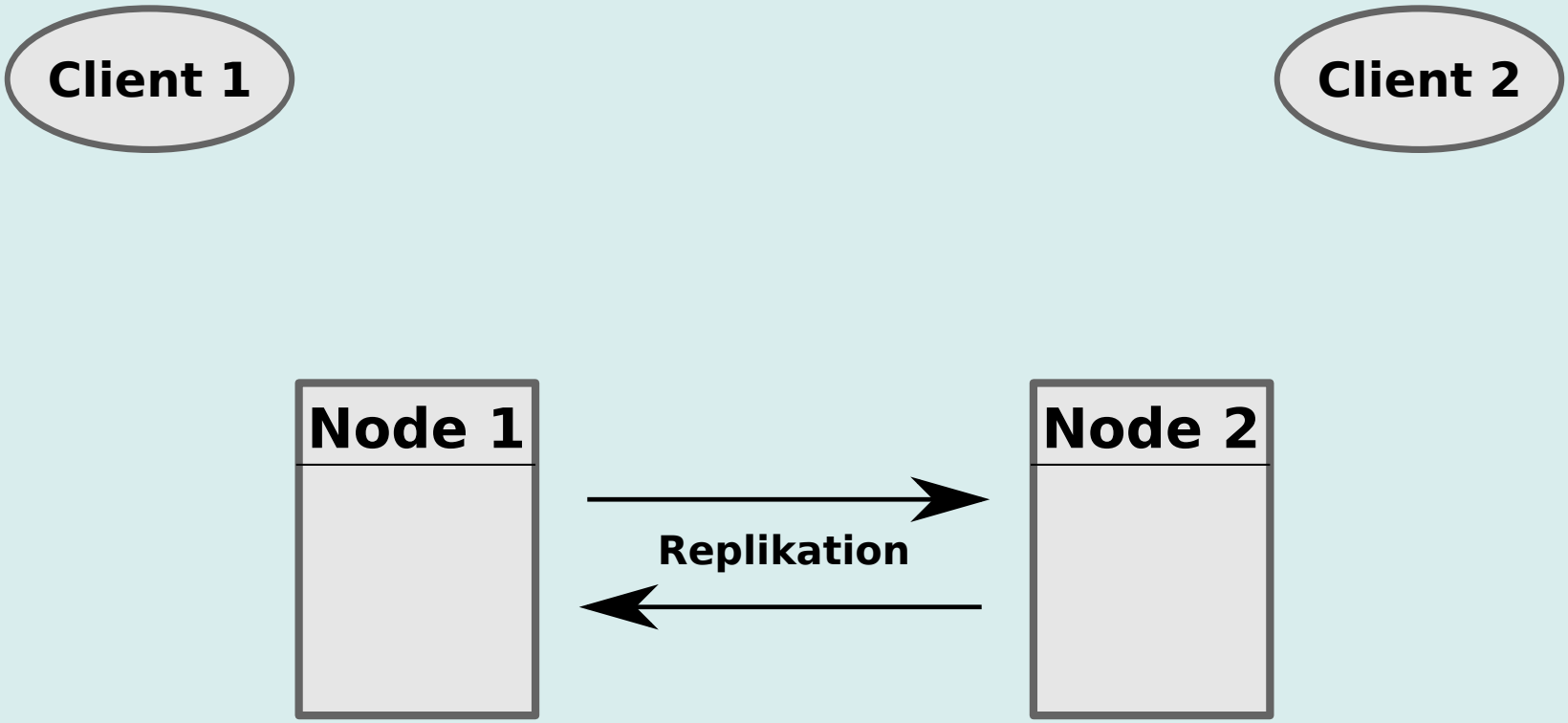
- Internet Unternehmen benötigen Zugriff auf **riesige Datenmengen**.
- Das **Skalieren** der Datenbank-Systeme ist schwierig.
- Der ”**One-Size-Fits-All**” Ansatz der RDMS funktioniert nicht.
- RDMS **garantieren zu viel**.

Die Lösung:

- Große Anzahl von Datenbank-Systemen, die nicht mehr dem RDMS Ansatz folgen (*NoSQL*)
- Unterschiedlichste Funktionen, Einsatzzwecke, Lizenzen und Programmiersprachen
- Sehr unterschiedliche Konzepte in Bezug auf Skalierbarkeit

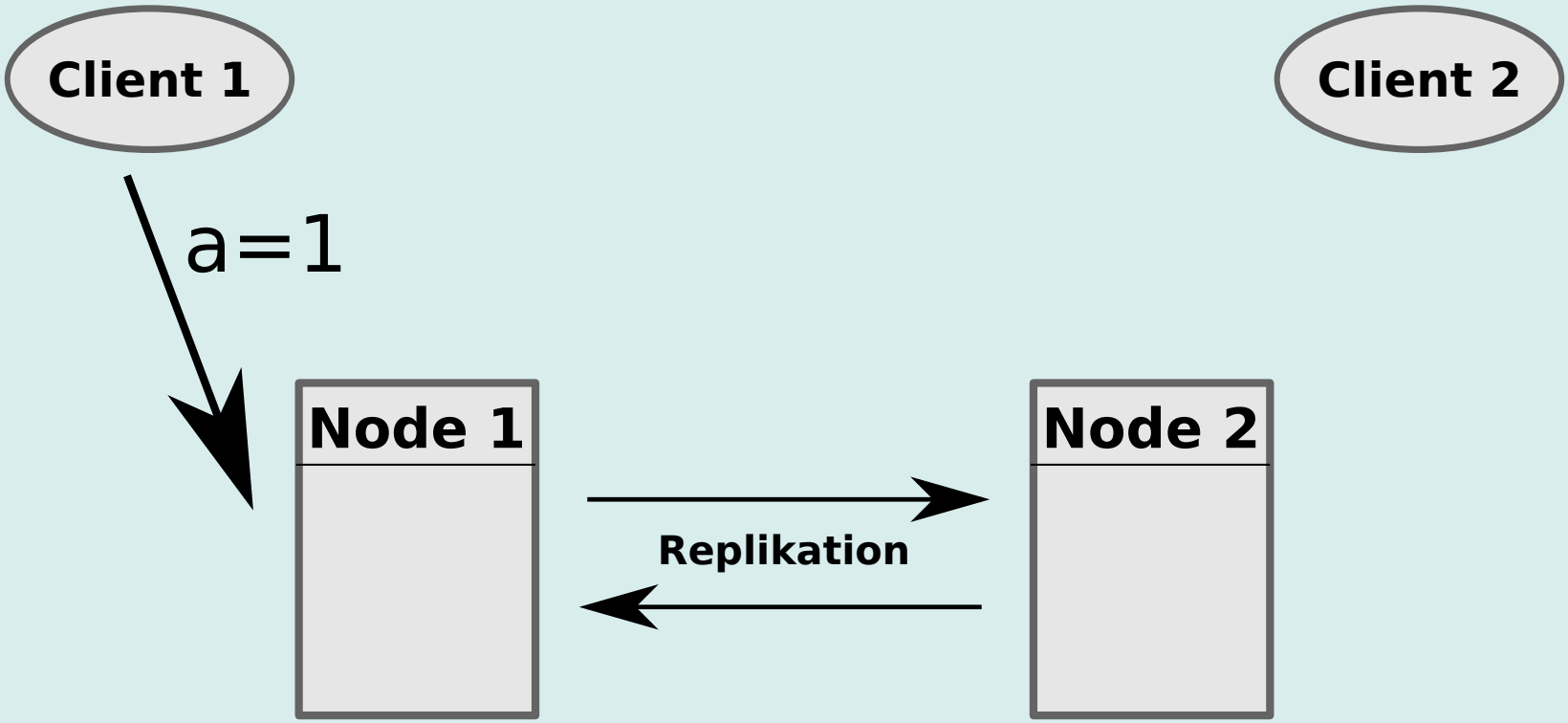
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



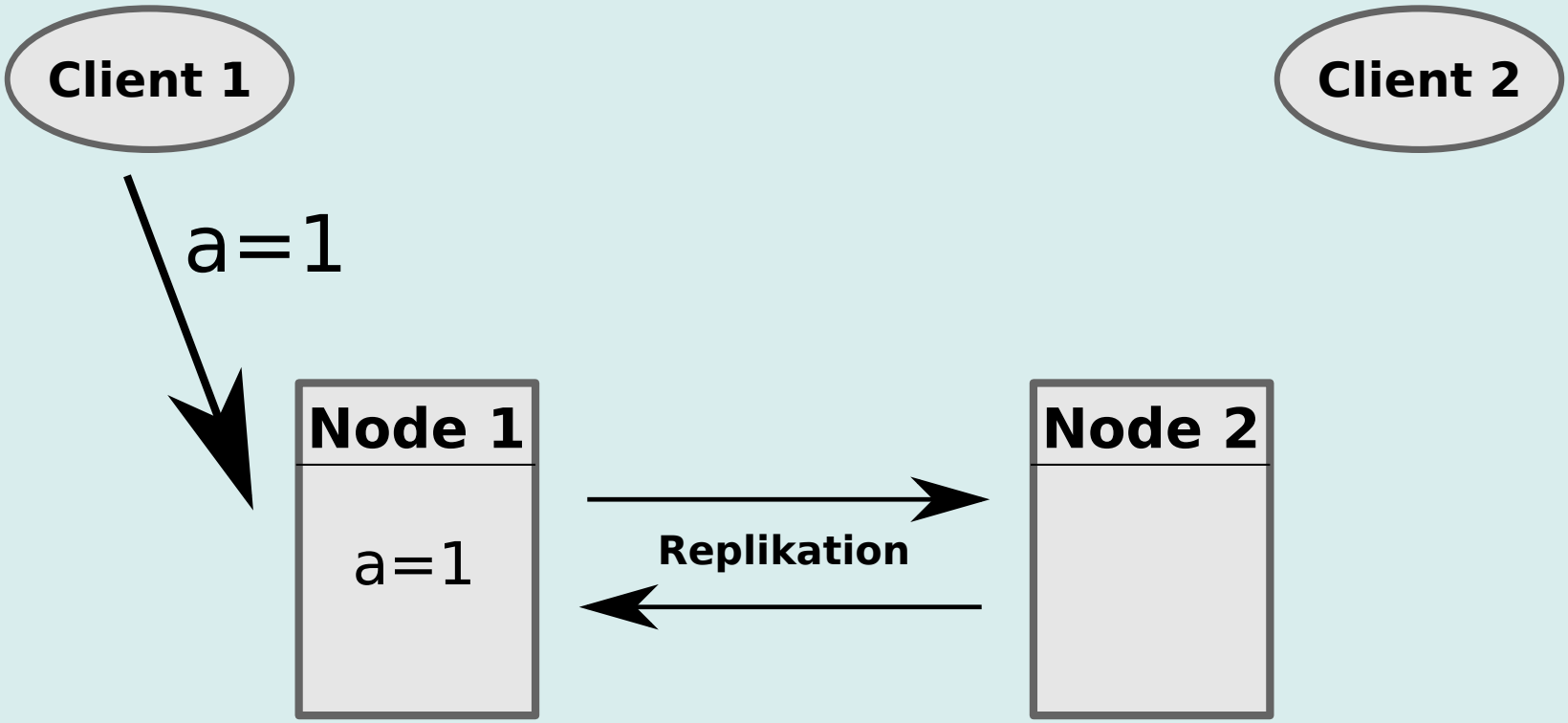
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



