

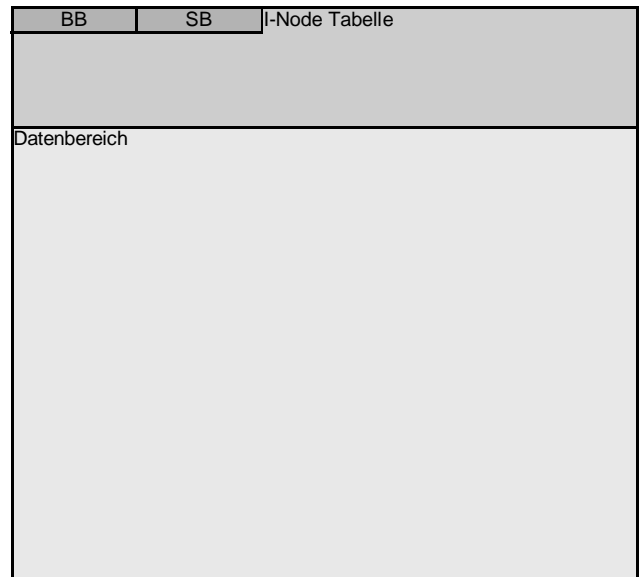
Name:	Lehrgang: Betriebssysteme	Datum:
Arbeitsblatt Nr.	Das Dateisystem von System V UNIX	Seite 1 von 2

Das Dateisystem System V UNIX von AT&T stellt in seiner Implementation die „Mutter“ aller UNIX Dateisysteme dar. Der Aufbau des Dateisystems entspricht in seiner Grobstruktur nebenstehender Grafik.

Im physikalisch ersten Block der Festplatte befindet sich der Bootblock (BB). Im Bootblock befindet sich der Bootloader zum Laden und Starten des Betriebssystems.

Der Superblock (SB), der physikalisch zweite Block, enthält Metainformationen über das gesamte Dateisystem

- Größe in Blöcken
- Größe eines logischen Blocks
- Name des Datenträgers
- Name der Dateistruktur
- mögliche Anzahl der Inodes
- tatsächliche Anzahl der Inodes
- erster freier Inode
- erster freier Datenblock
- Zustand des Dateisystems



Bei zerstörtem Superblock ist kein Zugriff auf die Daten der Festplatte möglich.

Nach dem Superblock folgt die Inode Tabelle. Der Begriff *Inode* entsteht aus der Zusammenziehung des Wortes *Information Node*. Die Größe der Inode Table wird beim Anlegen des Dateisystems mit dem Befehl `mkfs` festgelegt. Sie belegt i.d.R. ca. 2% der Kapazität des Datenträgers. Durch Angabe entsprechender Parameter läßt sich die Inode Dichte (d.i. die Größe eines logischen Blocks und damit verbunden die erforderliche Anzahl von Inodes) modifizieren. An die Inode Table schließt sich der Datenbereich der Festplatte an. In diesen Blöcken sind die eigentlichen Dateiinhalte gespeichert.

Die Inodes sind *die* zentrale Informationseinheit für den Zugriff auf die Daten im Datenbereich. Der Aufbau eines Inodes beim Dateisystem System V UNIX ist durch die folgende C-Struktur definiert:

```

/*
 * Inode structure as it appears on
 * a disk block.
 */
struct dinode
{
    unsigned short di_mode; /* mode and type of file */
    short di_nlink; /* number of links to file */
    short di_uid; /* owner's user id */
    short di_gid; /* owner's group id */
    off_t di_size; /* number of bytes in file */
    char di_addr[40]; /* disk block addresses */
    time_t di_atime; /* time last accessed */
    time_t di_mtime; /* time last modified */
    time_t di_ctime; /* time created */
};
#define INOPB 8 /* 8 inodes per block */
/*
 * the 40 address bytes:
 * 39 used; 13 addresses
 * of 3 bytes each.
 */

```

In einem Inode sind alle Informationen zu einer Datei abgelegt. Zu jeder Datei existiert ein Inode. Wo sich nun eine Datei physikalisch auf der Festplatte befindet, wird in dem Array `char di_addr[40]` gespeichert.

