Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Verteilte Dateisysteme

Mykhaylo Kabalkin

Übersicht

- Einleitung
  - Was ist ein Dateisystem?
  - Was ist ein verteiltes Dateisystem?
- Einige bekannte Systeme
  - Network File System
  - Andrew File System
  - BitTorrent
  - Google FS
  - Lustre FS
- Einordnung ins Project
- Quellen

Übersicht

- Einleitung
  - Was ist ein Dateisystem?
  - Was ist ein verteiltes Dateisystem?
- Einige bekannte Systeme
  - Network File System
  - Andrew File System
  - BitTorrent
  - Google FS
  - Lustre FS
- Einordnung ins Project
- Quellen

Was ist ein Dateisystem?

- Ein Dateisystem (File System): Ordnungs- und Zugriffssystem für Daten, die auf einem Datenträger gespeichert sind

- Bestandteil eines Betriebssystems
  - Betriebssystem ist ebenfalls in einem Dateisystem gespeichert

Was ist ein Dateisystem?

- Linux/Unix
  - Ext3 : third extended file system
  - Ext4 : fourth extended file system (Oktober 2006), seit Version 2.6.19 offizieller Bestandteil des Linux-Kernels
  - NFS : Network File System von Sun Microsystems. Ein „Protokoll", das den Zugriff auf Dateien über ein Netzwerk ermöglicht
  - Berkeley Fast File System wird in verschiedenen BSD-Derivaten (FreeBSD, NetBSD und OpenBSD) sowie in Solaris und NextStep verwendet

Was ist ein Dateisystem?

- Microsoft
  - File Allocation Table (FAT12, FAT16 und FAT 32)
  - Virtual FAT (VFAT), Unterstützung von längeren Dateinamen
  - New Technology File System (NTFS) z.Z. NTFS 3.1
  - WinFS 

Was ist ein Dateisystem?

- Apple/Macintosh
  - Apple DOS – erste Apple Dateisystem
  - Hierarchical File System (HFS), mit Mac OS entwickelt

Was ist ein verteiltes Dateisystem?

- Eine ähnliche Aufgabe wie normale Dateisysteme in konventionellen Betriebssystemen
- Zugriff auf Dateien in anderen, entfernten Rechnern
- Wird zur Realisierung von anderen Diensten benutzt, z.B.
  - Persistentes Speichern für Transaktionssysteme
  - Namensserver, Authentication-Server usw.
- "A distributed file system enables programs to store and access remote files exactly as they do on local ones, allowing users to access files from any computer on the intranet."

  Bina Ramamurthy

Was ist ein verteiltes Dateisystem?

- Access Transparency (dispersion of users)
  - Jeder Benutzer kann von jedem beliebigen Host auf Dateien zugreifen
- Concurrency Transparency (multiplicity of users)
  - Mehrere Benutzer können gleichzeitig auf dieselben Dateien zugreifen
- Location Transparency und Location Independence (dispersion of files)
  - Dateien eines Benutzers können auf verschiedenen Hosts sein
- Replikation Transparency (multiplicity of files)
  - Dieselben Dateien können im System mehrfach repliziert vorliegen

Was ist ein verteiltes Dateisystem?

- Migration Transparency
  - Wird eine Datei im System verlagert, so soll das den Zugriff nicht beeinträchtigen
- Failure Transparency
  - Das Dateisystem muss korrekt arbeiten trotz des möglichen Ausfalls von Servern oder des Verlustes von Nachrichten
- Performance Transparency
  - Trotz variierender Lasten am File-Server muss der Client mit genügender Geschwindigkeit arbeiten
- Skalierbarkeit
- Hardware Heterogenität

Übersicht

- Einleitung
  - Was ist ein Dateisystem?
  - Was ist ein verteiltes Dateisystem?
- **Einige bekannte Systeme**
  - **Network File System**
  - **Andrew File System**
  - **BitTorrent**
  - **Google FS**
  - **Lustre FS**
- Einordnung ins Project
- Quellen

Network File System

- Bereits 1985 von Sun als erstes verteiltes Dateisystem eingeführt
- Seit 1989 steht die Spezifikation auch anderen Herstellern zur Verfügung
- Hauptziel – einen transparenten Zugriff auf Dateien in lokal verteilten Systemen zu gewährleisten
- NFS Version 4 Protocol Specification RFC 3530
- Auf jedem Rechner sowohl die Client- als auch die Server-Komponenten installiert (bei großen Installationen – einige Rechner als Server konfiguriert)

