Jörg Schilling

Filesysteme

in der Vergangenheit und Heute

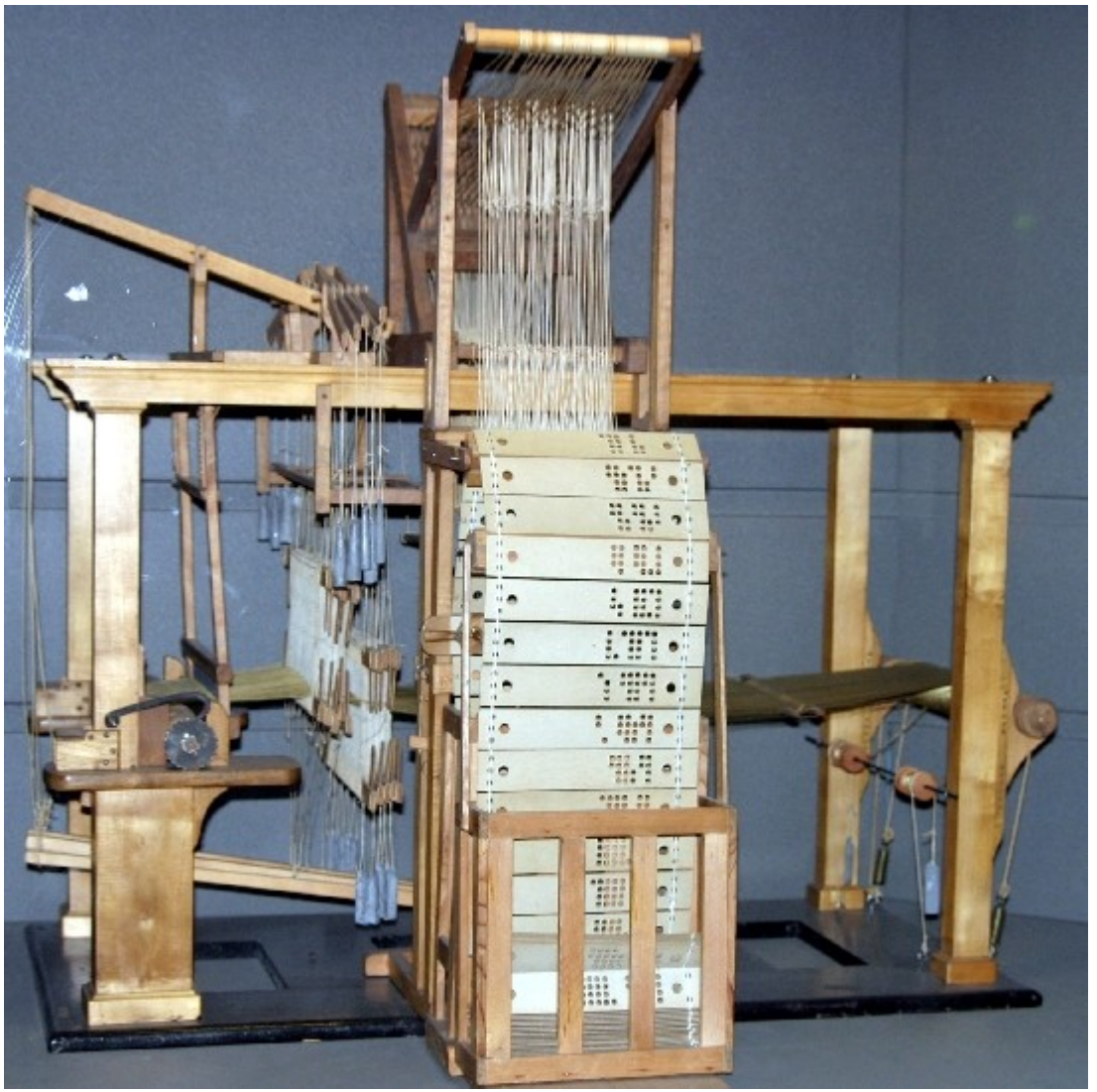
Fokus Fraunhofer

Fraunhofer
FOKUS

Datensammlungen in der Frühzeit

- Um 1750 erste Lochkartenähnliche Systeme
- 1796 Antoine Favre Walzenspieldose
- 1805 Jacquard Webstuhl mit Lochkartensteuerung
- 1837 Charles Babbage Lochkarten für Analytical Engine
- 1884 Herman Hollerith Patentierung der Lochkarte
