Deutsch

Fujitsu Software BS2000

POSIX
BS2000-Dateisystem bs2fs

Benutzerhandbuch

Stand der Beschreibung:
POSIX A49
BS2000 V21.0B

Ausgabe August 2025

Kritik… Anregungen… Korrekturen…

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Sie können uns Ihre Kommentare per E-Mail an bs2000.info@fujitsu.com senden.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2015

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001: 2015 erfüllt.

Copyright und Handelsmarken

Inhaltsverzeichnis

POSIX_bs2fs_de

1 Einleitung

Das BS2000-Dateisystem *bs2fs* ermöglicht den direkten und transparenten Zugriff auf BS2000-Dateien unter POSIX. Somit können sowohl "einfache" DVS-Dateien als auch Elemente von PLAM-Bibliotheken unter POSIX so bearbeitet werden, als ob es sich um POSIX-Dateien handelte.

Das BS2000-Dateisystem *bs2fs* wird seit POSIX V7.0 unterstützt.

1.1 Zielgruppen des Handbuchs

Das Handbuch richtet sich an alle Anwender des BS2000-Dateisystems *bs2fs*. Die Arbeit mit diesem Handbuch setzt voraus, dass Sie über Kenntnisse der Betriebssysteme UNIX und BS2000 verfügen und das Handbuch "POSIX-Grundlagen" vorliegen haben.

1.2 Wegweiser durch das Handbuch

Kapitel 2 und 3 geben einen Überblick über bs2fs-Dateisysteme, ihre Einbettung im POSIX und das Arbeiten mit bs2fs-Dateisystemen.

Kapitel 4 enthält die Beschreibung der Kommandos, Dämonen und der Verwaltungsdateien.

Kapitel 5 beschreibt den Zugriff auf bs2fs-Dateisysteme über NFS.

Im Kapitel 6 finden Sie Ratschläge zur Fehlerdiagnose und zur Fehlerbehebung sowie Maßnahmen, die Sie zur Leistungsverbesserung von bs2fs durchführen können.

Das Handbuch enthält ein Fachwort- und Literaturverzeichnis.

1.3 Änderungen gegenüber der vorigen Handbuchausgabe

Die Ausgabe dieses Handbuches enthält gegenüber der letzten Ausgabe (Bestellnummer: U41802-J-Z125-1) folgende Änderungen:

Unterstützung von Zugriffen auf bs2fs-Dateisysteme über NFS

Wiederherstellung von bs2fs-Dateien, die aufgrund von Fehlern beim Zurückschreiben ins bs2fs_lost+found-Verzeichnis verschoben wurden.

1.4 Darstellungsmittel

In diesem Handbuch werden folgende Darstellungsmittel verwendet:

Im Fließtext

kursive Schrift

Alle Elemente der Syntax sowie sonstige Dateinamen, Pfadnamen und Kommandos sind in *kursiver* Schrift dargestellt.

> **i** ...

Dieses Symbol kennzeichnet wichtige Hinweise und Ausnahmen, die Sie beachten sollten.

In der Syntax

normale Schrift

Variablen: Diese Zeichen sind Stellvertreter für andere Zeichen, die Sie auswählen und eingeben.

halbfette Schrift

Konstanten: Diese Zeichen sind direkt so einzugeben, wie sie gedruckt sind.

`[ ]`

Optional: Alles, was zwischen den eckigen Klammern steht, können, aber müssen Sie nicht eingeben. Die eckigen Klammern selbst dürfen Sie nicht eingeben, es sei denn, es ist ausdrücklich angegeben.

`' '`

Ein Leerzeichen, das Sie eingeben müssen.

`...`

Der vorhergehende Ausdruck kann wiederholt werden. Falls zwischen den Wiederholungen Leerzeichen eingegeben werden müssen, die nicht im Ausdruck enthalten sind, steht vor den Punkten ein Leerzeichen (`' '`).

`|`

Der senkrechte Strich trennt Alternativangaben.

In Beispielen

`halbfette Schreibmaschinenschrift`

Eingaben: Eingabezeilen werden bei Zeichenterminals mit der RETURN-Taste, bei Blockterminals mit der Tastenfolge EM DUE abgeschlossen, die Tastenangaben entfallen deshalb.

