Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences
Labor für Allgemeine Informatik

Kooperativer Speicher: Schwächen und Gegenmaßnahmen

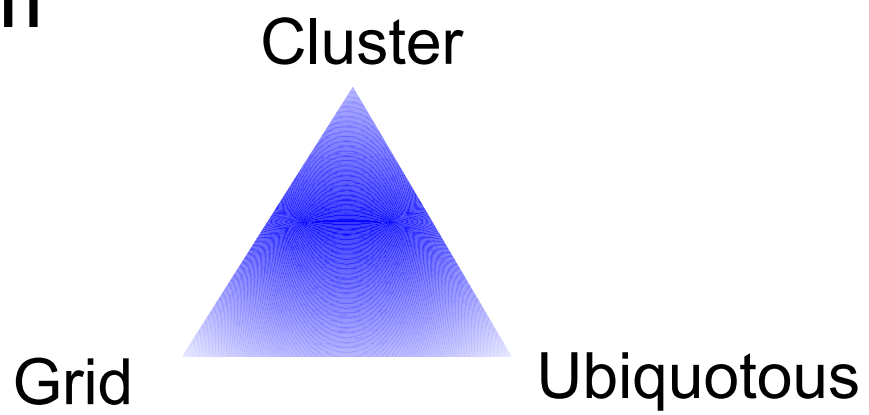
Cooperative storage: weaknesses and countermeasures

Lutz Behnke
2. Dezember 2005

Übersicht

- Was ist kooperative Speicherung und wer braucht sie?
- Existierende Lösungsansätze
- Offene Probleme
- Inhalt und Ziel meiner Arbeit

Das Problem



- Forderung nach Flexibilisierung von Remote Storage
  - Verfügbarkeit
  - Skalierbarkeit
  - Ad-Hoc/Discovery
  - Organisatorische Transparenz