- 1938 Zuse Z1 mechanisch, Programmierung mit Lochstreifen (35mm Film)
- 1939 DeHoMAG deutsche Volkszählung mit Lochkarten
- 1941 Zuse Z3 elektrisch, Programmierung mit Lochstreifen (35mm Film)
- 1956 magnetische Festplatte IBM 350, 1958 Trommelspeicher in Zuse Z22
- ca. 1965 Multics erstes hierarchisches Filesystem
  - 1967 erste lauffähige Multics Version

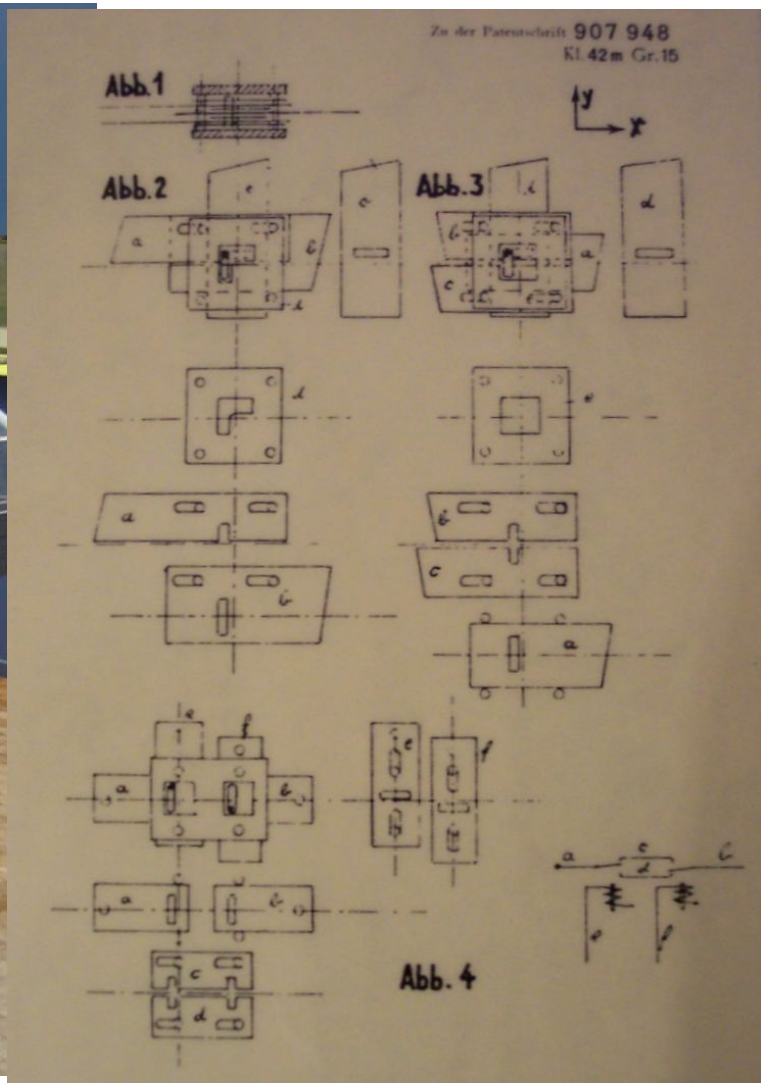
Jacquard Webstuhl (Modell)