Network File System

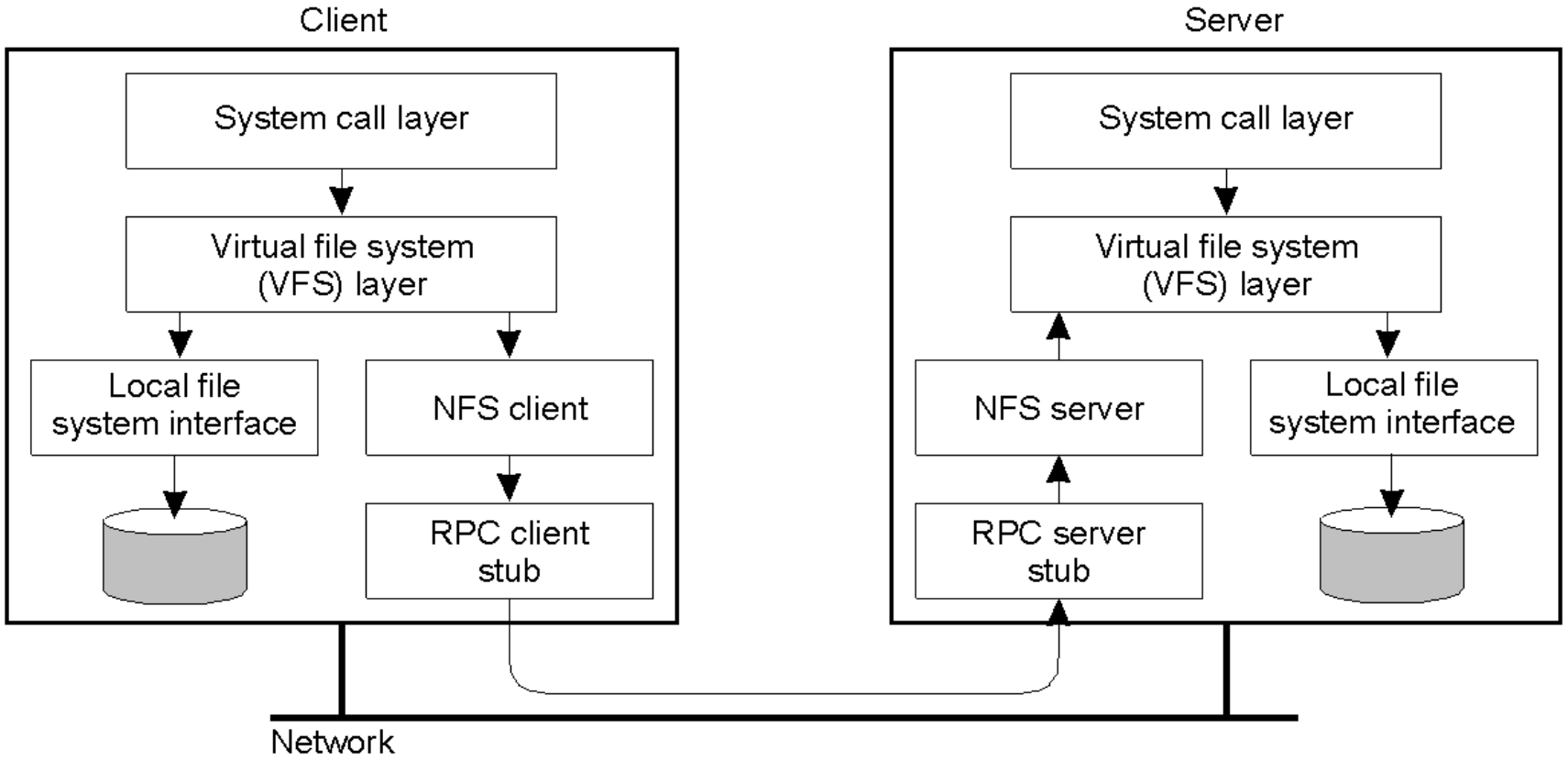
- Client-Server Architektur

- Server
  - Kann Verzeichnisse exportieren/freigeben
  - Verwaltet Zugriffsrechte
  - Zustandslos

- Client
  - Kann Verzeichnisse vom Server importieren (mounten)
  - Diese erscheinen dann wie lokale Verzeichnisse und können so genutzt werden
  - Die Dateien verbleiben bei Zugriff auf dem Server, wo auch die Operationen ausgeführt werden

Network File System

- NFS funktioniert über RPC (Remote Procedure Call)



The basic NFS architecture for UNIX systems

Network File System

- Keinen durch NFS erzwungenen netzwerkweiten eindeutigen Namen (Ortstransparenz nur durch zusätzliche Maßnahmen erreichbar)
- Kein dynamisches Migrations-Konzept
- NFS kann keine Konsistenz garantieren (z.B. neue Dateien können 30 sek. existieren, ohne dass andere Clients dies erfahren)
- Kein Replikationsmanagement
- Skalierbarkeit ist limitiert (Caching-Verfahren wurde nur für kleine lokale Netze entworfen)