`normale Schreibmaschinenschrift`

Ausgaben

2 Überblick und Einbettung in POSIX

In diesem Kapitel erhalten Sie einen Überblick über bs2fs-Dateisysteme. Sie erfahren, wie bs2fs-Dateisysteme ins POSIX eingebettet sind und welche Sicherheitsmechanismen Sie für den Schutz von Dateien in bs2fs-Dateisystemen benutzen können.

Außerdem enthält dieses Kapitel eine Sammlung von Beispielsitzungen.

2.1 Konzept und Einbettung des bs2fs-Dateisystems in POSIX

Das bs2fs-Dateisystem ermöglicht es, mit POSIX-Schnittstellen (Kommandos und Programmschnittstellen) auf BS2000-Dateien zuzugreifen.

Hierzu muss ein Anwender die Menge der Dateien, mit denen er arbeiten will, spezifizieren (per BS2000-Wildcard-Syntax) und sich diese Dateien (vom Systemadministrator) in POSIX als bs2fs-Dateisystem einhängen lassen (*mount*). Durch diesen *mount*-Vorgang werden diese BS2000-Dateien dem Anwender im POSIX zugänglich gemacht. Danach können diese Dateien im bs2fs-Dateisystem mit POSIX-Kommandos oder aus POSIX-Programmen heraus bearbeit werden.

Um diese Zugriffe zu ermöglichen, kopiert ein Hintergrundprozess (Dämon) die betroffenen Dateien beim ersten Zugriff im bs2fs-Dateisystem (erster open) aus dem BS2000 in ein spezielles ufs-Dateisystem in POSIX, das nur zu diesem Zweck eingehängt wurde (bs2fs-Container). Auf diese im bs2fs-Container temporär abgelegte Datei darf nur das System zugreifen. Der Zugriff durch einen Anwender erfolgt nur auf die unterhalb des Einhängepunkts im bs2fs-Dateisystem eingehängte Datei, diesen Zugriff lenkt das System auf die im bs2fs-Container abgelegte Datei um.

Bei schreibenden Zugriffen wird die Datei im BS2000 für andere Benutzer gesperrt, nicht aber die bs2fs-Datei für andere POSIX-Nutzer. Nach dem Abschluss der Verarbeitung im bs2fs-Dateisystem überträgt ein Dämon die Datei wieder zurück ins BS2000. Danach ist sie auch dort wieder für andere BS2000-Benutzer zugreifbar. Solange nur Auskunftfunktionen wie *ls* ausgeführt werden, bewirkt dies noch keine Kopieraktion durch einen bs2fsd-Dämonen. Das *ls*-Kommando gibt lediglich die im BS2000 mit FSTAT ermittelten Dateien als POSIX-Pfadnamen ab dem bs2fs-Einhängepunkt aus.

Zusammenfassend bringt der Einsatz des bs2fs-Dateisystems also den Vorteil, dass der Anwender nicht mehr jede einzelne Datei vom BS2000 in das POSIX-Dateisystem kopieren muss (z.B. mit *bs2cp*), um diese mit POSIX-Mitteln bearbeiten zu können. Er muss lediglich die gewünschte BS2000-Dateimenge festlegen und sich diese vom Systemadministrator einhängen lassen. Bei der festgelegten Dateimenge kann es sich sowohl um bereits existierende als auch um später neu zu erstellende Dateien handeln. Der Transfer zwischen BS2000 und POSIX und umgekehrt wird unsichtbar für den Anwender von Kopierdämonen durchgeführt, sobald eine Datei geöffnet wird oder wenn eine schreibende Verarbeitung abgeschlossen ist.

Der Einsatz von bs2fs-Dateisysteme bietet z.B. folgende Möglichkeiten:

- BS2000-Dateien bzw. Elemente von PLAM-Bibliotheken können mit dem POSIX-Kommando *grep* nach bestimmten Mustern durchsucht werden.
- Zur effizienten Erzeugung von Programmen oder Programmsystemen kann *make* genutzt werden.
- Geschachtelte Prozeduren, bei denen ein mehrmaliger Wechsel zwischen BS2000-Kommando-Ebene und der Shell erfolgt, können durch reine POSIX-Shell-Scripts ersetzt werden, wenn die benötigten BS2000-Dateien vorher in ein bs2fs-Dateisystem eingehängt werden.