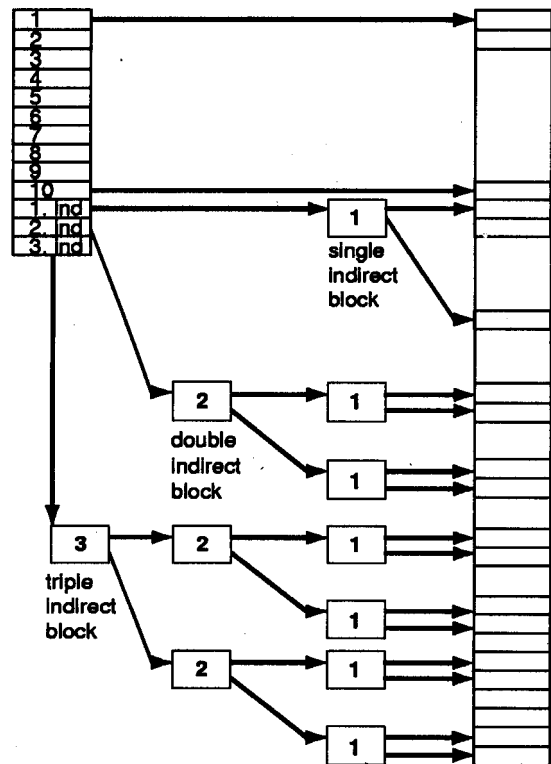
Ein Inode stellt 40 Bytes zur Speicherung von Adressen von Datenblöcken zur Verfügung. Eine Adresse eines Blocks belegt dabei 3 Bytes, so daß ein Dateisystem bei 512 Bytes Blockgröße insgesamt 8 GB groß sein darf.

In der Inode werden die Adressen der ersten zehn Datenblöcke direkt gespeichert (*direct blocks*). Die elfte Adresse ist die Adresse eines *indirect block*, der die Adressen von 128 weiteren Datenblöcken enthält. Entsprechend ist die zwölfte Adresse die Adresse eines *double indirect block*, der die Adressen von 128 Indirect Blocks enthält, und die dreizehnte Adresse ist die Adresse eines *triple indirect block*, der die Adressen von 128 double Indirect Blocks enthält.

Hierdurch sind die ersten zehn Blöcke einer Datei direkt zugreifbar. Erst wenn eine Datei größer als zehn Blöcke ist, muß auf diese verzeigerte Liste zurück gegriffen werden. In real existierenden Dateisystemen ist durch dieses Konzept für eine Vielzahl von Dateien ein schneller Zugriff gewährleistet.

Mit den Inodes stehen nun alle Verwaltungsinformationen zur Verfügung. Es fehlt jetzt allerdings noch die Möglichkeit, einen Dateinamen anzugeben. Auch fehlt noch die Möglichkeit einer Strukturbildung in Form von Verzeichnissen.

Verzeichnisse sind Dateien eines bestimmten Typs, der im ersten Byte des Inodes definiert ist. Der Aufbau dieser Verzeichnis-Dateien ist innerhalb des Dateisystems festgelegt: In den Verzeichnis-Dateien sind die Dateinamen und eventuelle Unterverzeichnisnamen gespeichert.



Eine Verzeichnis-Datei besteht aus einer Folge von 16 Byte langen Datensätzen. In den ersten beiden Bytes ist für jede Datei die zugehörige Inode-Nummer gespeichert. In den folgenden 14 Bytes ist der Name der dazugehörigen Datei abgelegt. Ist der Name kürzer als 14 Zeichen, wird er durch Null-Bytes aufgefüllt.

Wird in einem Verzeichnis ein Unterverzeichnis eingetragen, verweist die Inode dieses Unterverzeichniseintrags wiederum auf eine neue Verzeichnisdatei für dieses Unterverzeichnis.

Ausgehend vom Root-Verzeichnis oder vom aktuellen Verzeichnis ist so jeder Datei unter ihrem Namen zugreifbar.

Da als Ordnungskriterium die Inode dient, ist es möglich, mehr als einen Verzeichniseintrag für eine Datei zu haben. So ist es unter UNIX möglich sogenannte „Links“ zu erzeugen. Beide Verzeichniseinträge verweisen auf die gleiche Inode. Beide Namenseinträge sind gleichberechtigt. Die Rechte an solch einer Datei sind ebenfalls identisch.

Inode	Dateiname
16	.(Punkt)
17	..(Punkt Punkt)
36	.profile
118	.xconfig

Verzeichnisdatei