Andrew File System

- Seit 1983 im Rahmen von Andrew Projekt an der Carnegie Mellon University, Pittsburgh, PA in Kooperation mit IBM (weiter-) entwickelt
- OpenAFS 1.5.13 vom 28.12.2006 www.openafs.org
- NFS war nicht ausreichend
- Hauptziel – Skalierbarkeit
  - Z.B. für 10.000 Rechner auf dem CMU Campus

Andrew File System

- Besitzt eine Ansammlung „vertrauenswürdiger" Server; alle anderen gelten als unsicher
- Server sind zustandsbehaftet
  - Z.B. werden Clients vom Server benachrichtigt, wenn sich Dateien, auf die sie gerade zugreifen, geändert haben
- Jeder Server kennt eine komplette Liste von (Haupt-) Verzeichnissen und deren Fundorten

Andrew File System

- Besonderheit ist die Caching-Methode
  - Es werden immer ganze Dateien auf die Client-Seite kopiert und dort im lokalen Dateisystem gecached
  - Erst wenn die Datei geschlossen wird, erfolgt die Aktualisierung der Server-Seite
  - Die Datei bleibt weiterhin auf der Client-Seite im Working-Set, bis wieder auf diese zugegriffen wird oder sie aus dem Working-Set verdrängt wird

Andrew File System

- The dice project, University of Edinburgh „A Comparison Between AFS and NFSv4"

- OpenAFS wird in mehreren Institutionen eingesetzt
  - Duce University Office of Information Technology, North Carolina, USA
  - Kungliga Tekniska Högskolan Elektro Department, Institut of Technology, Stockholm, Schweden
  - New Jersey Institute of Technology University Information Systems
  - Dr. Wilhelm Andre Gymnasium, Chemnitz, Deutschland

BitTorrent

- Von Bram Cohen entwickeltes File-Sharing Protokoll
- Geeignet für die Verteilung großer Dateien
- Skaliert gut
- Referenzimplementierung von B. Cohen in Python
- Red Hat, Ubuntu, FreeBSD Distribution Verteilung

BitTorrent

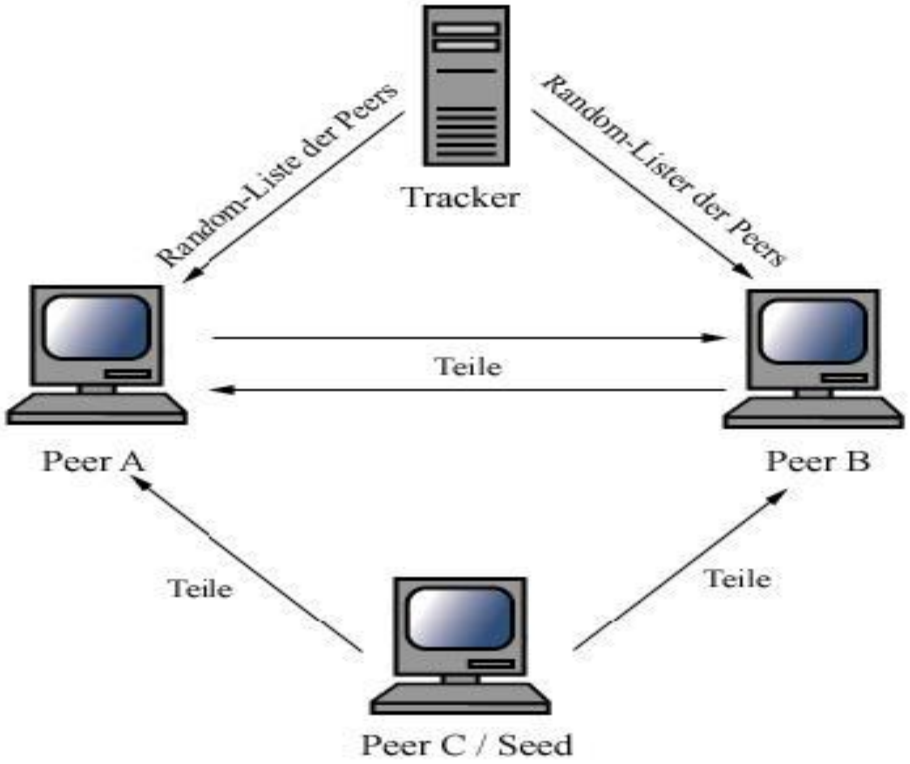
- Peers (Client)
  - Seed
    - Besitzt vollständige Datei und stellt sie zum Download bereit
    - Entweder der ursprüngliche Anbieter oder der, der nach dem kompletten Download einer Datei diese weiter zum Download zur Verfügung stellt
  - Leecher
    - Besitzt noch keine komplette Datei