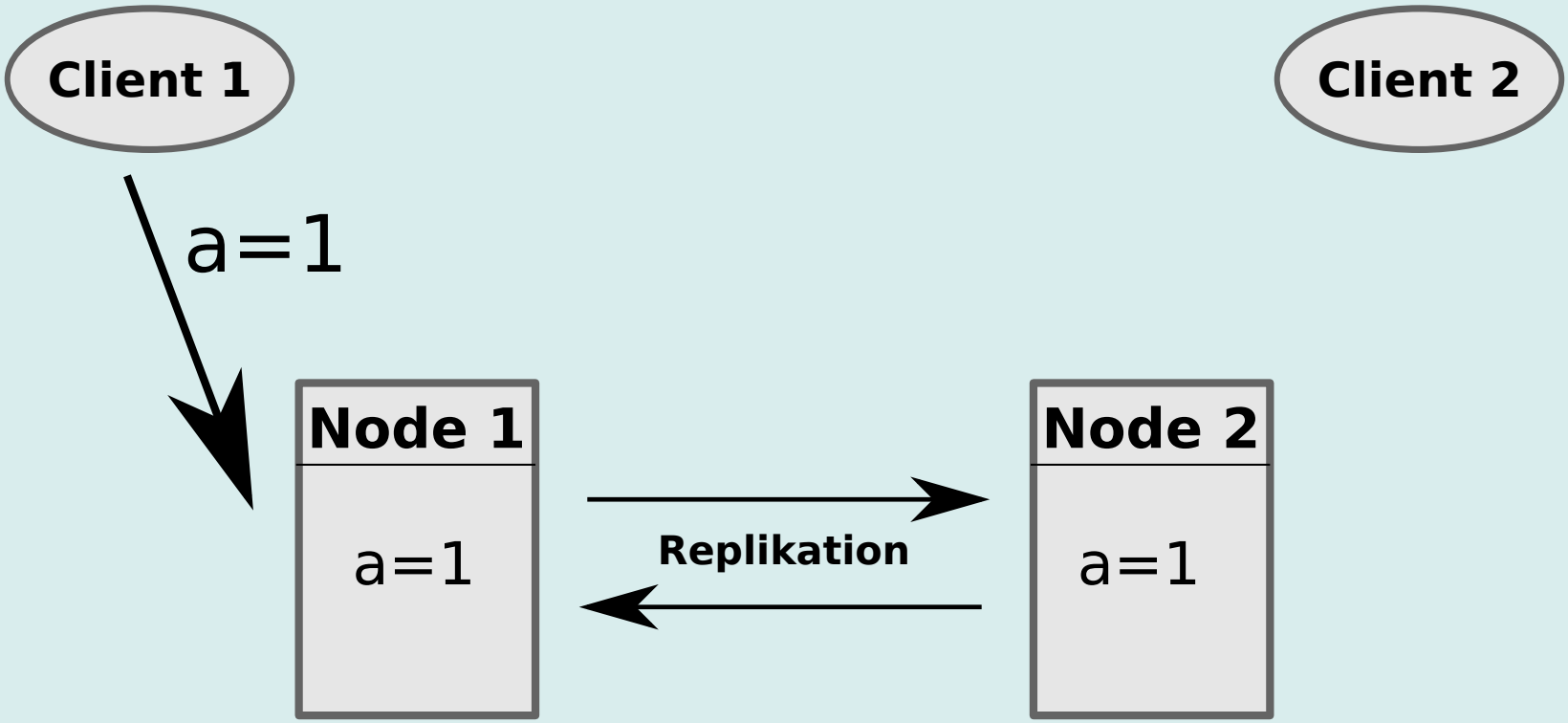
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



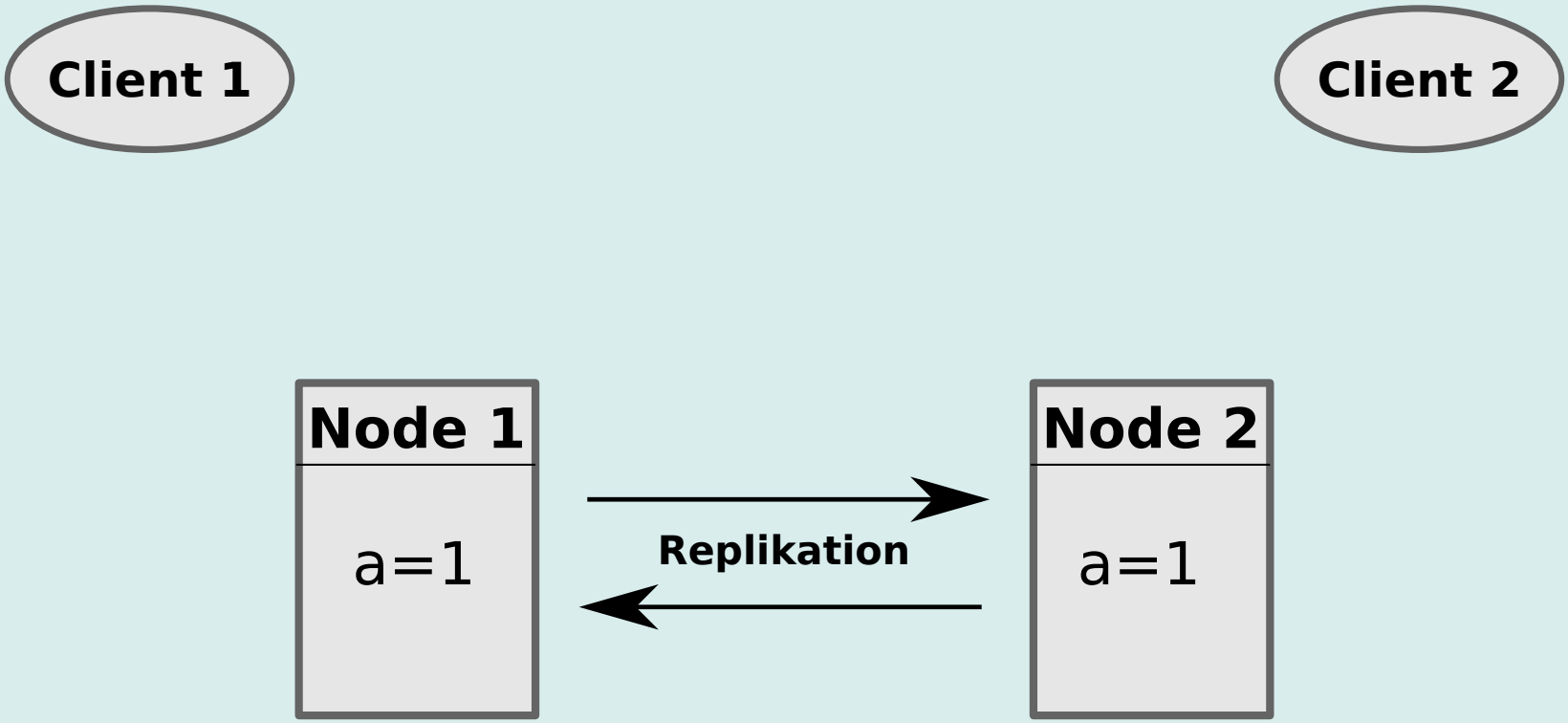
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