DeHoMAG Tabelliermaschine D11 1936

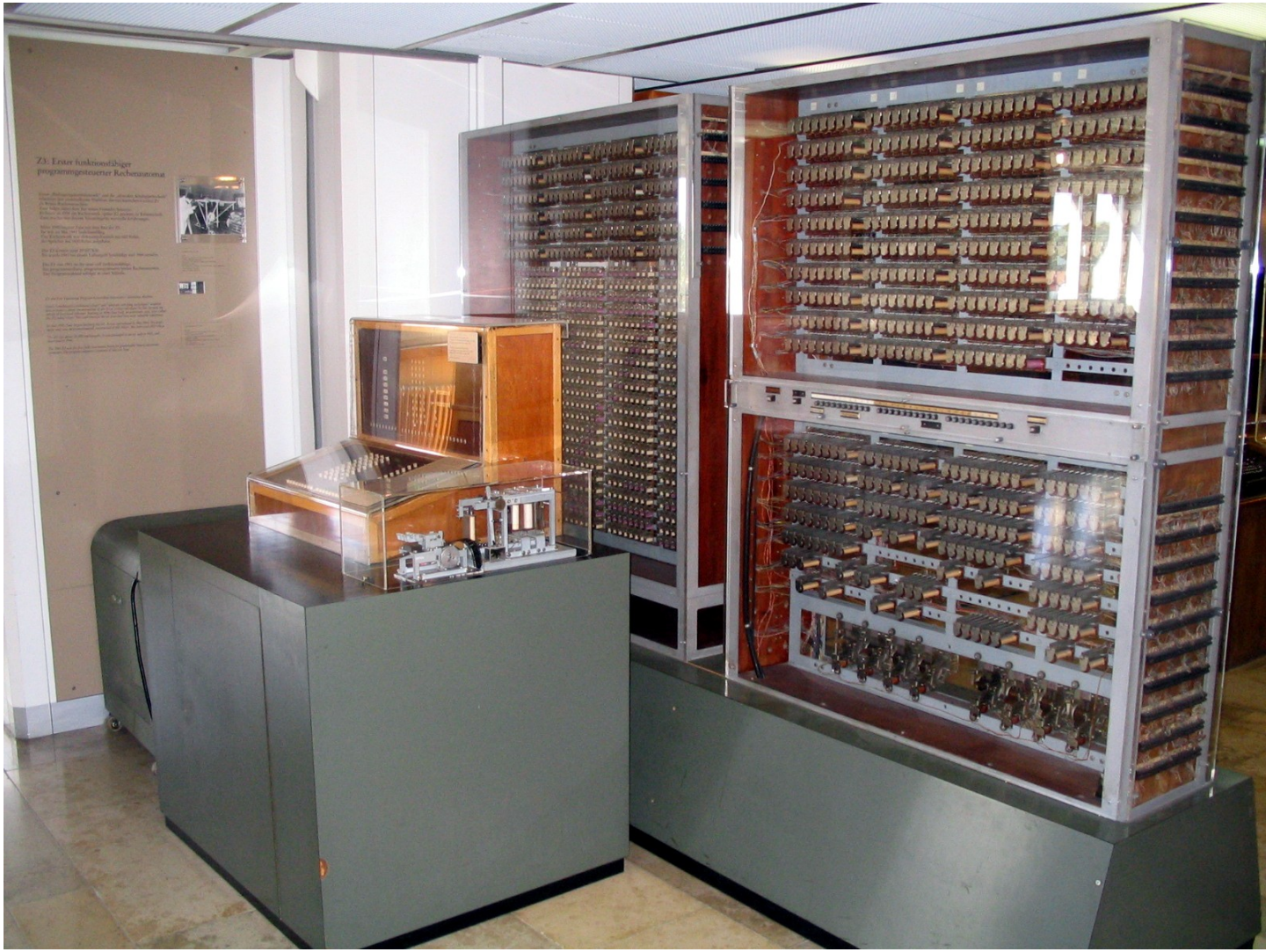
Fraunhofer FOKUS

Zuse Z1 Nachbau Technikmuseum Berlin



Fraunhofer FOKUS

Zuse Z3 Nachbau Deutsches Museum



Fraunhofer
FOKUS

Zuse Z4 (Original)

Um 1979 Lochkarten, IBM/370, Cyber 6000

- IBM/370: 32 Bit, 1MB Hauptspeicher, Betriebssystem CP/CMS, ca. 10 GB Festplatten (á 300 MB)
  - CMS: Einplatz Uniprozess Betriebssystem
  - CP: Virtualisierungs OS, lädt viele CMS Instanzen
  - Maschinencode: virtualisiert zur Abstraktion v. CPU
  - MMU zum Speicherschutz
- CDC Cyber 6000 Serie: 60 Bit, Betriebssystem NOS/BE
  - Keine MMU
  - Nur Batchverarbeitung
- Kartenlocher IBM 29 als Eingabemedium

