

# Apple HFS+

---

Seminar: Dateisysteme

Christian Lohse, 2007

# Inhalt

---

- Über HFS+
- Hauptziele der Entwickler
- Terminologie und B\*-Bäume
- Strukturen
- Weitere Konzepte
- Zukunft von HFS+
- Zusammenfassung

# Über HFS+

---

- Von Apple (Computer) Inc.
- Nachfolger für HFS
- Am 19.1.1998 mit OS 8.1 eingeführt  
(aktuell OS X 10.4)
- Eingesetzt auf allen Macintosh Computern und auch iPods

auch bekannt unter: HFS Plus, HFS Extended, Mac OS Extended  
Entwicklungsname: „Sequoia“ (deutsch: „Mammutbaum“)  
in den 90ern entwickelte Apple das Betriebssystem Copland, HFS+ als Beta-Dateisystem integriert  
Copland wurde nie veröffentlicht  
iPod im Mac-Format nutzt HFS+ (wenn unter Windows genutzt: FAT32)

# Hauptziele von HFS+

---

- effizientere Ausnutzung des Speicherplatzes
- länderunabhängige Dateinamen
- zukünftige Unterstützung für benannte Forks
- einfaches Booten von Nicht-Mac OS Betriebssystemen

# HFS+: effizientere Ausnutzung des Speicherplatzes

---

- Sprung von  $2^{16}$  auf  $2^{32}$  Bit Adressraum
- 2 Terabyte in 512 Bytes-Blöcken adressierbar
- Auslegung auf Speicherung von bis zu 16 Exabyte
  - Effizient?  
4 Gigabyte große Allocation Blocks!

Faktor  $2^{16}$  (65.536) an Adressen  
HFS: 32 Megabyte in 512 Bytes-Blöcken adressierbar

4 GB: sinnvoll für Multimedia (Videodaten)

Momentan nutzt Mac OS X 10.4  $2^{31}$  Adressen, bei max. 8KB Blockgröße -> 16 TByte Speicherplatz

# HFS+: länderunabhängige Dateinamen

---

- 255 Unicode Zeichen für Dateinamen
  - Unicode!
  - besser erklärende Dateinamen
  - Praktisch auch für generierte Dateinamen (zB:Java Class Names)

# HFS+: zukünftige Unterstützung für benannte Forks

---

- Dateien haben eine Data Fork und eine Resource Fork
- Dateien und Ordner besitzen bereits einen kleinen Teil an Metadaten im Catalog File
- Apple und andere Entwickler möchten jedoch die Möglichkeit für zusätzliche Metadaten haben
- Attribute File
  - beliebige Anzahl an benannten Forks für jede beliebige Datei/jeden beliebigen Ordner

zusätzliche Metadaten: Früher als extra Datei angelegt, aber bei FS-Operationen nicht beachtet

-> Inkonsistenzen als Folge

Mit 'Attribute' File vom System verwaltet.

# HFS+: einfaches Starten von Nicht-Mac OS Betriebssystemen

---

- Möglichkeit für Nicht-Mac OS Betriebssysteme von HFS+ Dateisystemem zu starten
- Besonders hilfreich für Systeme ohne HFS/ HFS+ Unterstützung im ROM
- Verallgemeinerung des HFS Boot Blocks

# HFS vs. HFS+

---

<i>Feature</i>	<i>HFS</i>	<i>HFS+</i>
Bezeichner	Mac OS Standard	Mac OS Extended
Anzahl der Zuordnungsblöcke	16 Bit Werte	32 Bit Werte
Länge der Dateinamen	31 Zeichen	255 Zeichen
Dateinamenkodierung	MacRoman	Unicode
Datei-/Ordnerattribute	FileInfo / ExtendedFileInfo	Ausbau der Metadaten
Catalog Knotengröße	512 Bytes	4096 Bytes
max. Dateigröße	$2^{31}$ Bytes	$2^{63}$ Bytes