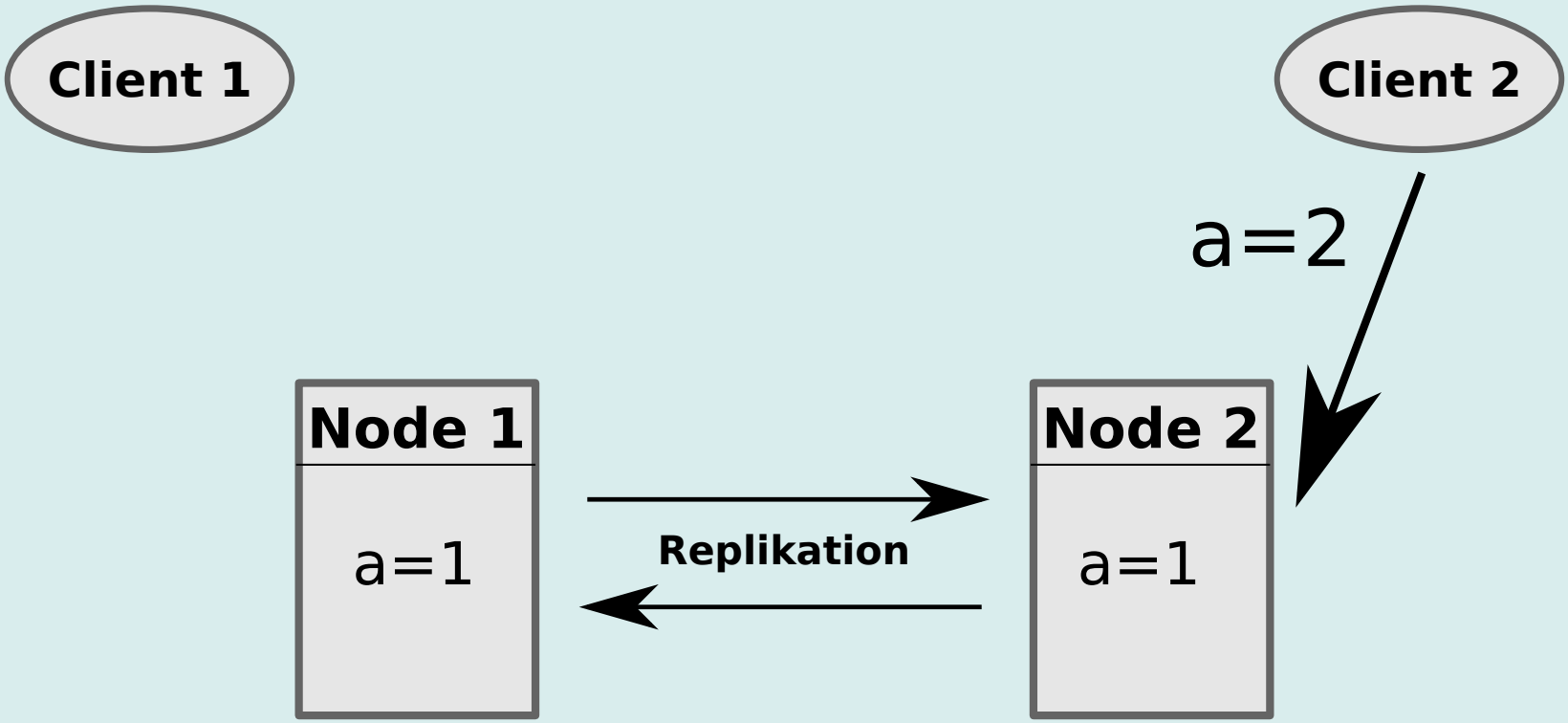
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



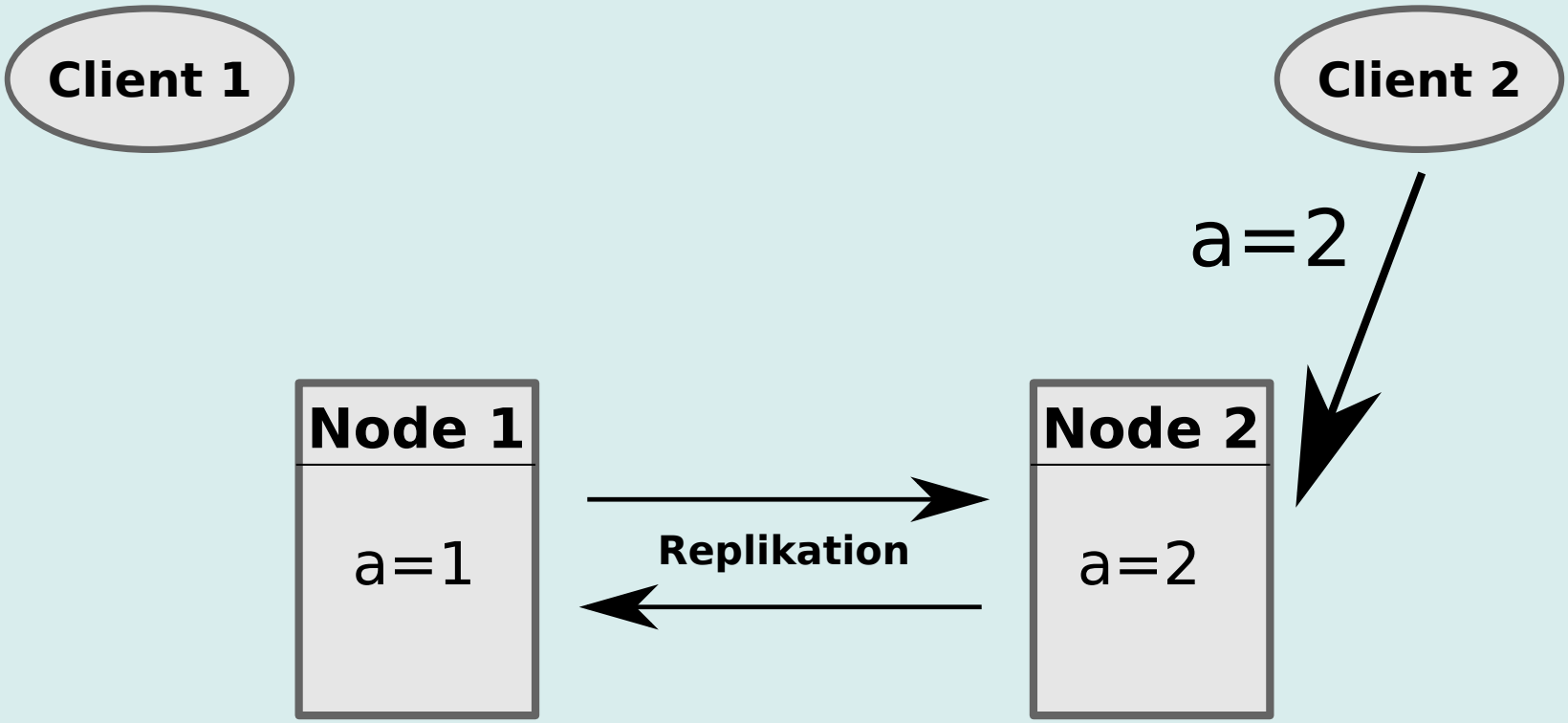
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



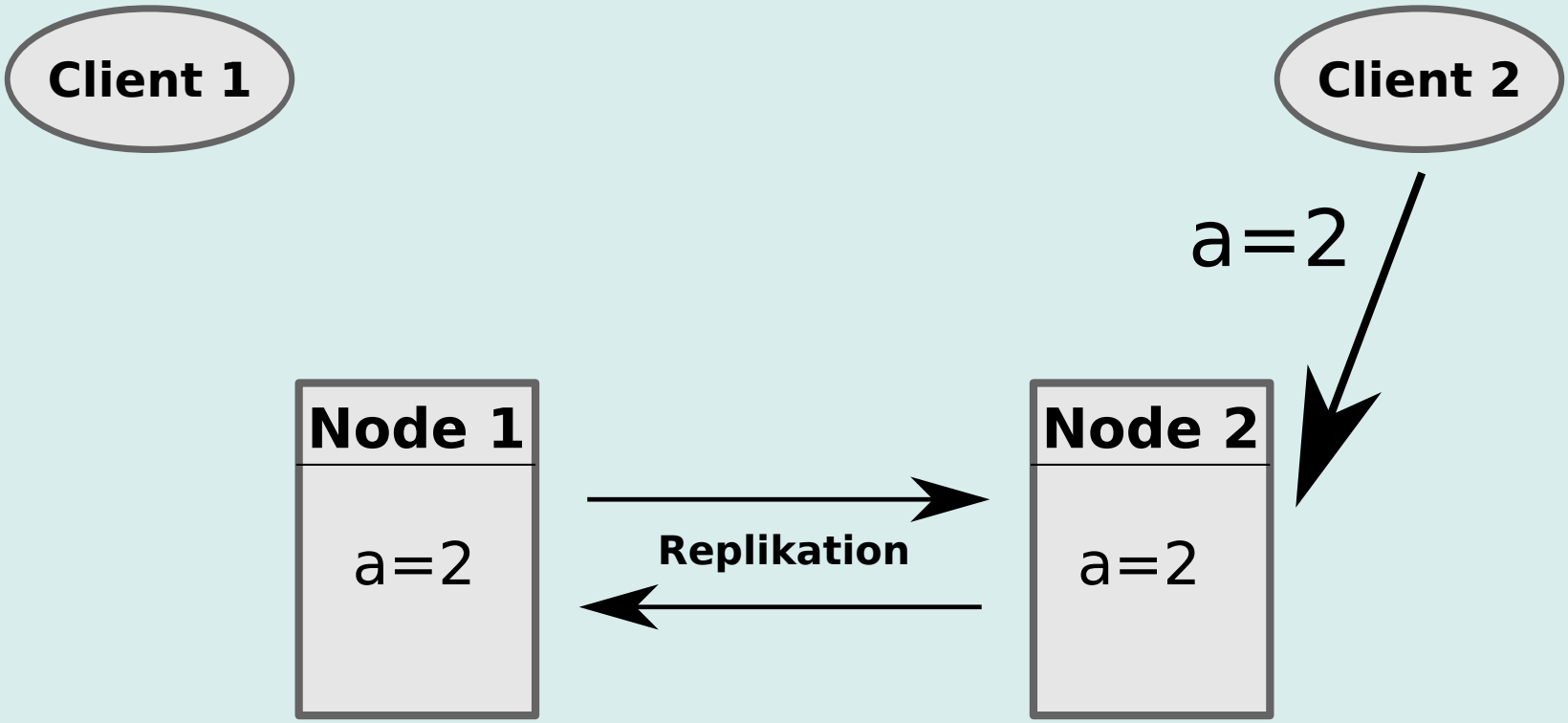
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