2.2 Der Einsatz des bs2fs-Dateisystems im Überblick

Mit dem bs2fs-Dateisystem können Sie BS2000-Dateien für die Bearbeitung unter POSIX bereitstellen und diese bearbeiten, als ob es sich um POSIX-Dateien handelte.

2.2.1 Bereitstellung von Dateien im bs2fs-Dateisystem

Das bs2fs-Dateisystem ist ein hierarchisches Dateisystem wie z.B. das UNIX-Dateisystem, das aus Dateiverzeichnissen und Dateien besteht.

Damit BS2000-Dateien in einem bs2fs-Dateisystem bearbeitet werden können, sind folgende Schritte durch den Systemadministrator erforderlich:

- Anlegen des bs2fs-Containers
- Einhängen des bs2fs-Containers (*mount*)
- Einhängen des bs2fs-Dateisystems (*mount*)

2.2.2 Darstellung von Dateien im bs2fs-Dateisystem

Die Darstellung von BS2000-Dateien in einem bs2fs-Dateisystem hängt davon ab, ob es sich um "einfache" DVS-Dateien oder PLAM-Bibliotheken handelt:

- DVS-Dateien werden in POSIX als Dateien dargestellt.
- PLAM-Bibliotheken werden in POSIX als Datei-Verzeichnisse mit fest vorgegebener zweistufiger Hierarchie dargestellt.

Die Namen dieser Dateien und Verzeichnisse sind identisch zu den Namen der entsprechenden DVS-Dateien, Bibliothekselemente oder PLAM-Bibliotheken. Allerdings gibt es im BS2000 bei Dateinamen keine Unterscheidung zwischen Groß- und Kleinschreibung.

Für die Schreibweise von Dateinamen im bs2fs-Dateisystem gilt:

- Die Ausgabe der Dateinamen erfolgt immer in Kleinbuchstaben.
- Die Eingabe kann im Allgemeinen in Groß- oder Kleinschreibung oder einer Mischung aus beiden erfolgen. Bei der Verwendung von wildcards in der Shell bzw. beim Kommando *find* müssen jedoch **ausschließlich Kleinbuchstaben** geschrieben werden, da sonst keine entsprechenden Dateinamen gefunden werden.
- Sonderzeichen der POSIX-Shell (z.B. '$' oder '*') müssen wie in der POSIX-Shell üblich bei der Eingabe explizit entwertet werden.

  Beispiele für die Entwertung eines BS2000-Dateinamens BS2DAT$$1:

  ```
  ls /home/bach/bs2fs1/bs2dat\$\$1
  ls /home/bach/bs2fs1/'bs2dat$$1'
  ```

2.2.3 Schnittstellen zur Verwaltung von bs2fs-Dateisystemen

Zur Verwaltung von bs2fs-Dateisystemen stehen Kommandos, Dämonen und Verwaltungsdateien zur Verfügung.

Kommandos

Die Kommandos *mount*, *mountall* und *umount*, *umountall* sind in POSIX-BC enthalten. Diese Kommandos können auch für das Ein- und Aushängen von bs2fs-Dateisystemen verwendet werden.

Dämonen

Dämonen sind Systemprozesse, die permanent und meist im Hintergrund laufen und die allgemeine Aufgaben durchführen. Der Kopierdämon *bs2fsd* transferiert Dateien aus dem BS2000 in den bs2fs-Container und umgekehrt. Beim Einhängen des bs2fs-Containers werden automatisch zwei *bs2fsd*-Dämonen gestartet.

Verwaltungsdateien

Die Verwaltungsdateien unterstützen die Verwaltung von Ressourcen. Sie enthalten entweder Informationen für den Benutzer, die mit Hilfe von Kommandos ausgegeben werden, oder Informationen für Kommandos, die entweder der Benutzer oder Kommandos in diese Dateien eingetragen hat. Für den Einsatz von bs2fs-Dateisystemen werden die Verwaltungsdateien */etc/mnttab* und */etc/vfstab* benutzt.