# HFS+ Terminologie

---

- Disk: Medium auf dem die Nutzdaten gespeichert werden
- Volume: Dateien, gemeinsam mit einer Struktur deren Daten zu erhalten
- Sector: kleinste Einheit welche der Festplattentreiber bei einer Operation liest oder schreibt
- Allocation Block: zusammenhängender Block von Sektoren
- Clump: zusammenhängende Gruppe von Allocation Blocks
- Extent: zusammenhängende Gruppe von Allocation Blocks welche einer Fork zugeordnet ist

10

10

Sektor: HDD – 512 Bytes, CD-ROM – 2048 Bytes

Eigentlich ist nur das Journal, da es auf Sektor-Basis arbeitet, von der Sektor-Größe abhängig.

aktuelle Implementierungen auf 4KB Allocation Blocks optimiert (16 Terabyte bei  $2^{32}$  Adressen)

Größe ist ein Speed-vs.-Space Kompromiss:

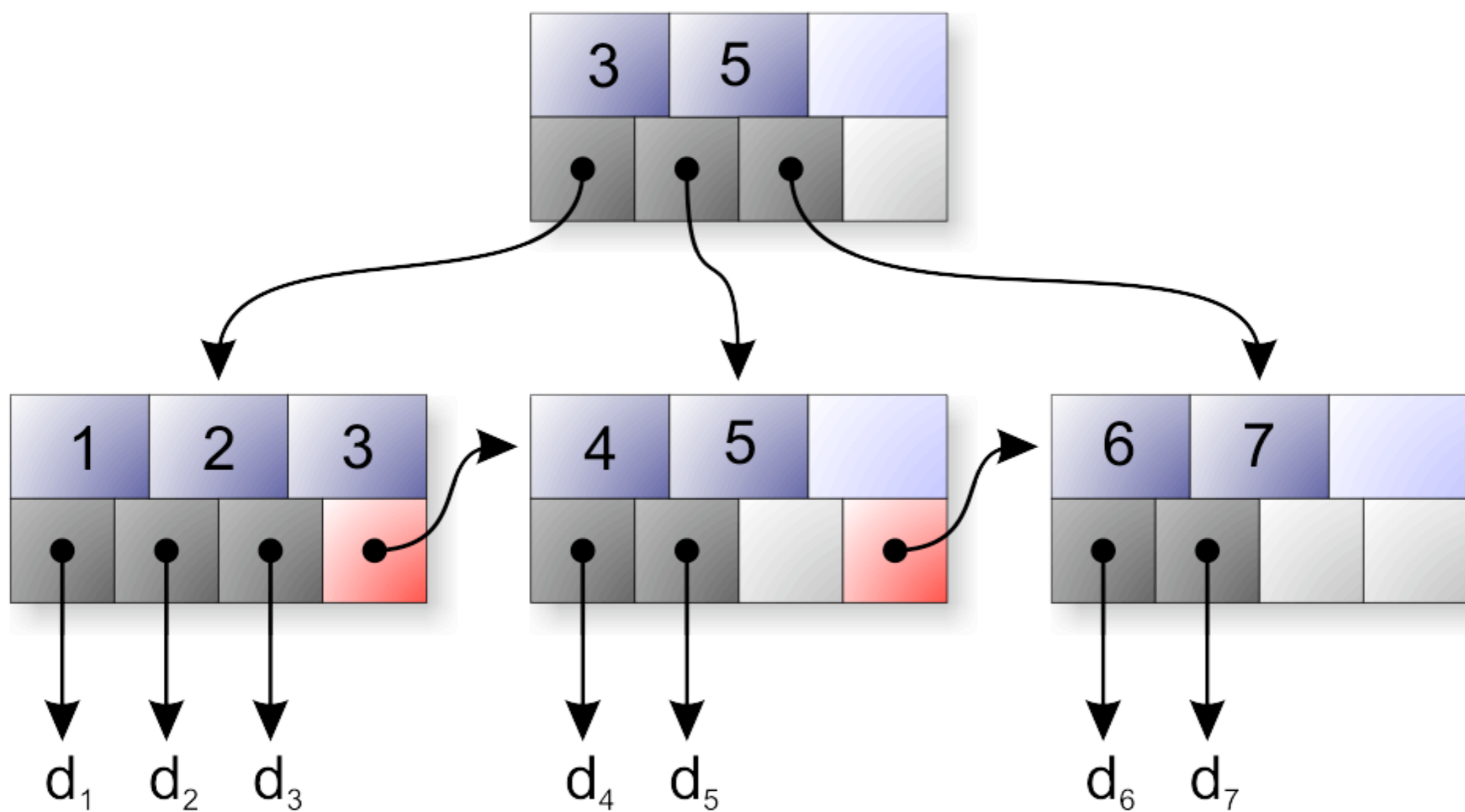
Große Alloc. Blöcke -> weniger einzelne Extents müssen manipuliert werden  
-> größere I/O-Zugriffe, weniger Overhead