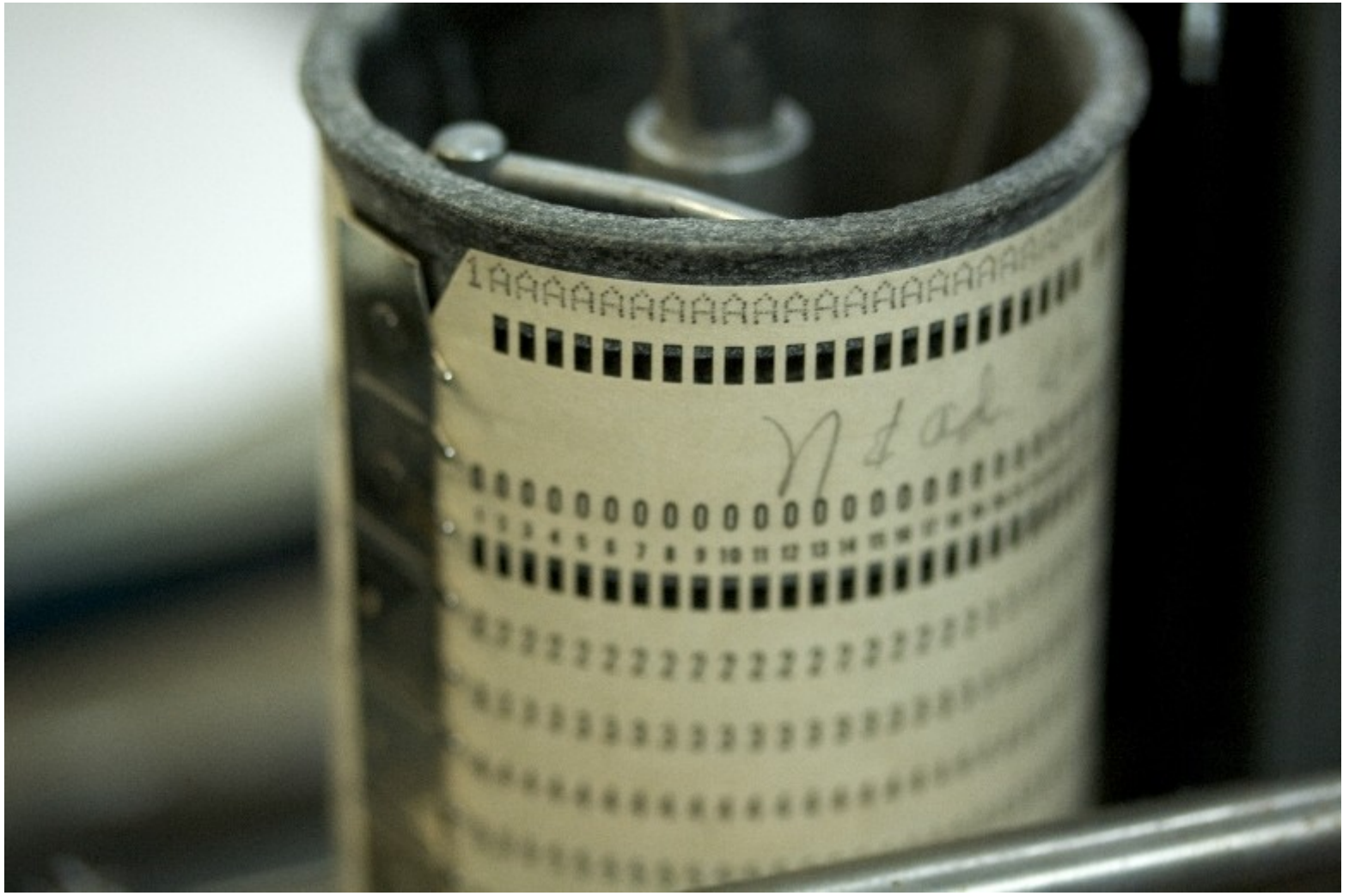
Kleines Rechenzentrum mit IBM/370

IBM 29 Kartenlocher

IBM 29 Tastatur

Fraunhofer FOKUS

IBM 29 Steuertrommel



Fraunhofer FOKUS

1979 MC68000, 1980 UNOS auf 68000

- **EXORmacs MC68000, 8 MHz, 16MB Hauptspeicher**
  - **Originalbetriebssystem: VERSADOS, Motorola**
  - **UNIX-Klon: UNOS, Charles River Datasystems**
  - **UNOS war realzeitfähig mit voll preemptiven Kern**
    - **Verwendung z.B. bei Raketensteuerungen**
  - **ca. 1983 MC68010, 8 MHz**
  - **1984: MC68010, 10 MHz mit VBERTOS**