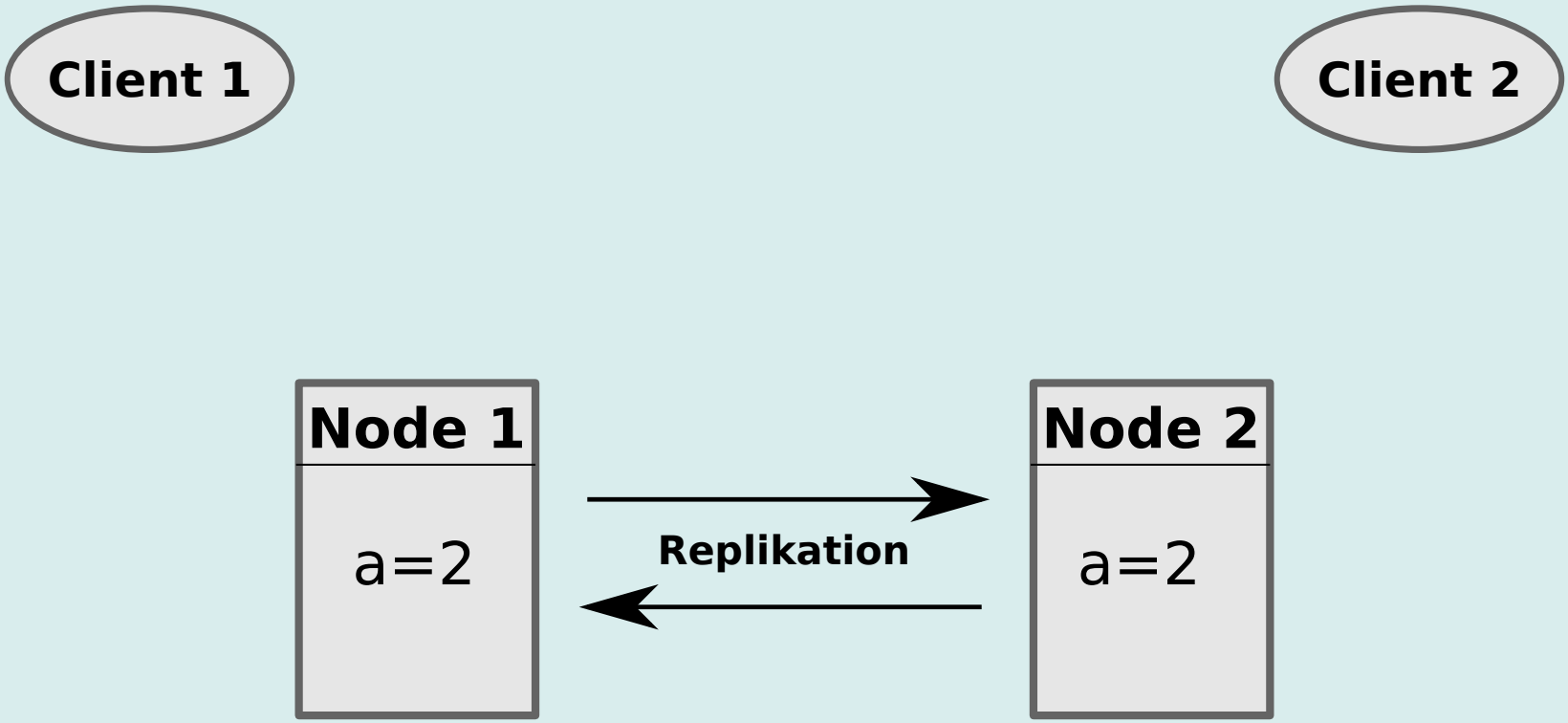
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



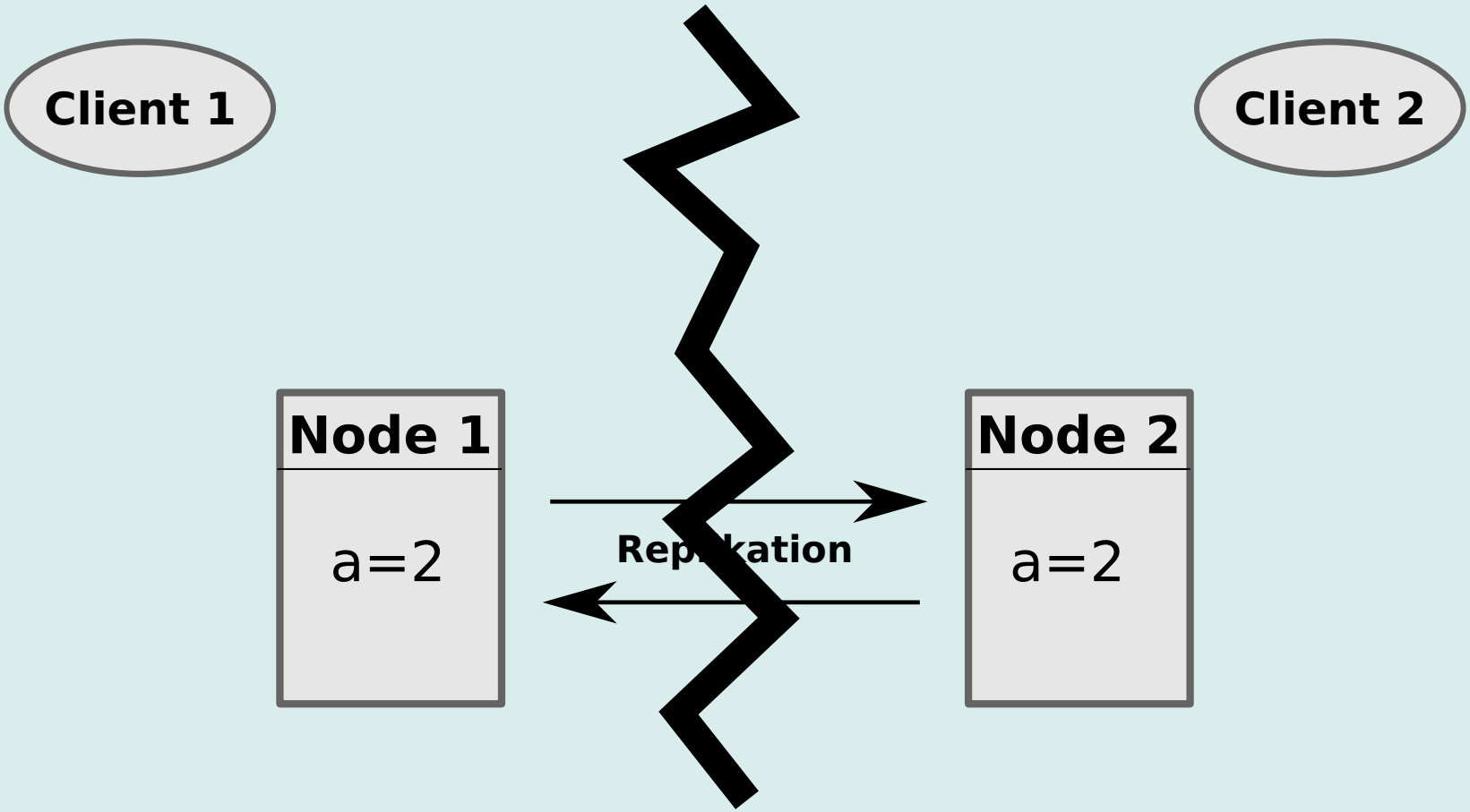
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



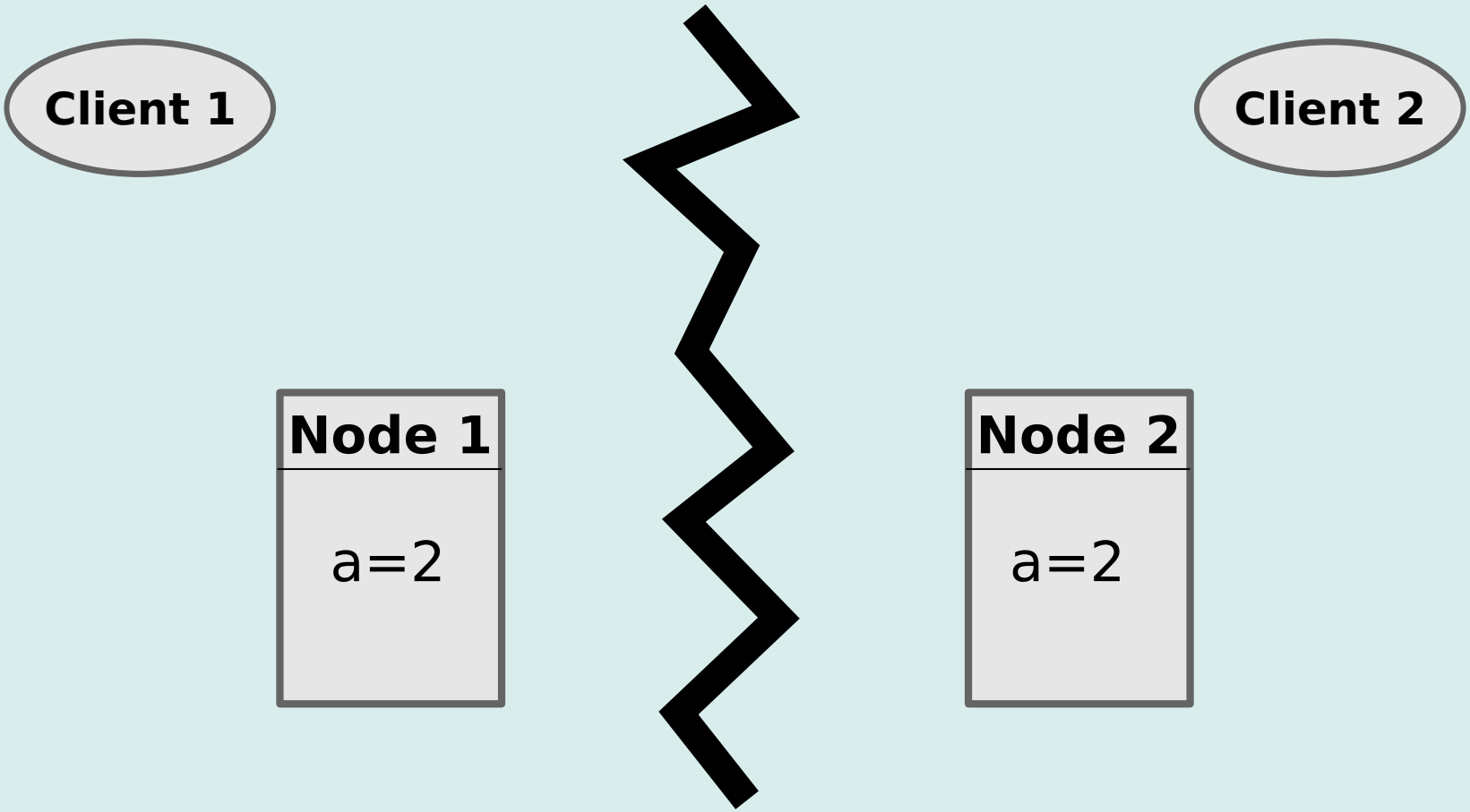
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