BitTorrent

- Zentraler Tracker
  - Verwaltet Informationen zu einer oder mehreren Dateien
  - Gibt den Clients bekannt, wer noch die Datei herunterlädt oder verteilt
  - Sobald ein Client ein Segment (chunk) der Datei erhalten und die Prüfsumme verifiziert hat, meldet er dies dem Tracker und kann dieses Dateistück schon an andere Clients weitergeben

- Torrent-Datei - .torrent
  - Enthält alle wichtige Meta-Informationen (die Adresse des Trackers, Dateiname, Größe und Prüfsumme der herunterzuladenden Datei)

BitTorrent



BitTorrent Architektur

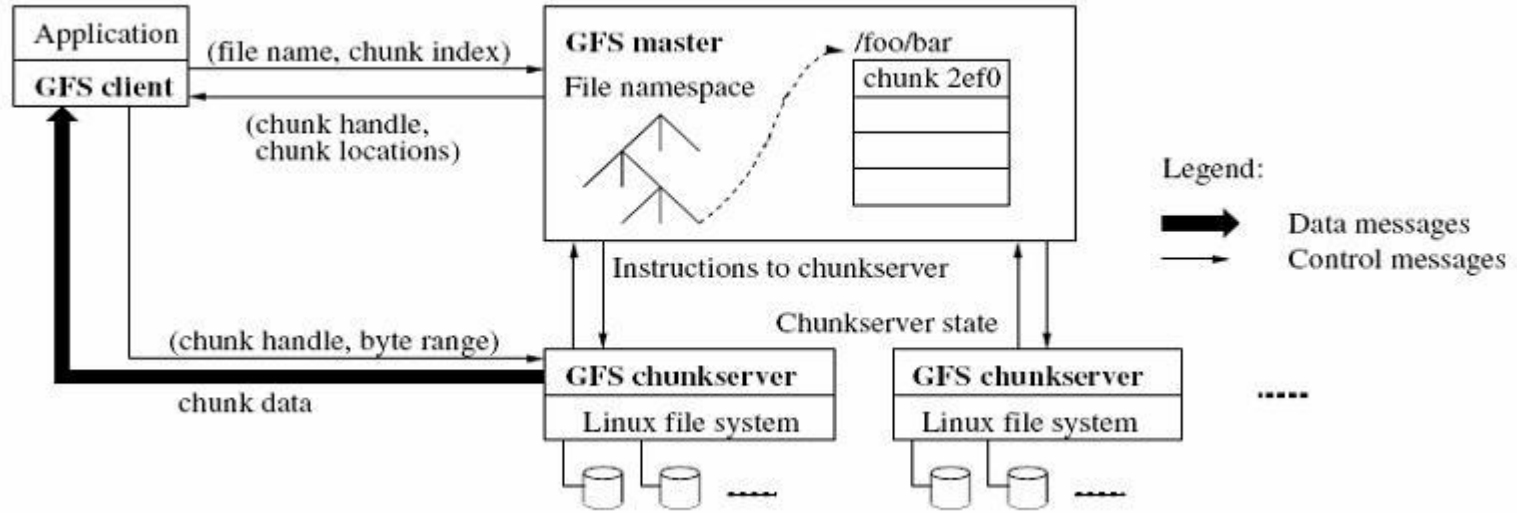
BitTorrent

- „Trackerlose" Systeme
  - Trackerfunktion wird von den Clients mit übernommen.
  - Dadurch wird fehlende Ausfallsicherheit des Trackers vermieden
  - Tracker kann dezentral als Distributed Hash Table auf den Clients (im Netz) selbst abgelegt und verwaltet werden.
  - Distributed Hash Table kann den Tracker vollständig ersetzen.