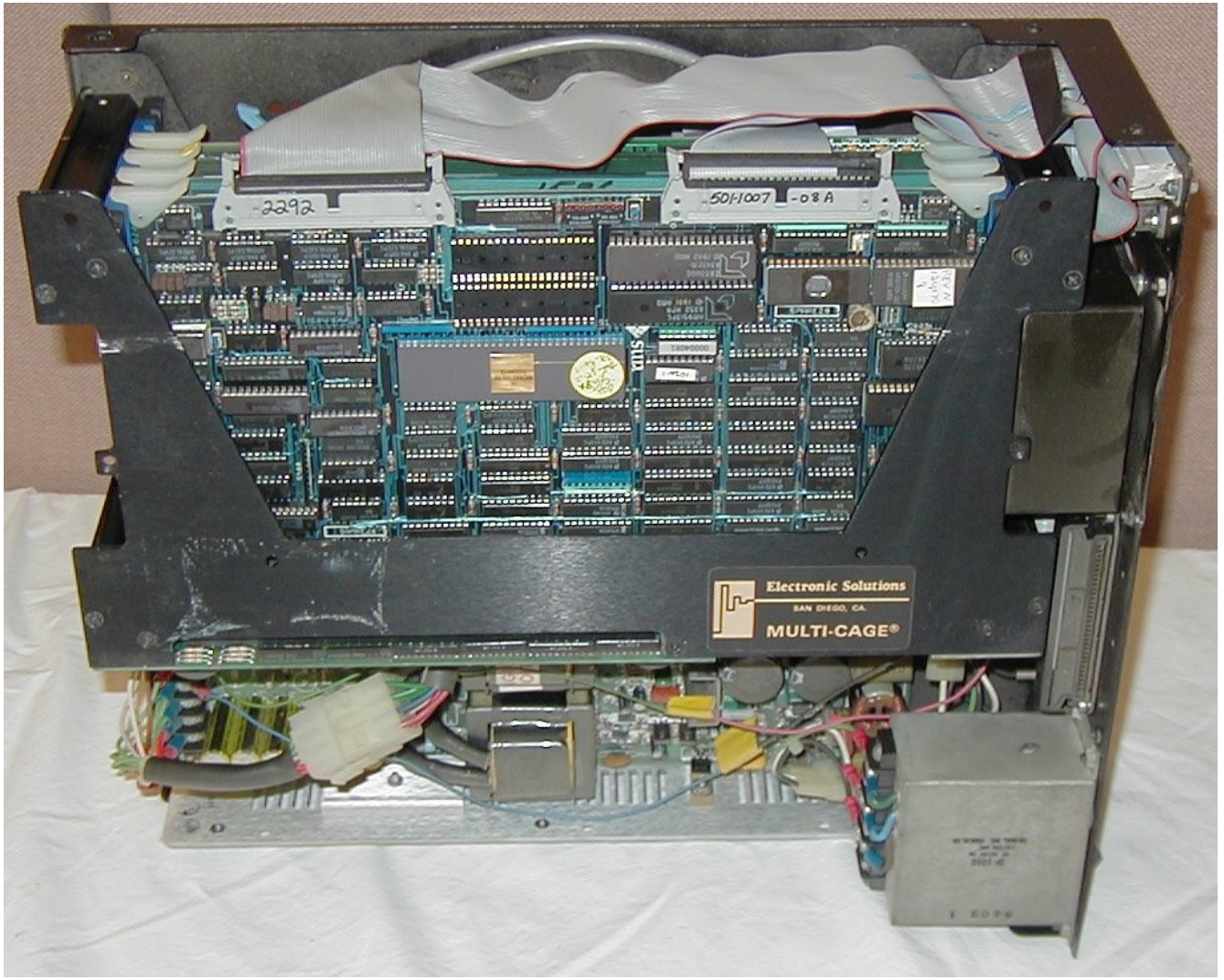
Erste Sun Systeme

- 1982 Sun-1 mit 10 MHz MC68000 und UNISOFT UNIX
  - Durch MC68000 Design-Bug kein virtuelles UNIX
  - Design Bug erfordert auch Änderung v. Bourne Shell
- Januar 1985 erste Sun-1 mit 10 MHz MC68010 SunOS
  - Auf Initiative der H.Berhold AG erste Sun in Europa
  - Damals 10-20 Nutzer bei H.Berthold AG
- August 1985 Sun-2/50 mit MC68010 10 Mhz
- 24.12. 1985 Sun-3/75 mit MC68020 20 MHz
- Seit 1982 alle Suns mit 1152x900 Bildschirmauflösung