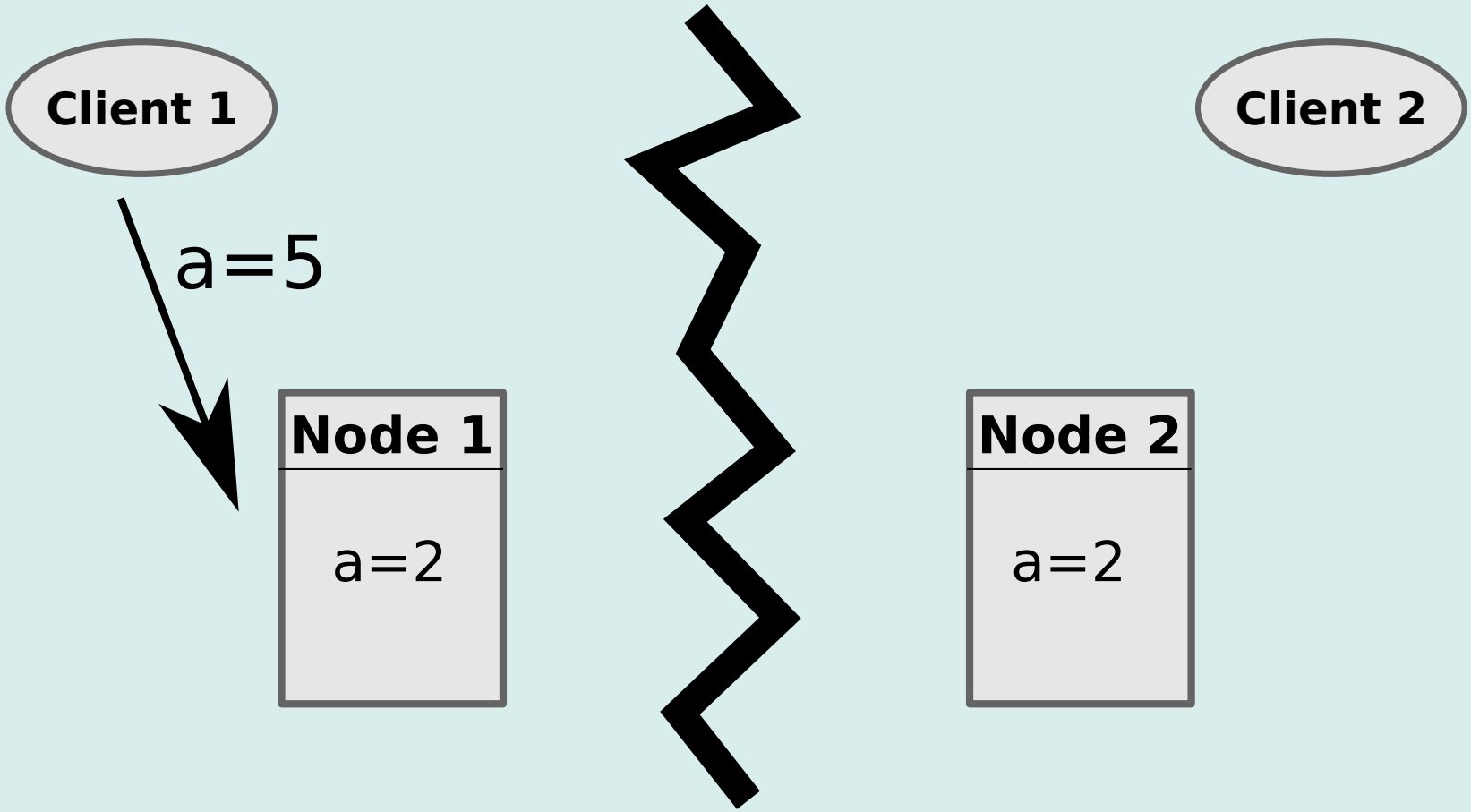
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



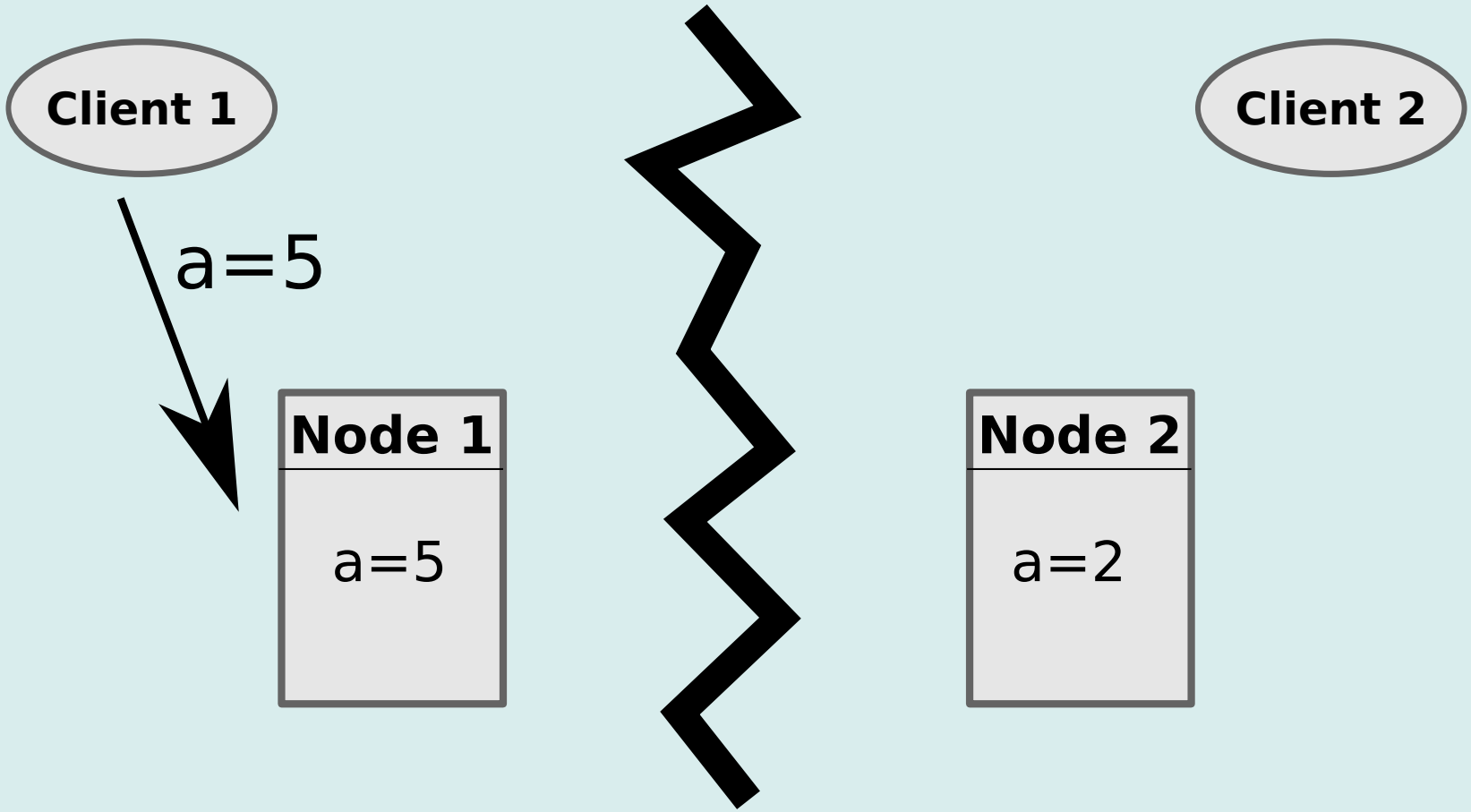
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