Google File System

- Von Google Labs entworfen, um die schnell wachsenden Anfragen von Google's Datenverarbeitungsprozessen zu verbessern
- Cluster-Filesystem mit folgenden Zielen
  - Hohe Performanz
  - Skalierbarkeit
  - Zuverlässigkeit
  - Verfügbarkeit
- Fehler-Toleranz und Auto-Recovery Integration
- Dateigröße ab 100 MB

Google File System



[GFS Architecture]

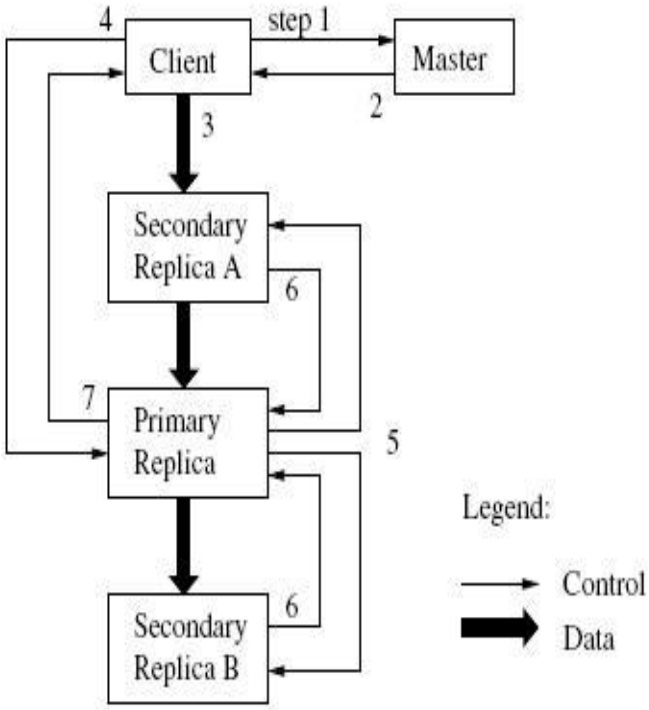
- Ein GFS Master
- Mehrere GFS Chunkserver
- Mehrere GFS Clients

Google File System

- GFS Master
  - Verwaltet Metadaten (Datei- und Chunk-Namensräume, Zugriffsrechte sowie das Mapping von Dateien nach Chunks und die Position jeder Chunkkopie)
  - Koordiniert systemweite Aktivitäten
- GFS Chunkserver
  - Führt Operationen auf den Chunks aus
  - Chunks werden auf anderen Chunkservern repliziert
- Chunks fester Größe (64MB)
  - Identifizierung durch unveränderlichen, globalen und einzigartigen 64 bit chunk handle. Handle wird vom GFS Master zugewiesen

Google File System

- Schreiboperation
  1. Client stellt eine Anfrage an den Master
  2. Master gibt den Ort der primären und sekundären Kopie zurück
  3. Der Client übermittelt die Daten an alle Kopien
  4. Der Client sendet einen Schreibbefehl an die primäre Kopie. Der Schreibbefehl identifiziert die vorher gesendeten Daten und wird mit einer eindeutigen seriellen Nummer durch die primäre Kopie versehen



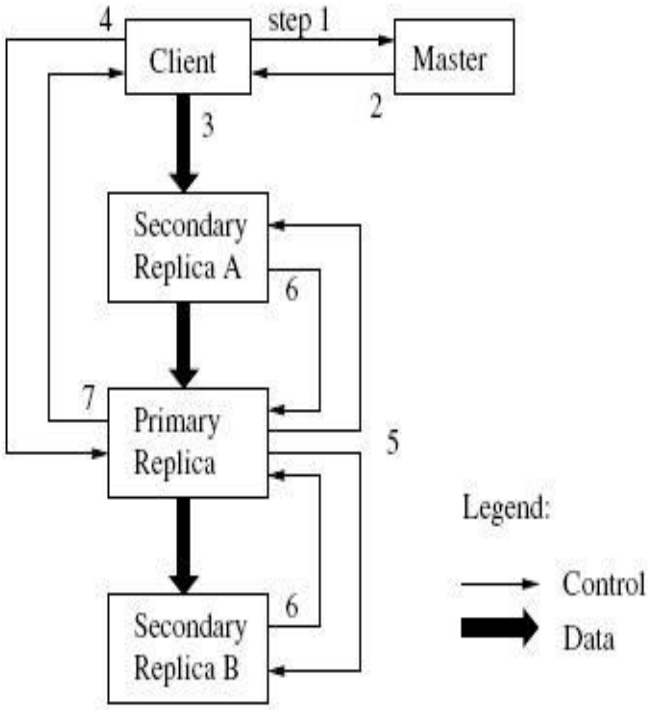
Google File System

- Schreiboperation

5. Die primäre Kopie sendet den Schreibbefehl an alle sekundären Kopien weiter, so dass alle Kopien dieselben Mutationen mit denselben eindeutigen Nummern halten

6. Die sekundären Kopien geben die Bestätigung der erfolgreichen Ausführung an die primäre Kopie zurück

7. Die primäre Kopie antwortet dem Client mit dem Erfolg des Schreibbefehls oder eventuell aufgetretener Fehler. Im Falle von Fehlern wird der Client die Schritte 3 bis 7 wiederholen, bis der Schreibbefehl erfolgreich ausgeführt wurde. Gegebenenfalls muss der ganze Schreibbefehl neu ausgeführt werden