2.3 Sicherheitskonzept

Mit bs2fs werden BS2000-Objekte (DVS-Dateien, PLAM-Bibliotheken, PLAM-Typen, PLAM-Elemente) in Form eines Dateisystems für POSIX-Schnittstellen sichtbar gemacht.

Jedem POSIX-Benutzer sollen dabei genau dieselben Rechte zustehen, die ihm als BS2000-Benutzer zustehen. Jedes BS2000-Objekt soll gegenüber POSIX-Funktionen genau denselben Schutz genießen wie gegenüber BS2000-Funktionen.

Nur wer die Schutzattribute eines BS2000-Objekts mit BS2000-Mitteln ändern darf, soll dies auch mit POSIX-Mitteln in einem bs2fs-Dateisystem dürfen. Wer ein BS2000-Objekt mit BS2000-Mitteln lesen, schreiben oder ändern darf, soll dies auch mit POSIX-Mitteln in einem bs2fs-Dateisystem dürfen; wer ersteres nicht darf, soll auch letzteres nicht dürfen.

Bei bs2fs-Dateizugriffen dürfen die Rechte des POSIX-Benutzers in keinem Fall eine Erweiterung gegenüber seinen Rechten als BS2000-Benutzer darstellen.

Dieses Konzept führt dazu, dass beim Zugriff auf bs2fs-Dateien mit POSIX-Mitteln nicht immer die gleichen Möglichkeiten gegeben sind wie in der UNIX-Welt.

Ein Beispiel: Das Löschen von BS2000-Objekten mit BS2000-Mitteln erfordert das Schreibrecht. bs2fs-Dateien können deshalb mit POSIX-Mitteln (z.B. *rm*) auch nur gelöscht werden, wenn das Schreibattribut gesetzt ist. In der UNIX-Welt ist das Löschen von Dateien auch ohne Schreibattribut möglich.

2.4 Beispielsitzungen

Dieser Abschnitt enthält Beispiele für den Einsatz von bs2fs-Dateisystemen. Es wird gezeigt, wie Sie bs2fs-Dateisysteme einrichten und verwalten können und wie Sie auf die Dateien zugreifen können. Außerdem werden einige Beispiele für die Nutzung von bs2fs-Dateisystemen vorgestellt.

In diesen Beispielen werden zwei bs2fs-Dateisysteme eingesetzt.

2.4.1 Einführungsbeispiel zum Schnelleinstieg in das bs2fs-Dateisystem

In diesem Abschnitt erfahren Sie, wie eine Sitzung mit zwei bs2fs-Dateisystemen typischerweise abläuft:

Zunächst richtet der Systemadministrator den bs2fs-Container ein und hängt ihn ein. Anschließend hängt er zwei bs2fs-Dateisysteme für eine Benutzerkennung ein. Danach kann der Benutzer mit den Dateien der bs2fs-Dateisysteme arbeiten. Zum Abschluss hängt der Systemadministrator die bs2fs-Dateisysteme wieder aus.

Einrichten und Einhängen des bs2fs-Containers mit dem POSIX-Installationsprogramm

Mit der append-Funktion des POSIX-Installationsprogramms (Administrate POSIX filesystems) wird ein neues ufs-Dateisystem mit der Option *bs2fscontainer* erstellt und gleich mit `Automount=Y` eingehängt. Das Erstellen eines neuen ufs-Dateisystems bzw. das Überschreiben eines existierenden ufs-Dateisystems mit der Option *bs2fscontainer* bewirkt, dass dieses intern die Eigenschaft bs2fs-Container erhält, die den Einsatz des ufs-Dateisystems als bs2fs-Container erst ermöglicht.

```
                    Definition of BS2000 Container File

BS2000 filename:  :V70A:$SYSROOT.FS.BS2FSCONTAINER

BS2000 filesize:  1000000   PAM-Pages

POSIX filesystem? (y/n): Y
A new BS2000 container file has been created
================================================================================

                       Definition of POSIX filesystem

Size of filesystem: 100000    PAM-Pages     Journaling? (y/n): N

POSIX mountpoint: /bs2fscont
Automount? (y/n): Y          Mountoptions:  bs2fscontainer

Overwrite existing filesystem? (y/n):       POSIX filesystem marker (y/n): Y
No filesystem in BS2000 container yet
================================================================================
Save definitions:  ENTER
Help            :  F1                                        terminate: F2
```