Kleine Alloc. Blöcke -> Durchschnittliche Anzahl verschwendeter Bytes pro Datei sinkt, effizientere Nutzung

Clumps: vielfaches der Allocation Block Größe

Extent: Zahlentupel – (erste Allocation Block Nummer, Anzahl der Allocation Blocks im Extent)

# B\*-Bäume



B\*-Tree um schnelle und effiziente Suchen durch große Hierarchien zu ermöglichen

B\*-Tree Knoten zu 2/3 gefüllt, nicht nur zu 1/2

René Kraneis hat in ReiserFS4 Vortrag hierzu berichtet

# HFS+ Strukturen

---

- Volume Header
- Catalog File
- Extents Overflow File
- Attributes File
- Allocation File
- Startup File

# HFS+: Volume Header

---

- Beginnt bei 1024 Bytes eines Volumes
- Informationen über das Volume (z.B. Datum und Uhrzeit der Erzeugung eines Volume, Anzahl der Dateien auf dem Volume, ...)
- Informationen über Lage der anderen Strukturen
- Muss vor einem Unmount aktualisiert werden
- Alternate Volume Header: eine Kopie des Volume Header, beginnt bei 1024 Bytes vor dem Ende eines Volumes

# HFS+: Catalog File

---

- B\*-Baum
- Beschreibt Ordner- und Dateistruktur
- Informationen für den Zugriff auf die Daten
- Informationen über die Daten selbst (Metadaten)
- Die jeweils ersten acht Extents werden hier abgelegt

# HFS+: Catalog File Record

---

```
struct HFSPlusCatalogFile {
    SInt16          recordType;
    UInt16          flags;
    UInt32          reserved1;
    HFS CatalogNodeID fileID;
    UInt32          createDate;
    UInt32          contentModDate;
    UInt32          attributeModDate;
    UInt32          accessDate;
    UInt32          backupDate;
    HFSPlusBSDInfo permissions;
    FileInfo        userInfo;
    ExtendedFileInfo finderInfo;
    UInt32          textEncoding;
    UInt32          reserved2;

    HFSPlusForkData dataFork;
    HFSPlusForkData resourceFork;
};
typedef struct HFSPlusCatalogFile HFSPlusCatalogFile;
```

15

15

recordType: für Dateien immer -> kHFSPlusFileRecord  
flags: Bitflags für die Datei (kHFSFileLockedBit/Mask, kHFSThreadExistsBit/Mask)  
reserved1: reserviert ;)  
fileID: catalog node ID (CNID)  
createDate: wann wurde die Datei erstellt  
...  
Permissions: Ähnlich zu den POSIX-Permissions  
userInfo: für den Mac OS Finder (Dateimanager)  
finderInfo: dito  
textEncoding: Hinweis auf die Kodierung aus der der Name stammt  
reserved2: reserviert ;)  
dataFork: Info über Größe und Adresse der DataFork  
resourceFork: dito