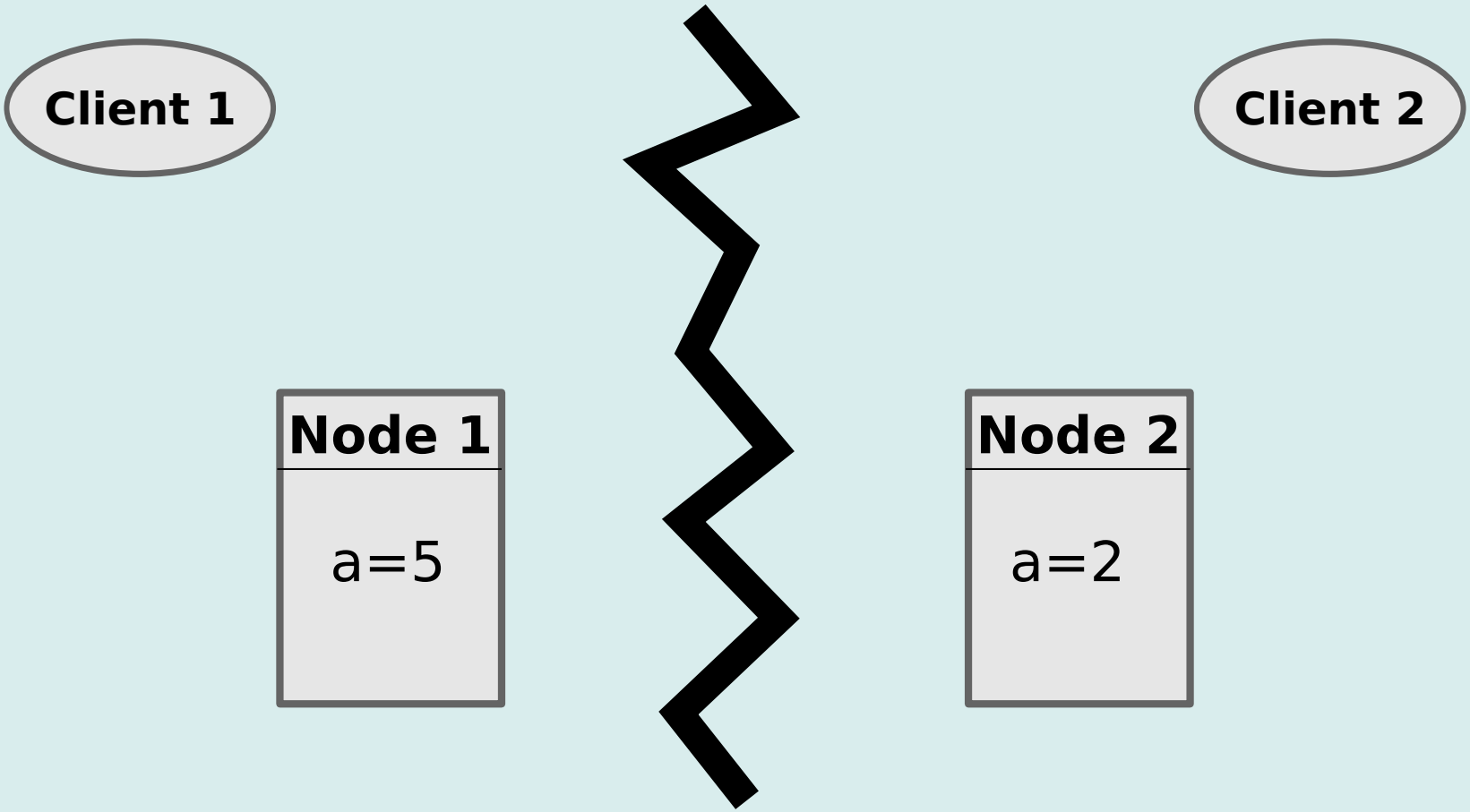
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



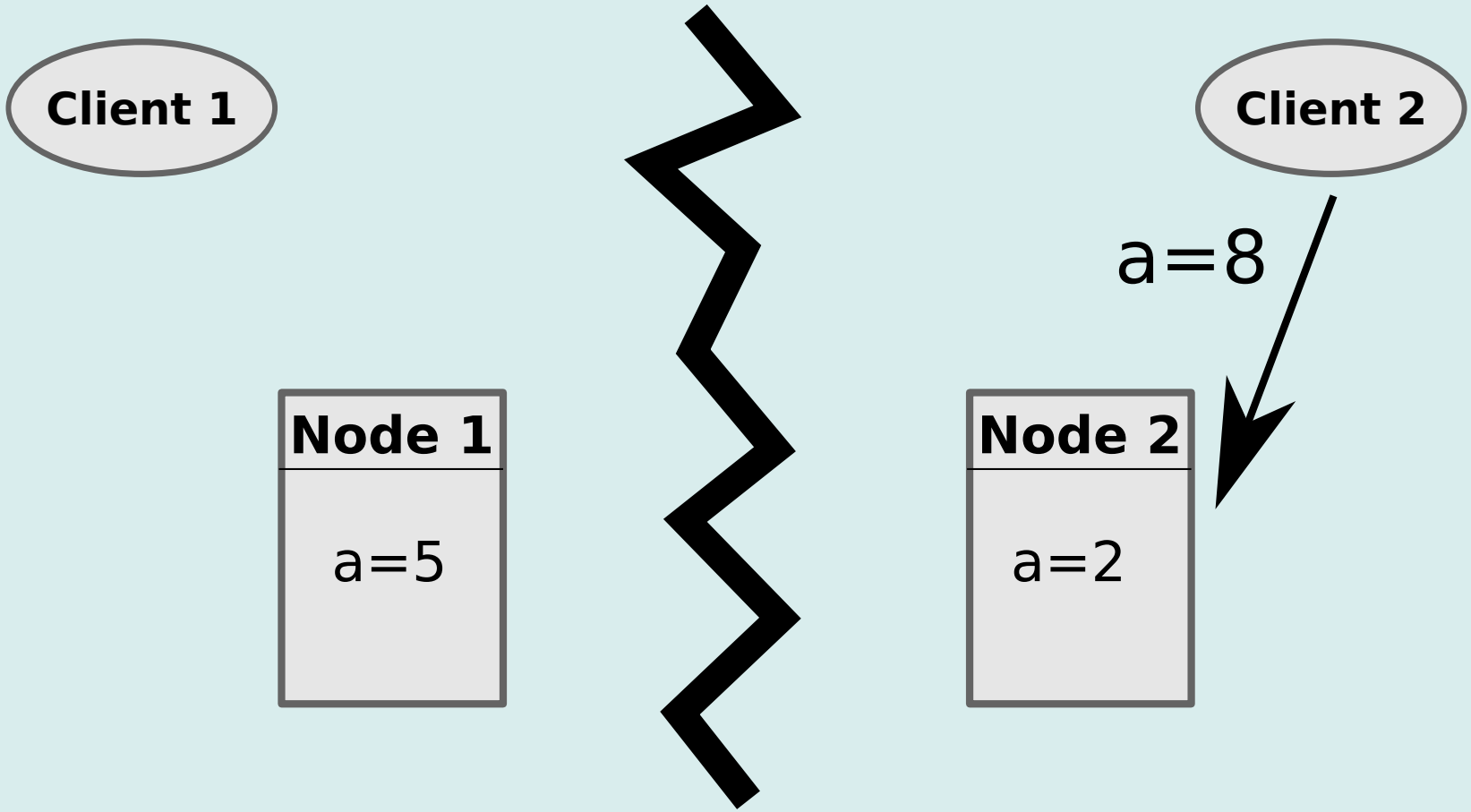
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



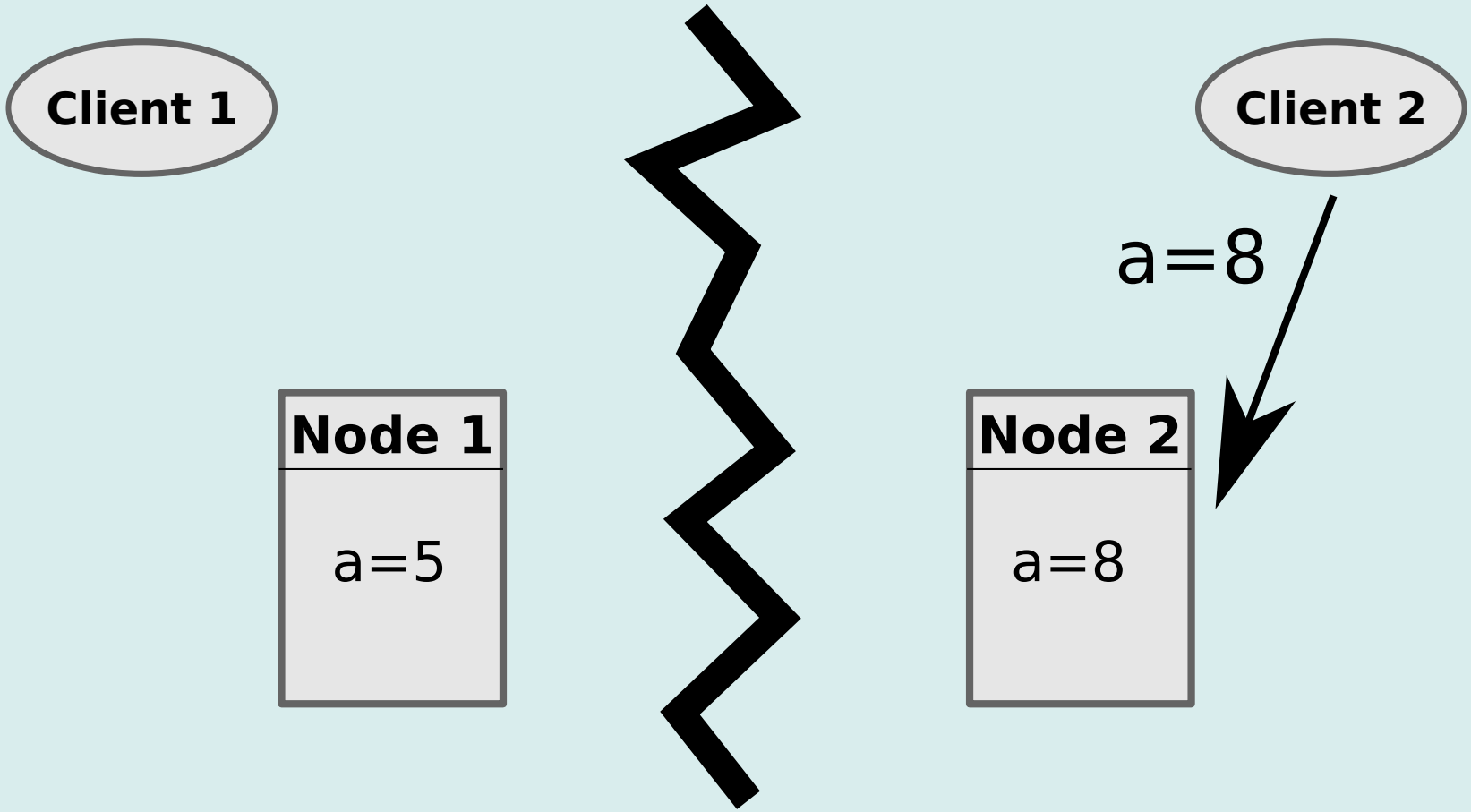
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