Nach dem erfolgreichen Einhängen und der Beendigung des Installationsprogramms werden automatisch zwei Kopierdämonen *bs2fsd* gestartet.

```
# ps -efT|grep bs2fsd
    ROOT   163  74MR     1  0 15:29:55 ?         0:00 [bs2fsd]
    ROOT   162  74MQ     1  0 15:29:55 ?         0:00 [bs2fsd]
```

Zu diesem Zeitpunkt hat der bs2fs-Container noch keinen Inhalt:

```
# ls -l /bs2fscont
Gesamt 0
#
```

Alternative: Einhängen des bs2fs-Containers mit dem mount-Kommando

Ein einmal mit dem POSIX-Installationsprogramm erstelltes bs2fs-Containerdateisystem (Eigenschaft bs2fs-Container, siehe oben) kann auch mit dem *mount*-Kommando unter Angabe der Option *–o bs2fscontainer* eingehängt werden, z.B. mit

```
# mount –F ufs –o bs2fscontainer /bs2fscont
```

Auch in diesem Fall werden zwei Kopierdämonen automatisch gestartet.

> **Zusatzinformationen zum Einhängen des bs2fs-Containers:**
>
> Das ufs-Dateisystem, das als bs2fs-Container dient, wird als leeres Dateisystem erwartet. Falls es nicht leer ist, wird sein Inhalt beim Einhängen mit der Option *-o bs2fscontainer* gelöscht.
>
> Die Löschung erfolgt bei allen mount-Vorgängen: *mount*-Kommando, Automount beim POSIX-Start, Automount durch das Installationsprogramm.
>
> Die Eigenschaft bs2fs-Container wurde als Sicherheitsvorkehrung eingeführt, um das unbeabsichtigte Leeren eines ufs-Dateisystems beim Einhängen als bs2fs-Container zu verhindern.

Einhängen von bs2fs-Dateisystemen

Nachdem der bs2fs-Container eingehängt ist und die Kopierdämonen gestartet sind, können bs2fs-Dateisysteme eingehängt werden. Dies erfolgt – analog zu nfs-Dateisystemen – nicht mit dem POSIX-Installationsprogramm, sondern mit dem *mount*-Kommando.

Folgende Dateien unter der Benutzerkennung **`BACH`** und mit der Katalogkennung **`:V70A:`** sollen in bs2fs-Dateisystemen bereitgestellt werden:

1. Alle Dateien, deren Name dem Muster **`ASS.*.S`** entspricht, sollen unter **`/home/bach/bs2.1`** bereitgestellt werden
2. Alle Dateien, deren Name dem Muster **`PLAMLIB*`** entspricht, sollen unter **`/home/bach/bs2.2`** bereitgestellt werden

Es werden also zwei bs2fs-Dateisysteme benötigt, die mit folgenden Kommandos eingehängt werden können:

```
# mount –F bs2fs –o ftyp=text ':v70a:$bach.ass.*.s'  /home/bach/bs2.1
# mount –F bs2fs –o ftyp=text ':v70a:$bach.plamlib*' /home/bach/bs2.2
```

Dabei wird die Datei */etc/mnttab* automatisch durch das *mount*-Kommando um folgende Einträge erweitert:

```
:V70A:$BACH.ASS.*.S   /home/bach/bs2.1  bs2fs  ftyp=text  1196152657
:V70A:$BACH.PLAMLIB*  /home/bach/bs2.2  bs2fs  ftyp=text  1196152666
```

Im bs2fs-Container befinden sich nun die (noch leeren) Verzeichnisse, die Kopien der BS2000-Dateien der beiden bs2fs-Dateisysteme aufnehmen sollen. Reine Auskunftfunktionen wie das *ls*-Kommando bewirken noch keine Kopieraktion. Erst bei einem *open()* auf eine bs2fs-Datei wird der bs2fsd-Dämon aktiv. Details zur Struktur des bs2fs-Containers finden Sie in "Der Ein- und Aushängevorgang im Überblick".

```
# ls -l /bs2fscont
Gesamt 16
drwx--x--x   2 SYSROOT  SYSROOT     2048 Nov 27 11:40 V70A.BACH.1
drwx--x--x   2 SYSROOT  SYSROOT     2048 Nov 27 11:41 V70A.BACH.2
#
```

Die Schutzbit-Attribute des bs2fs-Containers sind durch den *mount*-Vorgang automatisch so gesetzt, dass sein Inhalt für nicht privilegierte Benutzer nicht sichtbar ist (`drwx--x--x`).