# HFS+: Extents Overflow File

---

- B\*-Baum
- Wenn es mehr als acht Extents gibt, werden die zusätzlichen hier abgelegt
- Auch für die Special Files
- Bad Block File: verhindert den Zugriff auf beschädigte Teile des Datenträgers
  - In das Extents Overflow File integriert

meistens weniger als acht Extents nötig

wenn ein Sektor Bad ist, so wird der ganze Block als Bad markiert

einen Bad Block nur in der Allocation File zu markieren führt bei einem Consistency Check zu Problemen, da nur Allocated Blocks mit zugehörigen Extens allokiert bleiben, ohne Bad Block Record also freigegeben würden

# HFS+: Extents Overflow File Key

---

```
struct HFSPlusExtentKey {
    UInt16          keyLength;
    UInt8           forkType;
    UInt8           pad;
    HFSCatalogNodeID fileID;
    UInt32          startBlock;
};
typedef struct HFSPlusExtentKey HFSPlusExtentKey;
```

keyLength: Key

forkType: 0 für Data, 0xFF (256<sub>10</sub>) für Resource

pad: pad-Field

fileID: CNID der Datei zu welcher dieses Extent Eintrag gehört

startBlock: offset (in Alloc. Blöcken) in die Fork hinein

# HFS+: Attributes File

---

- B\*-Baum
- Zukünftig genutzt um Informationen über zusätzliche Forks zu speichern
- Nicht zwingend für ein Volume
- „Änderungen vorbehalten“

eines der Ziele: zukünftige Unterstützung für benannte Forks

# HFS+: Allocation File

---

- Enthält Informationen darüber, ob ein Allocation Block frei oder belegt ist
- Die ersten 1024+512 Bytes (Boot Block + Volume Header) und die letzten 512+512 Bytes (Alternate Volume Header + Reserved) werden hier standardmäßig als belegt markiert
- Muss nicht unbedingt zusammenhängend sein
- Variabel in der Größe (erweitern, schrumpfen)

Reserved: wird von Apple während der Computerherstellung benötigt

Bitmap: 1 Bit für jeden Allocation Block auf der HDD, 1 belegt, 0 frei

verschachtelt mit Userdaten -> höhere Zugriffszeiten

erweitern: Allocation Block Size verkleinern, Gesamtspeicher erhöhen

schrumpfen: Alloc. File in einem Disk Image ausgelegt auf größtmögliche Disk, wird geschrumpft wenn es auf eine kleinere Disk geschrieben wird