Sun-1

Sun-2/120

Sun Multibus Gehäuse



Sun-2/50


Sun 2/50 Workstation
The first workstation for under $10,000
Ca: 1983

Was ist ein Filesystem?

- Dateien
  - Dateien können am Namen identifiziert werden
- Subdirectories
  - Subdirectories ermöglichen beliebige Strukturen
- Zugriffsrechte
  - Zugriffsrechte sind wichtig für Multi-User Systeme
  - Hier müssen Dateien Eigentümer haben
- Meta-Daten

Was ist ein Filesystem?

- Meta Daten
  - Inode Nummer
  - Filesystem-ID
  - Dateigröße
  - Zeitstempel
  - Dateityp
    - Typ spezifische Informationen (z.B. major/minor)
  - Linkinformationen
  - Zugriffsrechte
  - Access Control Listen
  - Extended Attribute

Betriebssysteme und ihre Filesysteme

- 1965 Multics mit hierarchischem Filesystem
- 1967 CP/CMS auf IBM/360 nur Dateien
- ca. 1972 UNIX mit hierarchischem Filesystem
- 1973 CPM nur Dateien
- 1979 VMS mit hierachischem Filesystem
- 1981 BSD mit FFS/UFS mit langen Dateinamen

  1981 MS-DOS nur Dateien
- 1983 MS-DOS-2.0 mit hierarchischem Filesystem

Verallgemeinerungen

- ca. 1982 BSD/SunOS bekommen die Network Disk
- ca. 1984 SunOS bekommt NFS
- 1985 SunOS-3.0 bekommt VFS und damit NFS-root
- ca. 1986 BSD-tahoe letzter Code (incl. VFS) von SunOS

- Zuvor: Nur ein Filesystem im Betriebssystem
  - Filesystem und BS eng miteinander verwoben
  - Keine klare Schnittstelle

Dateisystem-Basis-Kommandos

- Datei erzeugen
- Datei löschen
- Datei öffnen
- Datei schließen
- Lesen
- Schreiben
- Seek (Lese- Schreib-Offset ändern)
- Directory (oder Spezialdatei) erzeugen
- Directory lesen

Das VFS Interface

- Erstes VFS Interface 1985 mit SunOS-3.0
- Eine Struktur mit Daten für jedes VFS
- Darin auch eine Struktur mit Funktionspointern
  - Vfs-ops Filesystem Operationen
  - Eine Funktion liefert den Root-Vnode
- Eine Struktur mit Daten für jeden Vnode
- Darin auch eine Struktur mit Funktionspointern
  - Vnode-ops Datei Operationen
- Und ein Pointer auf die VFS Struktur von diesem FS

Die VFS-OPS

- vfs_mount() mountet ein Filesystem
- vfs_unmount() entfernt ein gemountetes Filesystem
- vfs_root() liefert einen VNODE Pointer zur Rootdirectory
- vfs_statvfs() liefert Statisktiken
- vfs_sync() synchronisiert den Hintergrundspeicher
- vfs_vget() liefert einen VNODE Pointer zur NFS-File-ID
- vfs_mountroot() mountet das Filesystem als Root-Dir
  - Erlaubt „Diskless“ Klienten
- vfs_freevfs() allozierte Daten zum VFS freigeben

Die VNODE-OPS

- vop_open() öffnet Datei
- vop_close() schließt Datei
- vop_read() liest von Datei
- vop_write() schreibt in Datei
- vop_ioctl() ioctl() Einsprung
- vop_setfl() set filedescriptor Flags
- vop_getattr() Interface für stat()/lstat()
- vop_setattr() Interface für diverse Metadaten, z.B. chown
- vop_access() Prüfung der Zugriffsrechte UNIX+ACL