Arbeiten mit bs2fs-Dateisystemen

Nach dem Einhängen der beiden bs2fs-Dateisysteme kann das Ergebnis der Einhängevorgänge überprüft werden.

Zeigen Sie den Inhalt des unter **`/home/bach/bs2.1`** eingehängten Dateisystems an:

```
# ls -l /home/bach/bs2.1
Gesamt 3620
-rwx------   1 BACH     OS315      12288 Nov 22 14:54 ass.consio.s
-rwx------   1 BACH     OS315      18432 Nov 22 14:54 ass.lock.s
-rwx------   1 BACH     OS315       6144 Nov 22 14:54 ass.lockadm.s
-rwx------   1 BACH     OS315      18432 Nov 22 14:55 ass.lockwrk.s
-rwx------   1 BACH     OS315       2048 Nov 22 14:55 ass.posasto.s
-rwx------   1 BACH     OS315      18432 Nov 22 14:55 ass.posbs2fs.s
-rwx------   1 BACH     OS315     139264 Nov 22 14:56 ass.posbs2fx.s
...
-rwx------   1 BACH     OS315      69632 Nov 22 14:58 ass.posutil.s
-rwx------   1 BACH     OS315      51200 Nov 22 14:59 ass.posvert.s
-rwx------   1 BACH     OS315      49152 Nov 22 14:59 ass.posvmm.s
-rwx------   1 BACH     OS315       2048 Nov 22 14:59 ass.pprot.s
-rwx------   1 BACH     OS315      18432 Nov 22 15:00 ass.slpwkp.s
-rwx------   1 BACH     OS315      32768 Nov 22 15:00 ass.ttrap.s
-rwx------   1 BACH     OS315       2048 Nov 22 15:00 ass.uaddr.s
#
```

Unter **`/home/bach/bs2.1`** ist also eine Reihe von einfachen BS2000-Dateien eingehängt.

Zusatzinformationen zur Darstellung der bs2fs-Dateien:

Schutzbits

bilden entweder die BS2000-Standardattribute (USER-ACCESS, ACCESS) oder die BACL-Eigenschaften ab. In unserem Fall z.B. die Standardattribute USER-ACCESS=*USER-ONLY und ACCESS=*WRITE.

Eigentümer (USER)

ist immer die Kennung des bs2fs-mount.

Gruppe (GROUP)

ist immer die POSIX-Gruppe der Eigentümerkennung. Diese Gruppenzuordnung ist aber bei bs2fs-Zugriffen nicht relevant. Es gelten die BS2000-Gruppenzuordnungen mit / ohne SECOS. Siehe hierzu auch Kapitel 3.

Dateigröße

von nicht geöffneten Dateien ist ein Vielfaches von 2048 an belegten PAM-Seiten. Die kleinste Datei ist laut *ls*-Kommando also 2048 Byte groß.

Dateiname:

Namen werden in Kleinbuchstaben angezeigt. Dies gilt auch für Bibliotheks- und Elementnamen. Die Bezugnahme auf bs2fs-Objekte mit Shell-Kommandos oder C-Programmschnittstellen kann dagegen beliebig in Groß-/Kleinschreibung erfolgen.
Ausnahme: In Wildcard-Konstrukten in der Shell und mit dem Kommando *find* ist die Kleinschreibung erforderlich.

Zeigen Sie nun den Inhalt des zweiten unter `/home/bach/bs2.2` eingehängten Dateisystems an:

```
# ls -l /home/bach/bs2.2
Gesamt 5224
drwx------   2 BACH     OS315    1562624 Nov 19 17:39 plamlib.1
drwxrwxrwx   2 BACH     OS315    1112064 Nov 19 17:30 plamlib.2
```

Bei diesem Dateisystem handelt es sich um zwei PLAM-Bibliotheken, erkennbar an der Darstellung als Verzeichnis.