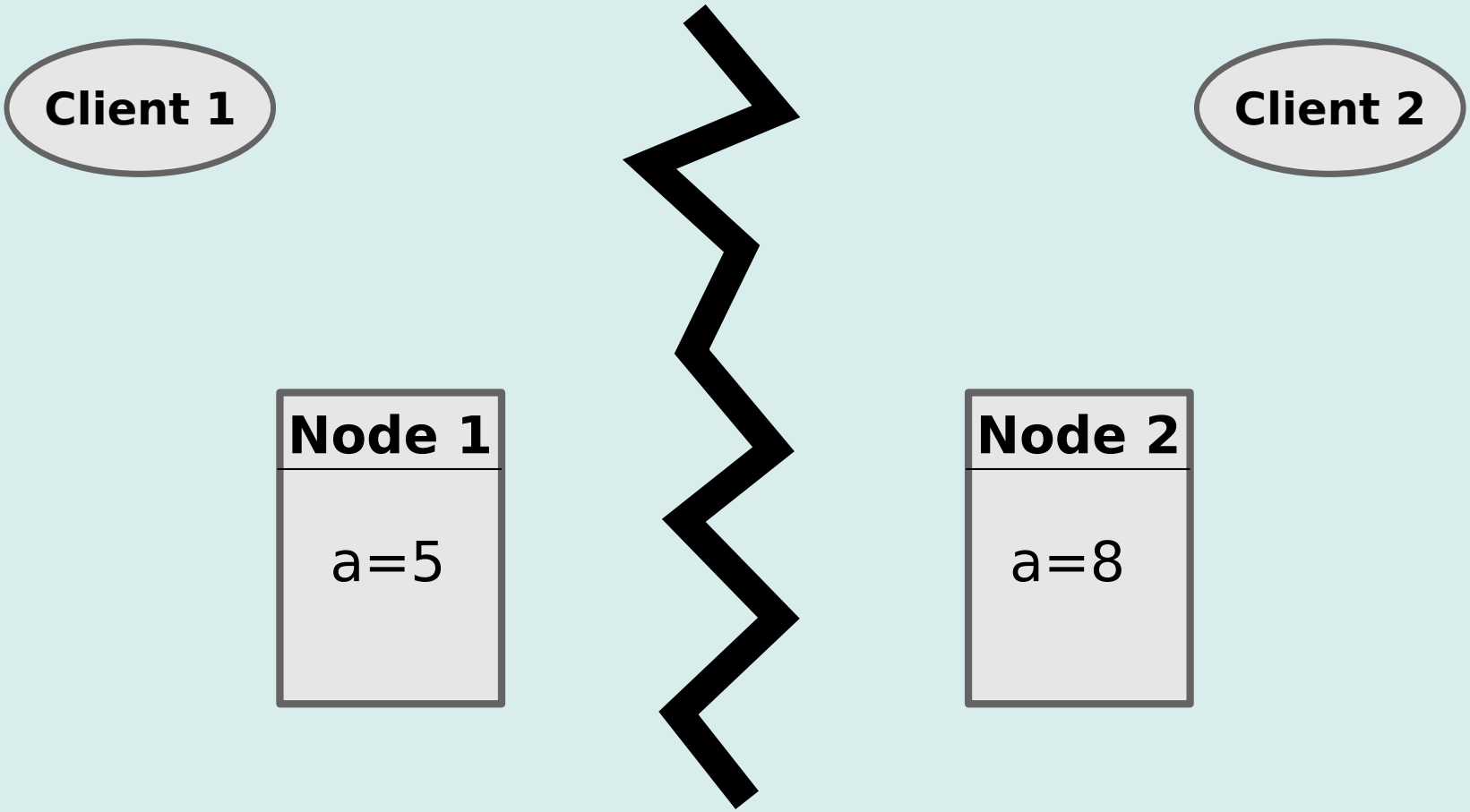
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



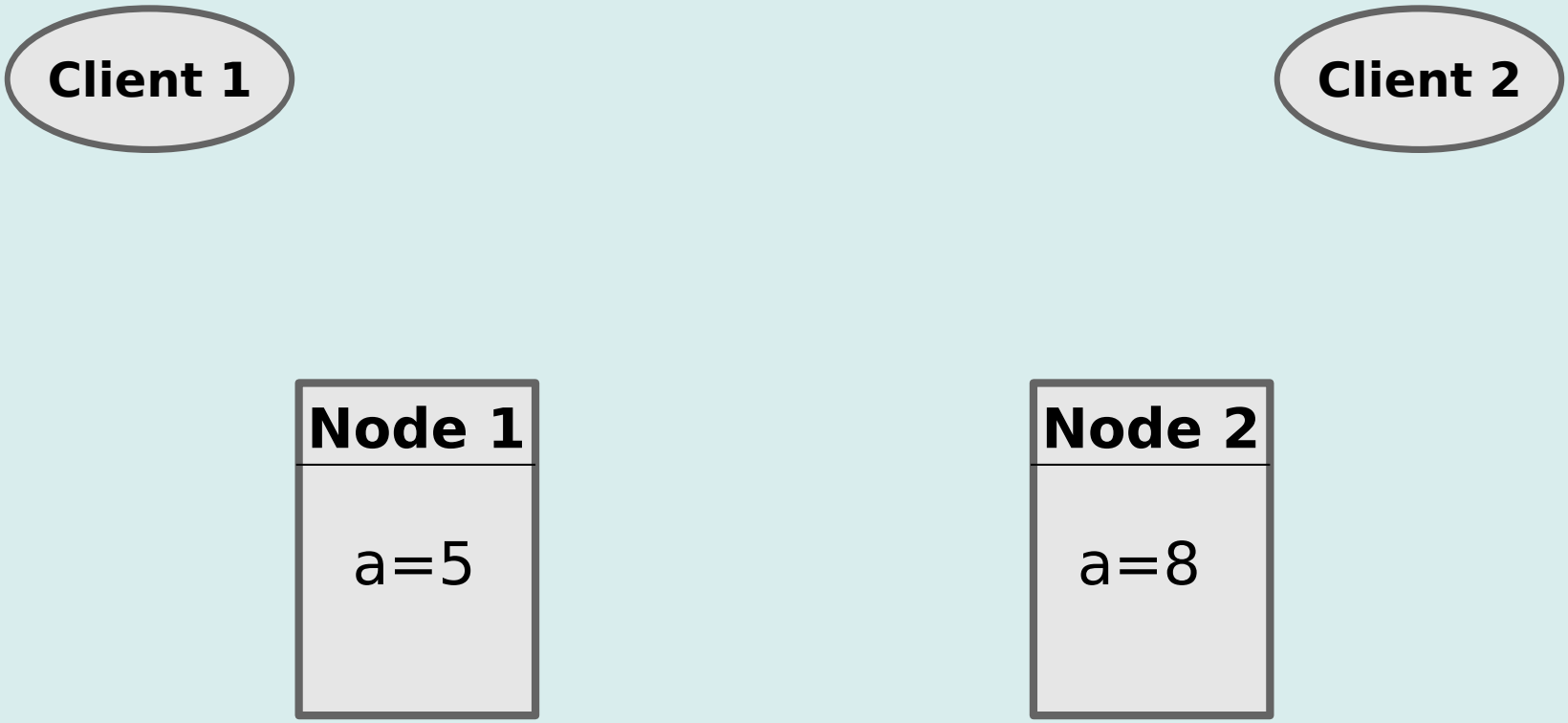
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



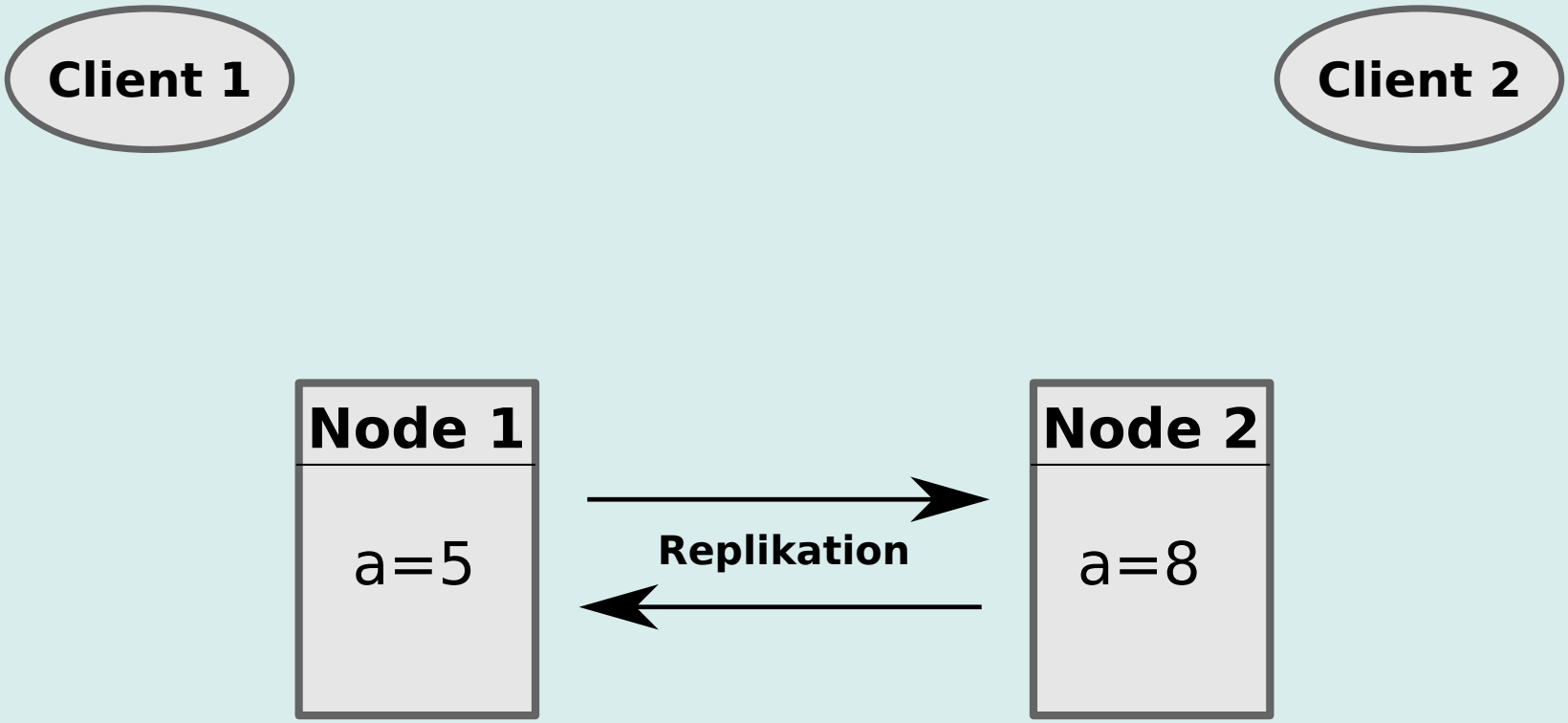
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