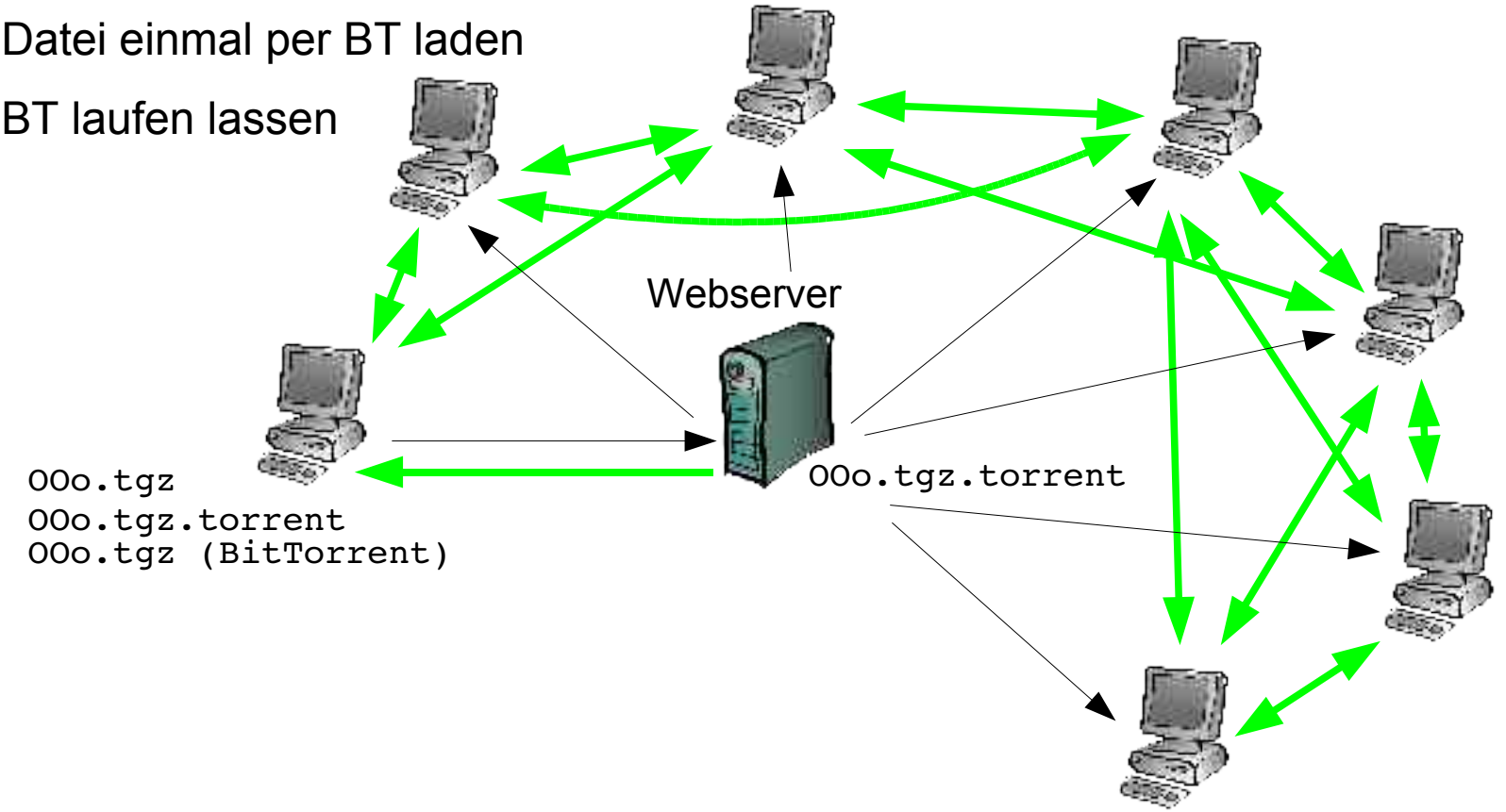
Lösung: Kooperative Speicherung

Jede Form der Datenspeicherung, bei welcher der Anwender keine oder unvollständige Kontrolle über das Speichermedium ausübt

- Ad-Hoc Konfiguration von 'Communication Comunities' [1]
- Anwendungen ohne/mit minimaler Benutzerschnittstelle
- Operative/Organisatorische Anforderungen
  - Forderung von End-to-End Encryption (E2EE)
- Outsourcing

BitTorrent: Peer-2-Peer gesellschaftsfähig

1. Tracker auf irgendeinem Webserver (Es gibt öffentliche)
2. Die `.torrent` Datei auf eigene Webseite
3. Datei einmal per BT laden
4. BT laufen lassen



Unterschiede zu klassischen Verfahren

- Netzwerk-Dateisystem (NFS,CIFS,Coda,AFS,....)
  - Bieten keine/eingeschränkte Discovery
  - skalieren nicht transparent
  - Rechteverwaltung ist Server-Zentriert
- Datenbank
  - Datenbank legt Struktur anhand von Schema *apriori* fest.
  - Coop-Storage ist auf Bulk-Daten fokussiert. Die werden auch in RDBMs oft separat in `text`/BLOb Bereichen abgelegt.