Google File System

- Zuverlässigkeit und Verfügbarkeit auf Software-Ebene
  - Replikationen
  - Fast Recovery
- Hohe Performanz bei Operationen
  - Minimale Involvierung des Masters
  - Trennung von Daten- und Kontrollfluss
- Datenkonsistenz
  - Atomare Operation „Record Append"

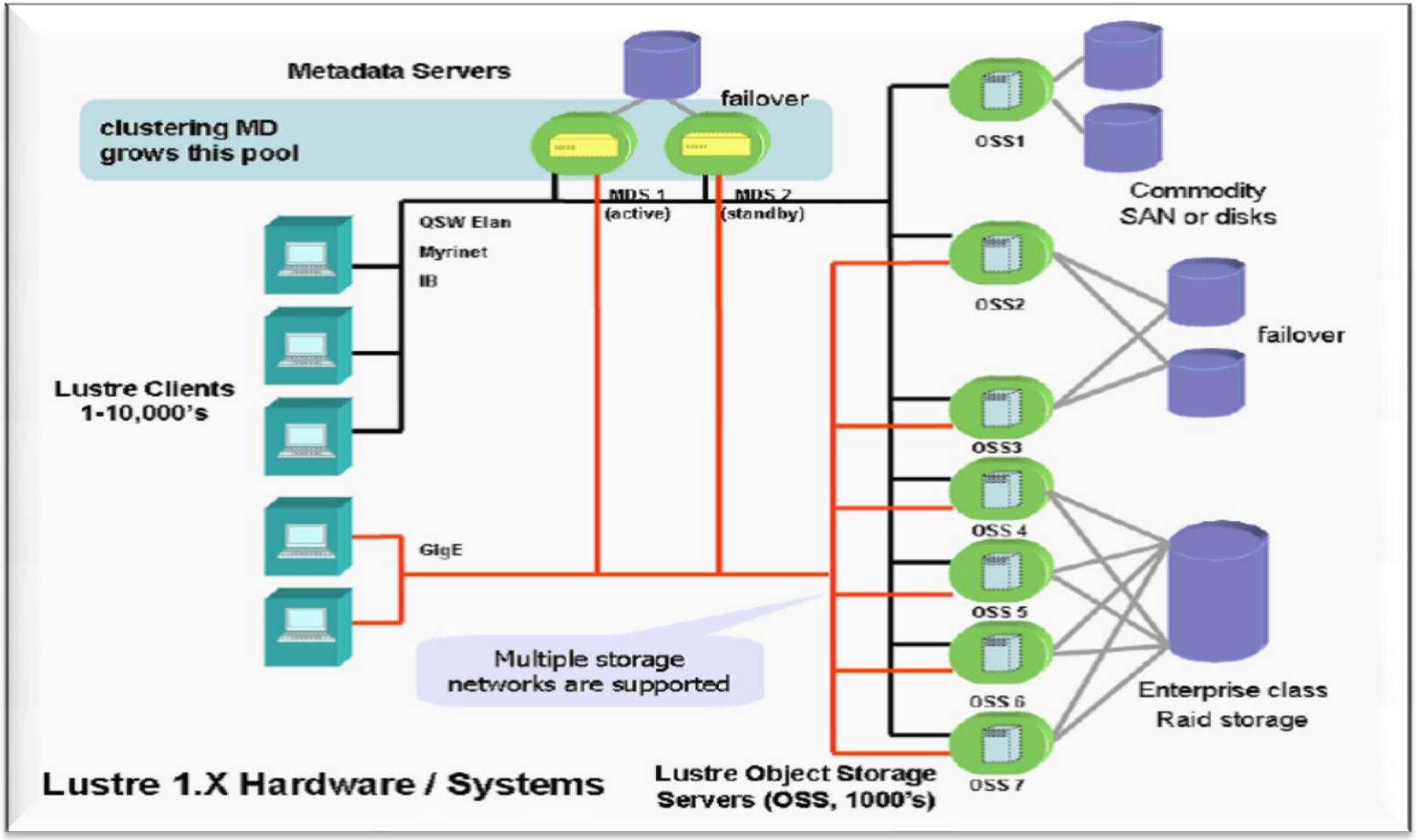
Lustre File System

- Skalierbares, sicheres, robustes, ausfallsicheres Cluster Datei System von Cluster File System Inc. http://www.clusterfs.com/