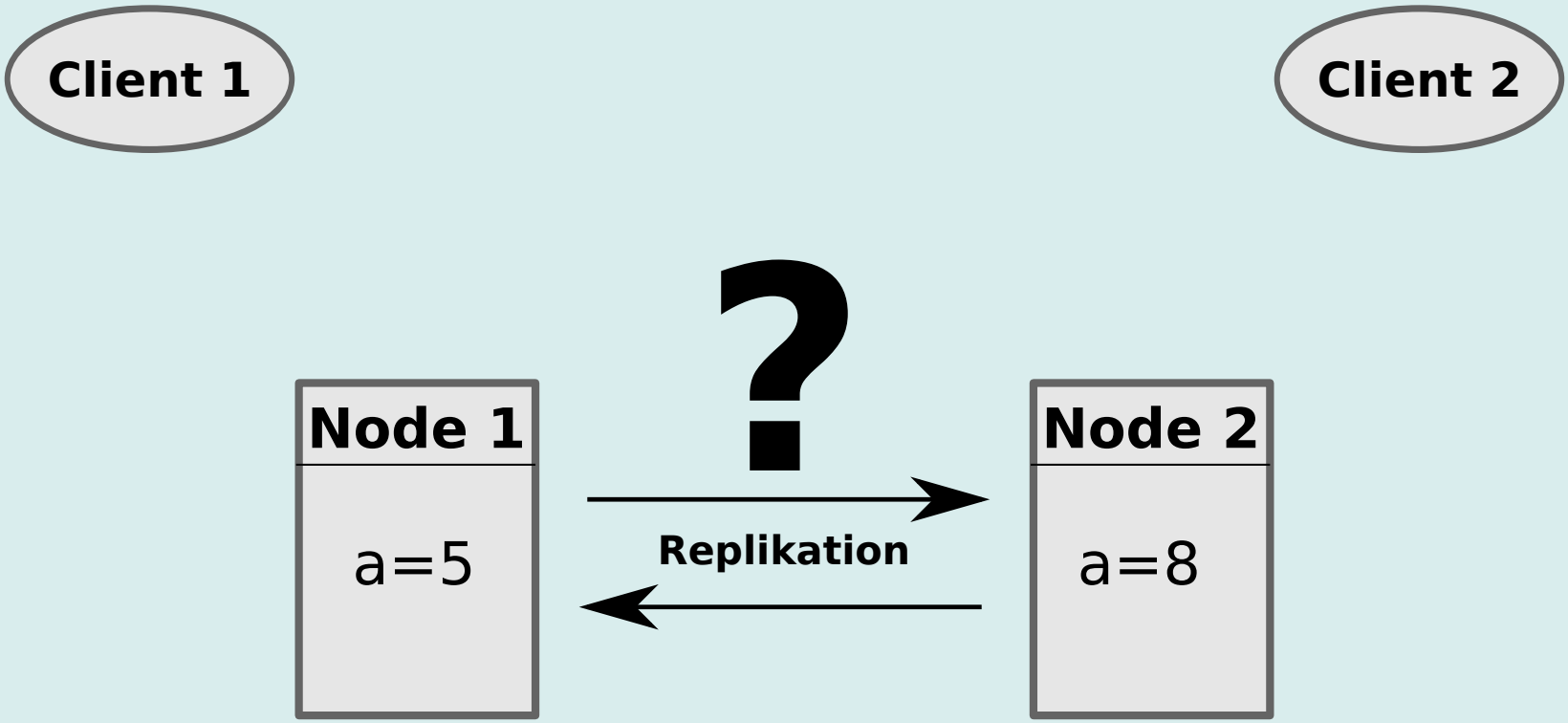
Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



Konsistenz Probleme

Inkonsistenzen, falls das Netzwerk in Partitionen zerfällt



Gruppierung

Acht *NoSQL* Datenbank-Systeme wurden untersucht und in drei Kategorien eingeteilt:

Ring

Project Voldemort

Master-Slave

Asynchr. Replikation

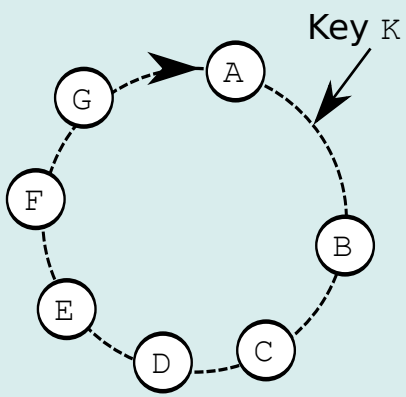
Redis

Ring

Eigenschaften

- Anordnung der Knoten auf einem Ring
- Dezentralisiert: Knoten sind gleich und beantworten jede Anfrage
- Kopien der Zeilen werden auf den Knoten verteilt
- Ausfälle unproblematisch

Ring Struktur

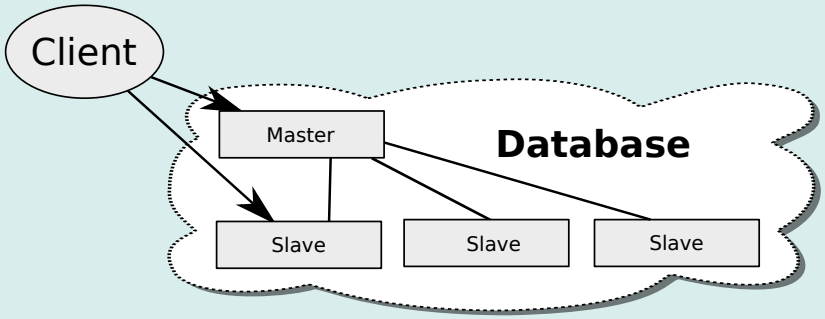


Variable Konsistenz

Für jede Schreib- und Lese-Operation kann ausgewählt werden, wie viele Kopien synchron geschrieben oder gelesen werden sollen. Der Rest wird asynchron im Hintergrund abgearbeitet (und kann damit fehlschlagen).

Master-Slave

Architektur



Aufgaben

Master: Zuordnung Zeile → Slave, Metadaten

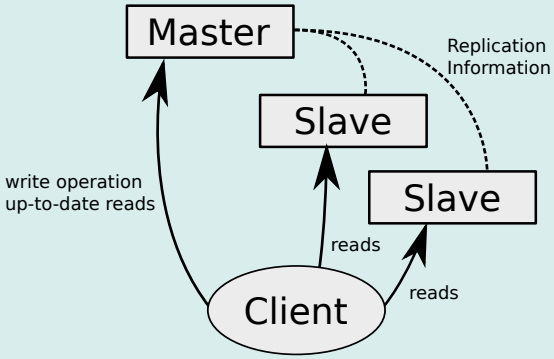
Slave: Eigentlichen Zeilen

Eigenschaften

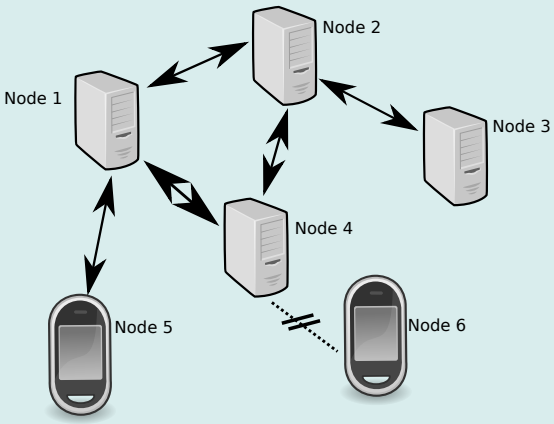
- Genau ein Slave ist für jede Zeile zuständig
- (Synchrone) Replikation der Slaves
- Strikte Konsistenz
- Probleme: Netzwerk-Fehler, Master-Ausfall

Asynchrone Replikation

Unidirektional



Bidirektional



Eigenschaften

- Unterscheidung: *unidirektionaler* und *bidirektionaler* Replikation
- Alle Knoten speichern kompletten Datensatz
- Asynchrone Replikation zwischen Knoten
- *Eventual Consistent*, aber immun gegen Netzwerkausfälle

Zusammenfassung

Zusammenfassung

- Spezielle Lösungen für spezielle Einsatzzwecke
- Kein *One-Size-Fits-All*
- Kategorisierung hilft bei der ersten Auswahl
- Große Unterschiede innerhalb der Kategorien abseits der Skalierungs Techniken (z.B. Daten-Model)
- Es konnten nicht alle *NoSQL* Datenbank-Systeme untersucht werden, der Markt ist in stetiger Bewegung.
- *You Are Not Facebook!*

Zusammenfassung

Zusammenfassung

- Spezielle Lösungen für spezielle Einsatzzwecke
- Kein *One-Size-Fits-All*
- Kategorisierung hilft bei der ersten Auswahl
- Große Unterschiede innerhalb der Kategorien abseits der Skalierungs Techniken (z.B. Daten-Model)
- Es konnten nicht alle *NoSQL* Datenbank-Systeme untersucht werden, der Markt ist in stetiger Bewegung.
- *You Are Not Facebook!*

Weiteres

Vielen Dank für die Aufmerksamkeit. Fragen?