Anforderungen

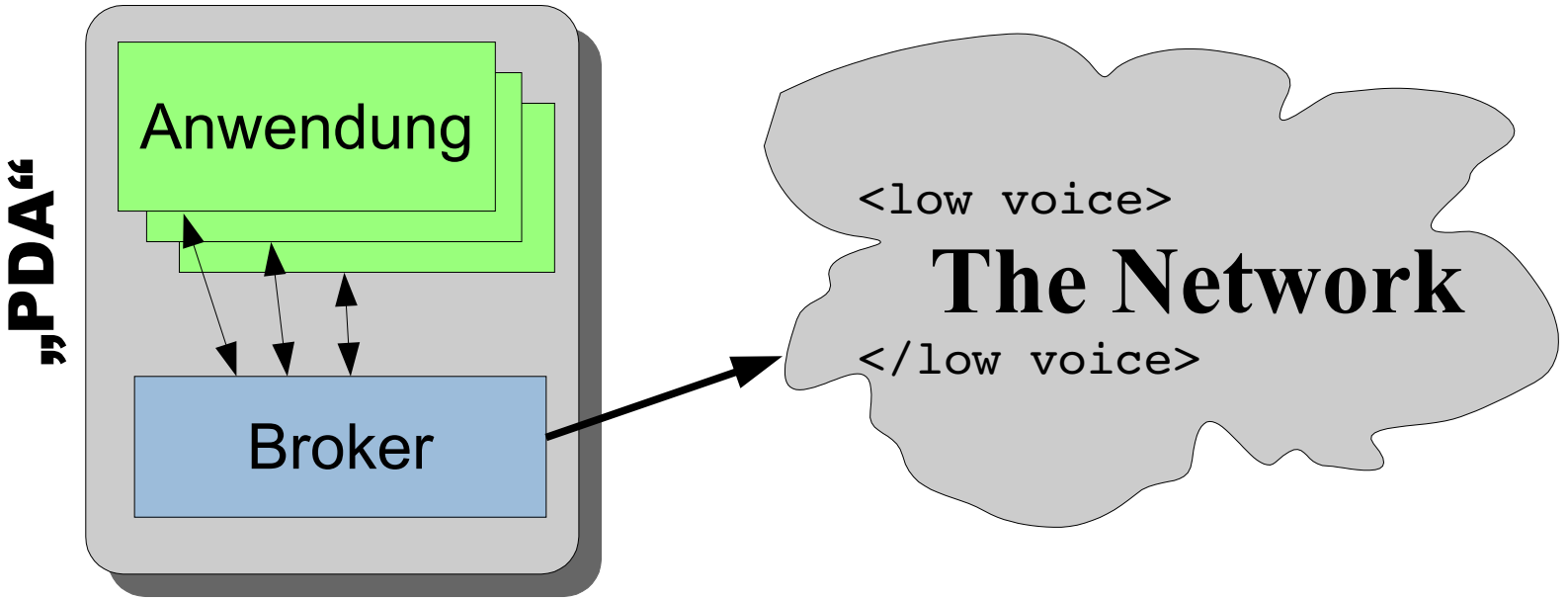
- Discovery
- Read-Write
- End-2-End Verschlüsselung
- Freshness Garantie
  - Data Freshness
  - Access Permission Revocation
- Skalierbarkeit
- Stabilität gegen Datenverlust

Lösungsansätze

- Cooperative File System (Oxigen/MIT)
  - ☺ P2P-Basis, Ad-Hoc, robust, skalierbar
  - ☹ Read-only, keine End-to-End Verschlüsselung (E2EE)
- GoogleFS, Lustre (Google, LLNL/Cluster FS Inc.)
  - ☺ Cluster FS, massiv skalierbar, redundate Speicherung, Meta-Daten Trennung
  - ☹ Keine (Meta-Daten) Server Discovery, keine E2EE
- SiRiUS (Stanford)
  - ☺ Unterschiedliche Backends (NFS,CIFS,Yahoo!)
  - ☹ Replikation durch Backend, Keine Server Discovery

Vision

- Simpelste API
  - `FileHandle sendData(byte[] data)`
  - `byte[] getData(FileHandle handle)`
- Alles andere macht die Middleware!