- Besteht aus drei Hauptsystemen
  - Meta Data Server (MDS)
  - Object Storage Server (OSS)
  - Lustre Clients
    - Interaktion mit OSSs für Daten I/O
    - Interaktion mit MDS für die Metadaten

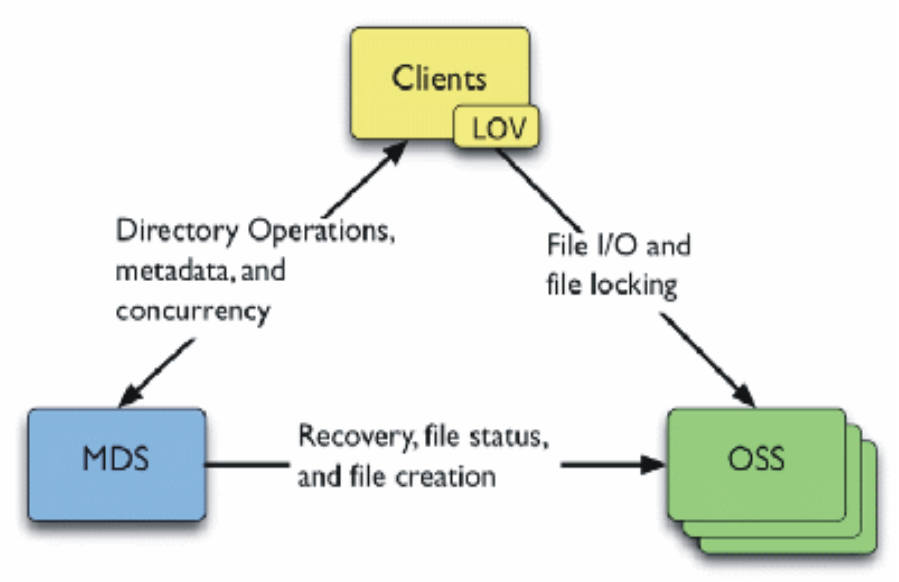
- Trennung von Metadaten und echten Daten

Lustre File System



A Lustre Cluster

Lustre File System



Interaktion zwischen Systemen

- Auch wenn Client, OSS und MDS Systeme getrennt sind, sieht Lustre wie ein Cluster Dateisystem mit einem Dateimanager aus

Lustre File System

- Meta Data Server
  - Bietet Back-End Speicher für Metadaten Service
  - Speichert Referenz zu echten Daten
  - Aktualisiert diesen Service bei jeder Transaktion über Netzwerkschnittstelle
  - Benutzt ein Journaling-Dateisystem
  - Das Lustre Dateisystem beinhaltet geclusterte Metadaten.
    Die Bearbeitung von Metadaten wird mit Hilfe von Lastverteilung durchgeführt, im Ergebnis ist der gleichzeitige Zugriff auf Metadaten sehr komplex