# HFS+: Startup File

---

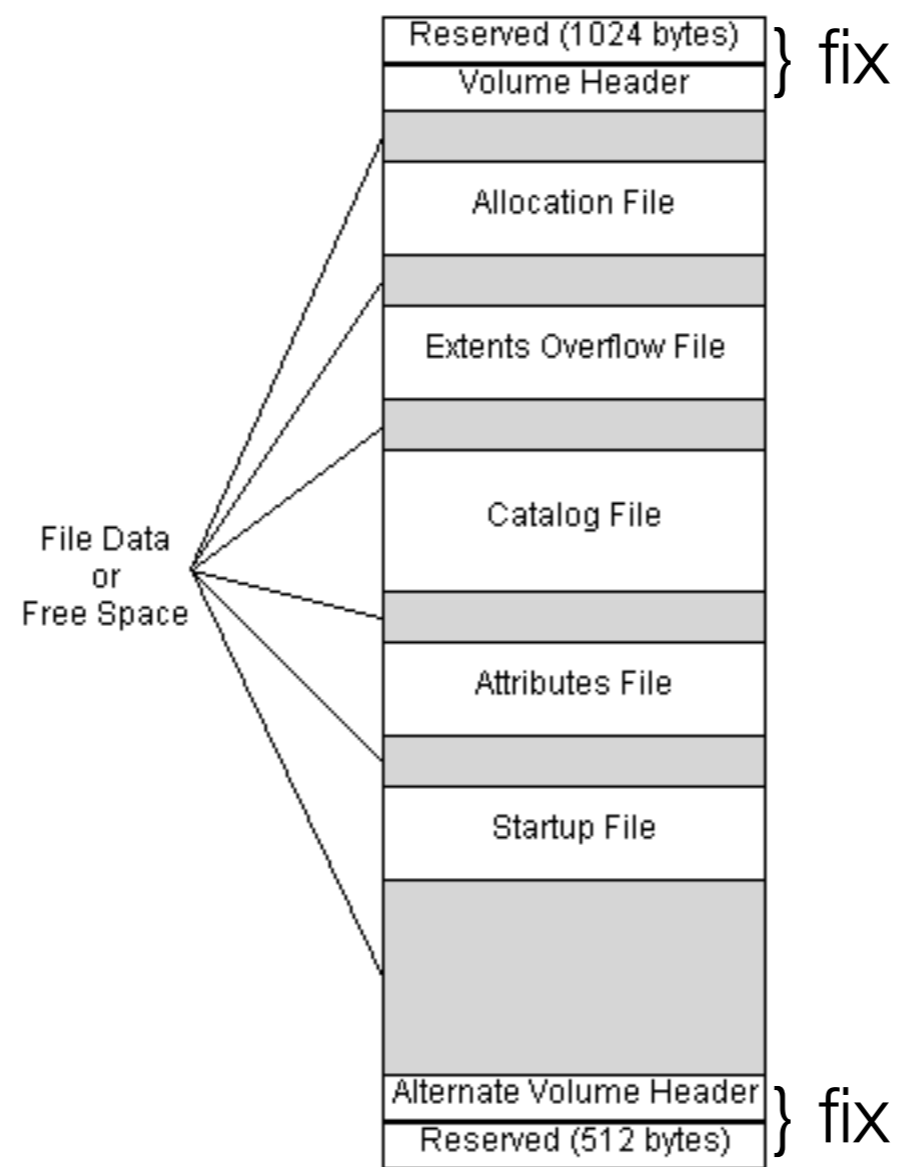
- Zur Unterstützung für nicht-Mac OS Computer um von HFS+ Volumes zu booten
- Ein Bootloader kann das Startup File ohne detaillierte Kenntnisse des HFS+ Format (B\*-Bäume, Catalog File, ...) finden
- Volume Header enthält hierfür die ersten acht Extents des Startup File
- Mac OS selbst benutzt das Startup File nicht

Startup File kann mehr als acht Extents haben (dann im Extents Overflow File), aber dann kein Bootvorgang mehr möglich :(

Mac OS benutzt HFS Wrapper!

# HFS+: Organisation

---



ersten 1024 Bytes: Boot Block

letzten Reserved 512 Bytes: für Computerherstellung von Apple genutzt

# HFS+: Hard Links / Symbolic Links

---

- Hard Links sind normale Dateien im Catalog File, welche auf Indirect Node Files verweisen
  - Indirect Node Files befinden sich im Metadata Directory, beinhalten den „Inode“
  - Wozu? Damit auch Backup-Programme welche nicht mit Hard Links umgehen können ein richtiges Backup erzeugen können
- Symbolic Links: ganz gewöhnlich gehandhabt