Die VNODE-OPS

- vop_lookup() prüfen ob Dateiname existiert
- vop_create() neue Datei erzeugen
- vop_remove() Datei löschen
- vop_link() Hardlink anlegen
- vop_rename() Datei umbenennen
- vop_mkdir() Directory anlegen
- vop_rmdir() Directory löschen
- vop_readdir() Dateinamen aus Directory lesen

Die VNODE-OPS

- vop_symlink() Symlink anlegen
- vop_readlink() Symlink Ziel auslesen
- vop_fsync() Hintergrundspeicher f. Datei synchronisieren
- vop_inactive() vnode freigeben
- vop_fid() NFS Filehandle für Datei erzeugen
- vop_rwlock() read/write Lock setzen
- vop_rwunlock() read/write Lock löschen
- vop_seek() Datei-Offset holen oder ändern
- vop_cmp() 2 vnodes vergleichen

Die VNODE-OPS

- vop_frlock() flock() Interface
- vop_space() Datei Preallocation Handling
- vop_realvp() Return real vnode Pointer bei FS Stack
- vop_getpage() Interface für Pagefault Handler
- vop_putpage() Interface für Pagefault Handler
- vop_map() mmap() Interface
- vop_addmap()
- vop_delmap() munpacp() Interface
- vop_poll() poll() Interface

Die VNODE-OPS

- vop_dup() Kernel Core Dump auf Gerät schreiben
- vop_pathconf() POSIX pathconf() Interface
- vop_pageio() Erweiterung für getpage/putpage
- vop_dumpctl() Dump Management alloc/... ZFS
- vop_dispose() Freigabe von Platz aus vop_dumpctl()
- vop_getsecattr() ACL Interface
- vop_setsecattr() ACL Interface
- vop_shrlock() SMB Locking Interface
- vop_vnevent() Solaris 11 Event interface

- Ein Server
- Mehrere Klienten
- Unempfindlich gegen Absturz des Servers
  - Statuslos (NFSv1..NFSv3), Statusarm (NFSv4)
- UNIX Semantik auf dem Klienten
- Protokoll unabhängig von UNIX
- Kommunikation mittel NFS-Filehandle
  - Eine Kombination aus FS-ID, Inode-Nummer, Gen-#
  - Gen-# wird bei kritischen Operationen inkrementiert

- Unabhängig von Byteorder und Wortgröße der CPU
  - Implementierung mittels „XDR“
  - XDR: eXternal Data Representation
- Implementierung mittels RPC (Remote Procedure Call)
- Frühe NFS Protokolle mit UNIX Zugriffsrechten
  - Server und Klient müssen identische UIDs haben
  - NFSv4 verwendet Benutzer-Namen u. Mapper
- Seit 1988 Zugriffssicherung auch mit Kerberos
  - Seit Solaris 10 Kerberos auch außerhalb der USA

Beispiel: Klient-Zugriff auf eine NFS Datei

- NFS Filehandle für Mount-Punkt vom Server holen
- Mount-Punkt lokal mounten
- Datei auf Server suchen
  - Durch eine Folge von readdir/lookup
- Open: NFS-Filehandle für Datei --> lokaler Filedeskriptor
  - NFS-Filehandle in lokaler vnode Struktur speichern
- Write: zunächst in lokalen Cache
  - Danach sync zum Server
  - Auf dem Server: FH->vnode, open/seek/write/close

Filesystem Entwicklung seit 1980

- 197x AT&T „V7 Filesystem“ Namen max. 14 Buchstaben
- 1981-1982 BSD Namen bis zu 255 Buchstaben & readdir(), FFS
- 1988 SunOS-4.0 mmap() auf Files
- 1990 SunOS-4.1 Write-Klustering und Clean-Flag
- 1993 Solaris-2.4 ACLs (undokumentiert in Solaris-2.3)
- 1997 Solaris-2.7 UFS Logging
- 2001 Solaris-2.9 Extended Attributes
- 2000 Solaris ZFS mit COW und NTFS ACLs, Entwicklung startet
  - 2001 ZFS Entwickler müssen $HOME auf ZFS haben
  - 2005 ZFS wird veröffentlicht

Danke!

- PDF dieser Vorlesung: http://cdrecord.org/Files/