Lassen Sie sich den Inhalt der Bibliothek plamlib.1 anzeigen:

```
# ls -l /home/bach/bs2.2/plamlib.1
Gesamt 32
drwx------   2 BACH     OS315         48 Nov 19 17:39 d
drwx------   2 BACH     OS315         48 Nov 19 17:39 j
drwx------   2 BACH     OS315         48 Nov 19 17:39 l
drwx------   2 BACH     OS315         48 Nov 19 17:39 m
drwx------   2 BACH     OS315         48 Nov 19 17:39 p
drwx------   2 BACH     OS315       4032 Nov 19 17:39 s
drwx------   2 BACH     OS315         48 Nov 19 17:39 x
```

Das Bibliotheksverzeichnis enthält Unterverzeichnisse, deren Namen den Elementtypen (D, J, L, M, P, S, X) entsprechen.

> **Zusatzinformationen zu den Bibliotheks- und Typverzeichnissen:**
>
> Die Verzeichnisstruktur und die Verzeichnisattribute sind nicht veränderbar. Aktionen wie Löschen, Umbenennen, Neueinrichten, Attributänderung auf Bibliotheks- und Typverzeichnisse werden generell abgewiesen.
>
> Es kann also beispielsweise mit *mkdir* keine neue PLAM-Bibliothek eingerichtet werden.

Zeigen Sie nun die Elemente des Typs S an:

```
# ls -l /home/bach/bs2.2/plamlib.1/s
Gesamt 6024
-rwx------   2 BACH     OS315       8192 Nov 19 17:33 acct.c
-rwx------   2 BACH     OS315       8192 Nov 19 17:33 acct.c+001
-rwx------   2 BACH     OS315      51200 Nov 19 17:33 bio.c
-rwx------   2 BACH     OS315      51200 Nov 19 17:33 bio.c+001
-rwx------   2 BACH     OS315       8192 Nov 19 17:33 bitmap.c
-rwx------   2 BACH     OS315       8192 Nov 19 17:33 bitmap.c+001
-rwx------   2 BACH     OS315       4096 Nov 19 17:33 bitmasks.c
-rwx------   2 BACH     OS315       4096 Nov 19 17:33 bitmasks.c+001
-rwx------   2 BACH     OS315      12288 Nov 19 17:34 bs.c
-rwx------   2 BACH     OS315      12288 Nov 19 17:34 bs.c+001
...
-rwx------   2 BACH     OS315      14336 Nov 19 17:39 vm_pageout.c
-rwx------   2 BACH     OS315      14336 Nov 19 17:39 vm_pageout.c+001
-rwx------   2 BACH     OS315      18432 Nov 19 17:39 vm_pgout.c
-rwx------   2 BACH     OS315      18432 Nov 19 17:39 vm_pgout.c+001
-rwx------   2 BACH     OS315       6144 Nov 19 17:39 vm_subr.c
-rwx------   2 BACH     OS315       6144 Nov 19 17:39 vm_subr.c+001
-rwx------   2 BACH     OS315       4096 Nov 19 17:39 xmmu.c
-rwx------   2 BACH     OS315       4096 Nov 19 17:39 xmmu.c+001
-rwx------   2 BACH     OS315       8192 Nov 19 17:39 xsys.c
-rwx------   2 BACH     OS315       8192 Nov 19 17:39 xsys.c+001
#
```

Das Bibliotheksverzeichnis enthält einige Elemente des Typs S. Jedes dieser Elemente liegt im Beispiel mit der einzigen Version '001' vor. Daher sind für jedes Element zwei Einträge vorhanden: Ein Eintrag für den Elementnamen ohne Versionsbezeichnung und einer mit der (einzigen) Versionsbezeichnung.

Zur Erläuterung:

Im bs2fs-Dateisystem kann das Element mit der höchsten Version auch als Elementname ohne Version angesprochen werden (intern realisiert als Hard Link). Die Darstellung und der Bezug auf Elemente mit niedrigeren Versionen erfolgt dagegen immer mit Version. Siehe auch Kapitel 3.