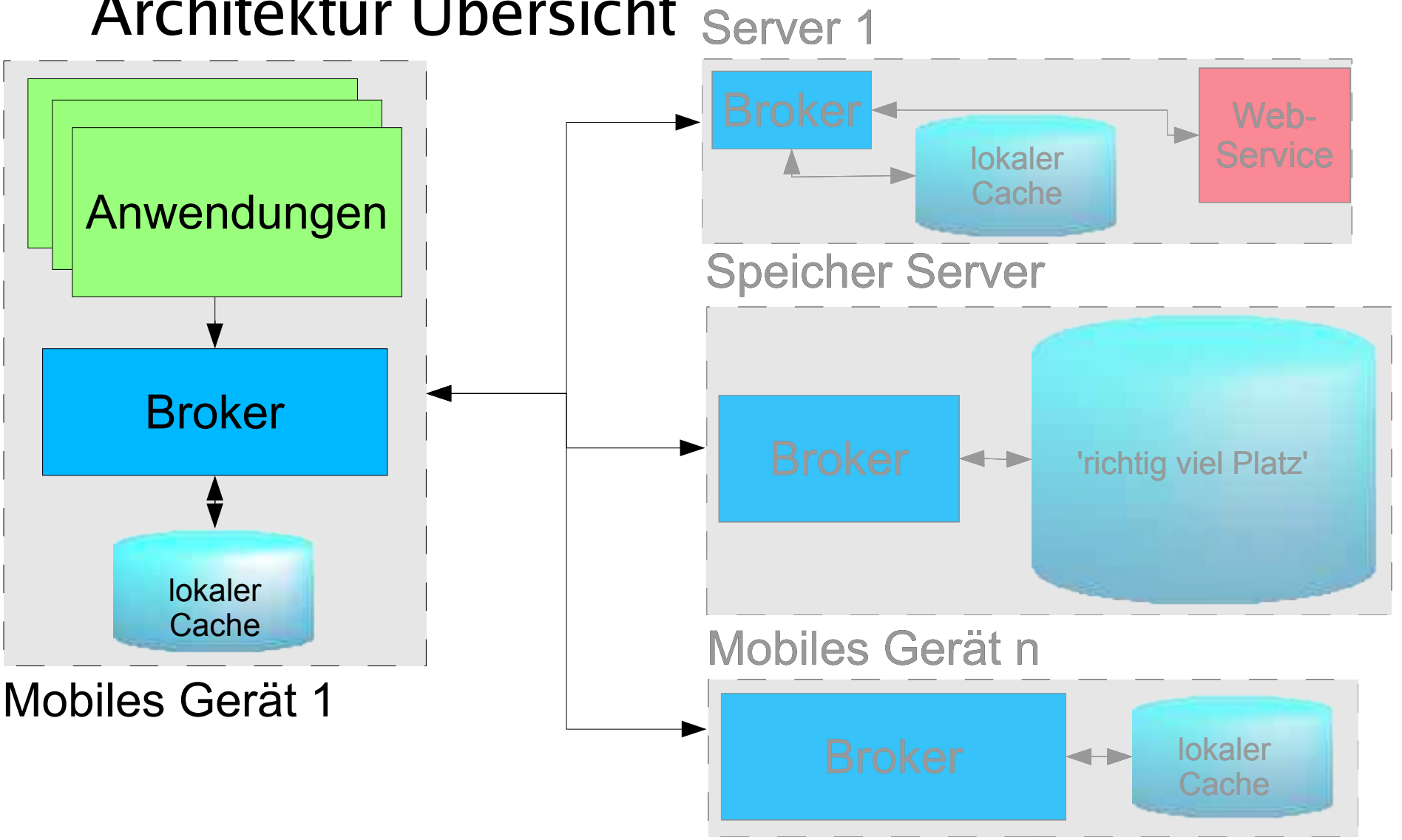
Ziele

- Client-Komponente soll auf abstraktem mobilem Geräten laufen
- Daten werden abgespeichert
- Vollständige Discovery
  - Geräte, Resourcen, Konfigurationen
- End-2-End-Encryption
- Annahmen
  - Crypto existiert (oder wird ignoriert)
  - API für Netz-Discovery existiert

Herangehen für Master-Arbeit

- Wotan: Testplattform für Kooperative Speicherung
  - Ergebnis des Projektes WS05
  - Abstrakte Schnittstelle für Speicherung von Daten
- Zusammenstellung von Fehlerformen für Kooperative Speicherung
- Implementation von Lösungen
  - Discovery
  - End-2-End Verschlüsselung

Architektur Übersicht



Methodik

- Evolutionäres Prototyping
  - Test-driven
  - volle Build/Test Automatisierung
  - mindestens ~80% Code Coverage in Unit Tests
- Client besteht aus Library und Broker
  - Interface für Library zum ersten Meilstein eingefroren
  - Selten Updates der Implementation möglich.
- geplanter Status Feb '06
  - Prototyp in Java vorhanden (no Savety, no Security)

Risiken

- Gewähltes Verfahren liefert keine zufriedenstellenden Ergebnisse bezüglich
  - Stabilität
  - Antwortverhalten
  - Komplexität der Handhabung
- Risikobeherschung:
  - Simple Plattform: Der µPC von heute ist der PDA von morgen
  - Simple Anforderung: UI und Handhabung sind nicht meine Baustelle.

Bibliographie (Web)

[1]Projekt Oxigen (MIT) http://www.oxygen.lcs.mit.edu

- Netzwerk Anforderungen: http://www.oxygen.lcs.mit.edu/Network.html

[2]Stanford Secure Computer Systems Group http://www.scs.stanford.edu

[3]Weitere Dokumente und Veröffentlichungen finden sich in der schriftlichen Ausarbeitung