Indirect Node Files entsprechen UNIX Inodes

Symbolic Link beinhaltet ganz normal Verweis auf Datei/Ordner

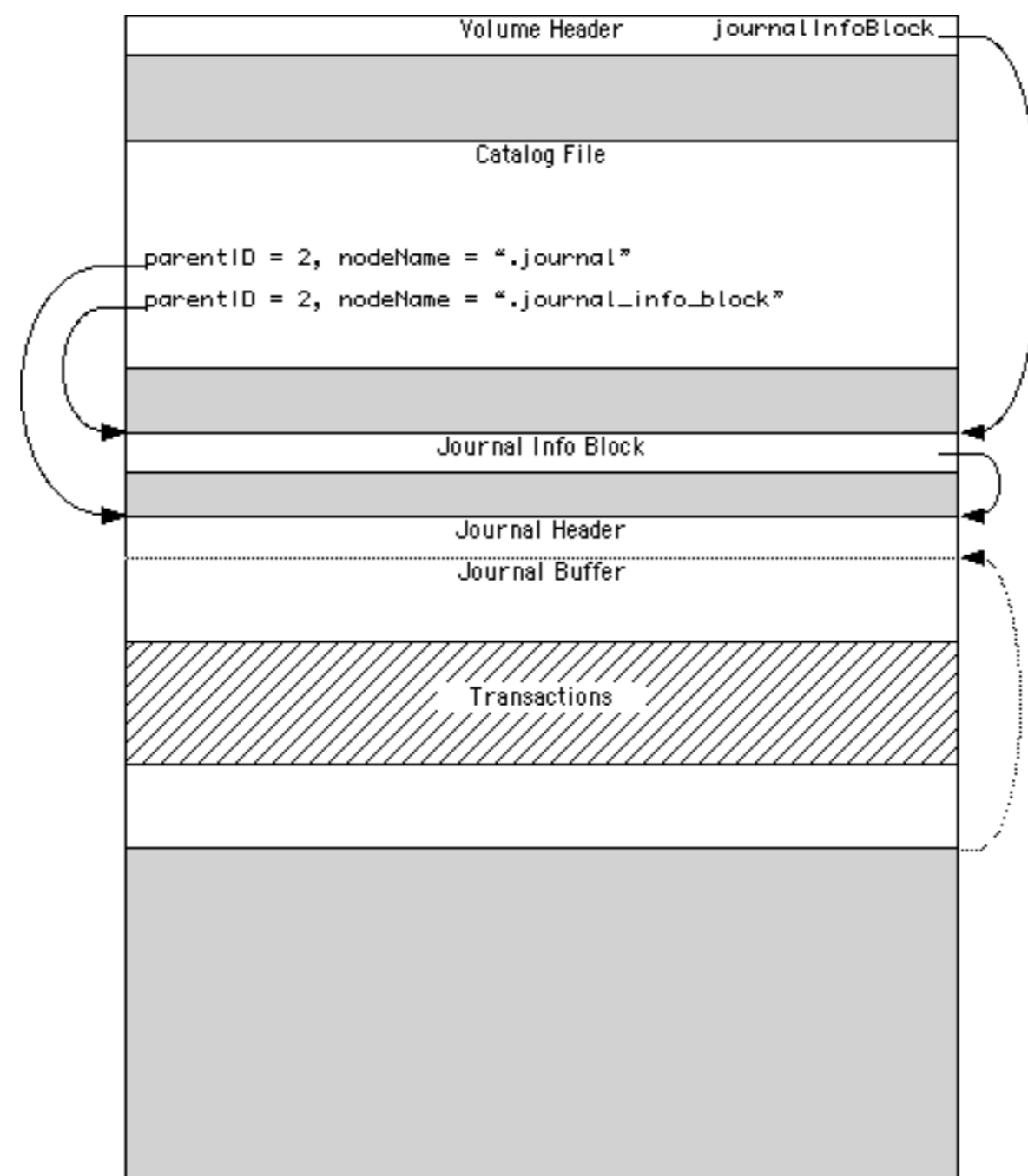
# HFS+: Journal

---

- Für die schnellere Wiederherstellung nach einem unsicheren Unmount
- Nur für Metadaten und Volumenstruktur
- Hauptbestandteile: Journal Info Block, Journal Header und Journal Buffer

Wie alle Journals, nicht für die Daten Fork  
ansonsten Journaling wie im Vortrag von Andreas Beyer beschrieben  
Journal Info Block: Wo sind Journal Header und Buffer, Link zu JIB im Volume Header  
Journal Header: Hauptaufgabe ist Ort der Transaktionen im Buffer zu beschreiben  
Buffer enthält Blöcke von Transaktionen, zirkulär! (Start != Ende)

# HFS+: Journal



Wie alle Journals, nicht für die Daten Fork  
ansonsten Journaling wie im Vortrag von Andreas Beyer beschrieben  
Journal Info Block: Wo sind Journal Header und Buffer, Link zu JIB im Volume Header  
Journal Header: Hauptaufgabe ist Ort der Transaktionen im Buffer zu beschreiben  
Buffer enthält Blöcke von Transaktionen, zirkulär! (Start != Ende)