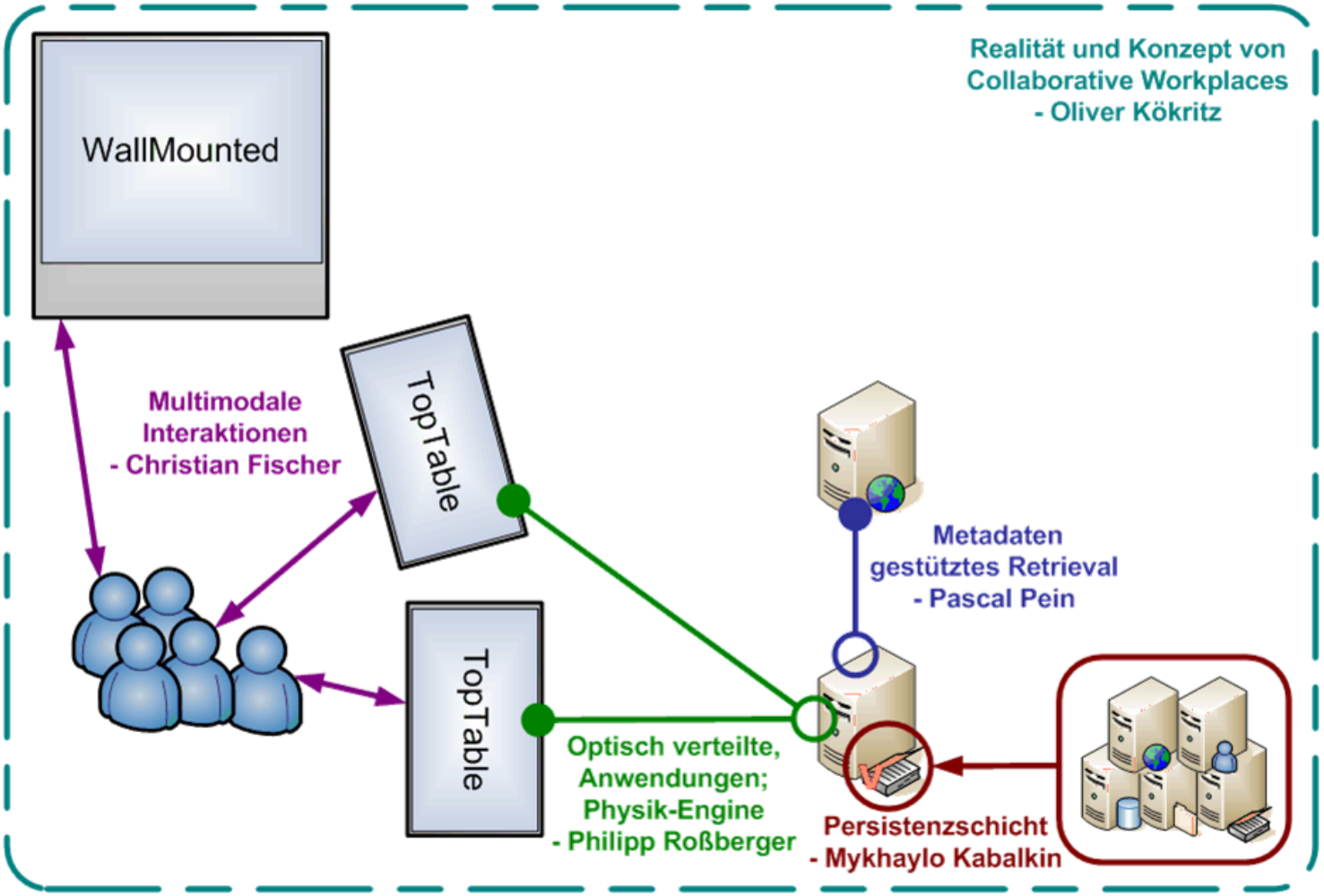
Lustre File System

- Object Storage Server
  - Kann mehrere Netzwerkschnittstellen und normalerweise eine oder mehrere Festplatten haben
  - Bietet File Input/Output Service
  - Speichert echte Daten

Übersicht

- Einleitung
  - Was ist ein Dateisystem?
  - Was ist ein verteiltes Dateisystem?
- Einige bekannte Systeme
  - Network File System
  - Andrew File System
  - BitTorrent
  - Google FS
  - Lustre FS
- **Einordnung ins Project**
- Quellen

Einordnung ins Projekt



Übersicht

- Einleitung
  - Was ist ein Dateisystem?
  - Was ist ein verteiltes Dateisystem?
- Einige bekannte Systeme
  - Network File System
  - Andrew File System
  - BitTorrent
  - Google FS
  - Lustre FS
- Einordnung ins Project
- Quellen

Quellen

Distributed File System, State University of New York at Buffalo, Dr. Bina Ramamburthy
http://www.cse.buffalo.edu/gridforce/fall2004/DistributedFileSystemSept29.pdf

Network File System (NFS) version 4 Protocol, RFC 3530 http://tools.ietf.org/html/rfc3530

An AFS-based mass storage system at the Pittsburgh Supercomputing Center, D. Nydick, K. Benninger, B. Bosley, J. Ellis, J. Goldick, C. Kirby, M. Levine, C. Maher, M. Mathis
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=160222

NFSv4 Protokoll http://www.nfsv4.org/

OpenAFS http://www.openafs.org/

The dice project, University of Edinburgh „A Comparison Between AFS and NFSv4"
http://www.dice.inf.ed.ac.uk/groups/services/file_service/docs/newfs-choice.html

Incentives Build Robustness in BitTorrent, Bram Cohen
http://www.bittorrent.com/bittorrentecon.pdf

Quellen

Robust and Efficient Data Management for a Distributed Hash Table, Josh Cates
http://pdos.csail.mit.edu/papers/chord:cates-meng.pdf

The Google File System, Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung
http://labs.google.com/papers/gfs-sosp2003.pdf

Bigtable: A Distributed Storage System for Structured Data, Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber
http://labs.google.com/papers/bigtable-osdi06.pdf

Cluster File System Inc., Lustre Manual, Version 1.4.7.1-man-v36
https://mail.clusterfs.com/wikis/lustre/LustreDocumentation?action=AttachFile&do=get&target=LustreManual36.pdf

Cluster File System Inc., Lustre: A Scalable, High-Performance File System
http://www.lustre.org/docs/whitepaper.pdf

Vielen Dank!