# HFS+: Metadata Zone

---

- Richtlinie, wo Platz für Daten reserviert werden soll, um die Leistung für den Nutzer zu erhöhen
- Metadaten und häufig genutzte kleine Dateien (“Hot Files“) werden beieinander abgelegt
- Reduzierte Suchzeiten

mit Mac OS X 10.3 eingeführt  
weniger Kammbewegungen der Festplatte, schnellerer Zugriff  
Volume Metadata: Dateien um Inhalt zu managen (Alloc. Bitmap File, Extentes Overflow File, Catalog File und Journal File)  
(Alternate) Volume Header sind ebenfalls Metadaten, aber mit fixen Positionen  
User Quota und Groups Quota auch in Metadata Zone, da häufig genutzt, aber keine Metadata und keine Hot Files (zu groß)  
normale Files werden außerhalb der M.Zone platziert, wenn der Platz dort voll ist, dann in Metadata Zone – wenn Metadata Zone mit Metadaten voll, dann geht’s außerhalb weiter

# HFS+: Hot Files

---

- B\*-Tree
- Die meisten Dateien auf einer Festplatte werden selten benutzt
- Auf einige kleine Dateien wird jedoch häufig zugegriffen
- Diese „heißen Dateien“ werden defragmentiert in die Metadata Zone geschrieben
- Während der Aufzeichnung wird die Anzahl der gelesenen Bytes einer Datei durch deren Größe geteilt, hieraus ergibt sich die „Temperatur“
- Schritte zum Hot File: Aufzeichnung, Auswertung, Ausweisung, Annahme

26

26

defragmentiert: von wenigen Extents (da klein) in einen geschrieben -> adaptive hot file clustering

Aufzeichnung: welche Dateien werden häufig zugegriffen, Auswertung: welche sind schon hot, Ausweisung: die in der M.Zone befindlichen cold files fliegen raus, Annahme: die neuen hot files kommen in die M.Zone (Recording, Evaluation, Eviction, Adoption)

# HFS+: Unicode Feinheiten

---

- Prinzipiell: aufgespaltene Zeichen
  - Aus é (u+00E9) wird e (u+0065) mit einem zugehörigen Akzent-Symbol ´ (u+0301)
- Ausnahmen bestätigen die Regel
  - Zeichen zwischen u+2000 und u+2FFF werden nicht aufgespalten
  - Beispiel „(a)“ wird nicht zu (, a und )

# HFS+: Volume Consistency Checks

---

- HFS+ verfügt über eine komplexe Struktur, wenn es hier Fehler gibt, kann dies zu Datenverlust führen
- Daher wird ein Volume beim Mounten geprüft, ob es Inkonsistenzen aufweist
- Prüfung ob die „nextCatalogID“ korrekt ist
- Prüfung ob der Allocation File korrekt ist

P.1: alle Einträge werden geprüft und die höchste ID inkrementiert und gesetzt

P.2: Alloc. File wird 0 gesetzt und jeder Extent eingetragen

# HFS+: Defragmentierung

---

- Folgende fünf Bedingungen müssen erfüllt sein:
  - Dateigröße ist kleiner als 20 MByte
  - Datei ist nicht bereits in Gebrauch
  - Datei ist nicht schreibgeschützt
  - Datei ist fragmentiert  
(mehr als acht Extents benötigt)
  - System läuft seit mindestens drei Minuten
- Dann wird on-the-fly defragmentiert durch umplatzieren der Datei

# HFS+: Verschlüsselung?

---

- Nicht für das ganze Volume
- Stichwort: FileVault
  - Der Home-Ordner wird durch ein HFS+ Image ersetzt
  - Dieses Image wird mit AES-128 verschlüsselt

# HFS Wrapper

---

- HFS+ Volume befindet sich innerhalb eines HFS Volume
- Vorteil 1: Computer ohne HFS+ Unterstützung können von HFS+ starten
- Vorteil 2: Benutzer werden nicht verwirrt, wenn Volume unter nur-HFS System gemountet wird, da es keine Fehlermeldungen gibt und eine nette Readme über das aktuelle Problem aufklärt

# HFS Wrapper

---

```
$ sudo mount -t hfs /dev/sda /mnt
```

```
$ ls /mnt
```

```
Desktop DB Desktop DF Finder System Where_have_all_my_files_gone?
```

```
$ cat /mnt/Where_have_all_my_files_gone\?
```

Why can't you see your files?

This hard disk is formatted with the Mac OS Extended format. Your files and information are still on the hard disk, but you cannot access them with the version of system software you are using.

How can you access your files?

To access your files you must mount this hard disk on a computer that has Mac OS 8.1 or later installed. To determine the version of system software you're currently using, choose About This Computer from the Apple menu. If you're using a version of the Mac OS earlier than 8.1, you must do one of the following:

- A) upgrade the system software on your computer,
- B) start up the computer from a hard disk or CD that has Mac OS 8.1 or later, or
- C) connect the hard disk to another computer with Mac OS 8.1 or later installed.

If you want to access the files without upgrading your system software, start up your computer with the Mac OS 8.1 CD or the Disk Tools PPC disk, then access your files. Apple recommends that you have the Mac OS 8.1 CD if you plan to reformat any hard disk using Mac OS Extended format.

To continue to use this hard disk with this computer, you must upgrade your system software to Mac OS 8.1.

How do you upgrade your system software?

If you have a version of system software earlier than Mac OS 8, you can order Mac OS 8.1 on the Internet or buy it at a local Apple software reseller.

If you have Mac OS 8 on your computer, you can download the Mac OS 8.1 update from the Internet at <http://www.info.apple.com>.

Copyright 1998 Apple Computer, Inc. All rights reserved.

Apple and Mac OS are trademarks of Apple Computer, Inc., registered in the U.S. and other countries. PowerPC is a registered trademark of International Business Machines Corporation, used under license therefrom.

# HFSX

---

- Erweiterung von HFS+, aber inkompatibel zu HFS+
- Erlaubt Groß-/Kleinschreibung
- Neue Funktionen integrierbar, aber nicht näher spezifiziert
- HFSX benutzt keinen HFS Wrapper

quasi die nächste Version nach HFSX (Version 5)  
case-sensitive (muss aber nicht)  
neue Features über Volume Attribute Bits erkennbar

# Zukunft: ZFS?

---

- Momentan nichts über Weiterentwicklungen an HFS+/HFSX bekannt
- Viele Spekulationen darüber, ob ZFS das Dateisystem von Mac OS X 10.5 „Leopard“ wird (erscheint im Oktober 2007)
- In der aktuellen Beta-Version kein Hinweis auf ZFS enthalten
- Stichwort: TimeMachine
  - ZFS würde sich mit seinen Snapshots

# Zusammenfassung

---

- Umfassende Struktur, viele B\*-Bäume
- Aktuelle Standards vertreten
- Platz für Innovationen
- Noch genug Adressreserven für größere Festplatten
- Apple: “Volume format specifications are ~~fun~~ exhausting.”

aktuelle Standards: Unicode, lange Dateinamen, Journal, ...

Zitat: Apple – steht so in der TechNote geschrieben!

# Quellen

---

- Literatur

- <http://developer.apple.com/technotes/tn/tn1150.html>
- <http://docs.info.apple.com/article.html?artnum=24319>
- <http://de.wikipedia.org/wiki/HFS%2B>
- [http://en.wikipedia.org/wiki/HFS\\_Plus](http://en.wikipedia.org/wiki/HFS_Plus)

- Bilder

- Folie 11: <http://upload.wikimedia.org/wikipedia/commons/a/a8/Btree.png>
- Folie 23: <http://developer.apple.com/technotes/tn/tn1150.html>