Auf die Dateien der bs2fs-Dateisysteme können Sie jetzt z.B. POSIX-Kommandos anwenden:

```
# grep ETPND /home/bach/bs2.1/*.s
/home/bach/bs2.1/ass.consio.s:    ETPND   &MODNAME,VER=&MODVERS,PATCH=200
/home/bach/bs2.1/ass.lock.s:      TITLE    '&MODNAME - ETPND/XREF'
/home/bach/bs2.1/ass.lock.s:      ETPND    &MODNAME,VER=&MODVERS,PATCH=200
# tail /home/bach/bs2.1/ass.posvert.s
         HDRCHECK UNIT=228,FUNCT=(187,1)
         @BEND
         @BEND
         @EXIT
         @END
         EJECT
****************************** ETPND **********************************
         ETPND &MODNAME,VER=&MODVERS,PATCH=200
*
         END
#
```

Aushängen der bs2fs-Dateisysteme

Wenn die Dateien der bs2fs-Dateisysteme in POSIX nicht mehr benötigt werden, können die entsprechenden Dateisysteme wieder ausgehängt werden.

Um einzelne bs2fs-Dateisysteme auszuhängen, verwenden Sie das Kommando *umount*, in diesem Beispiel also:

`umount /home/bach/bs2.1`

und

`umount /home/bach/bs2.2`

Um alle bs2fs-Dateisysteme auszuhängen, können Sie das Kommando *umountall* verwenden:

entweder

`umountall –F bs2fs`

oder

`umountall –b`

Nach dem Aushängen mit *umount* oder *umountall* sind die entsprechenden Einträge in der Datei */etc/mnttab* gelöscht.

Automatisierung der bs2fs-mount-Vorgänge

Es gibt u.a. folgende Möglichkeiten, die bs2fs-*mount*-Vorgänge zu automatisieren:

1. Die *mount*-Kommandos in Shell-Scripts eintragen und diese Scripts zu einem beliebigen Zeitpunkt ausführen.
2. Einträge in der Dateisystemtabelle */etc/vfstab* mit `Automount=Y` erstellen.
   Die bs2fs-*mount*-Vorgänge werden dann beim Start des POSIX-Subsystems automatisch ausgeführt. Außerdem wird die Tabelle */etc/vfstab* vom *mountall*-Kommando ausgelesen, falls keine andere Dateisystemtabelle angegeben wird.

   Um alle bs2fs-Dateisysteme aufgrund der Einträge in der Tabelle */etc/vfstab* einzuhängen, können Sie das Kommando *mountall* verwenden:

entweder

```
mountall –F bs2fs
```

oder

```
mountall –b
```

3. Einträge in einer eigenen Dateisystemtabelle (analog zu */etc/vfstab*) erstellen. Diese Tabelle kann dann als Argument des Kommandos *mountall* verwendet werden.

   Beispiel: Eintrag in einer mountall-Tabelle erstellen (/etc/vfstab wäre analog)

   Um eine Datei zu erstellen, die bereits eine geeignete Zeile im vfstab-Format enthält, können Sie das Kommando *mount -p* nutzen:

```
# mount -p |grep /home/bach/bs2.1 >> /etc/bs2fstab
# cat /etc/bs2fstab
:V70A:$BACH.ASS.*.S - /bs2test/bs2fs bs2fs - no  ftyp=text
```

   Um das automatische Einhängen mit dem Kommando *mountall* zu ermöglichen, müssen Sie den Automount-Eintrag von `no` auf `yes` ändern, z.B. mit:

```
# edtu /etc/bs2fstab
:V70A:$BACH.ASS.*.S - /bs2test/bs2fs bs2fs - yes  ftyp=text
```

   Um alle bs2fs-Dateisysteme aufgrund der Einträge in der Tabelle */etc/bs2fstab* einzuhängen, können Sie das Kommando *mountall* verwenden:

   entweder

```
mountall –F bs2fs /etc/bs2fstab
```

   oder

```
mountall –b /etc/bs2fstab
```

2.4.2 Nutzungsszenarien für bs2fs-Dateisysteme

In diesem Abschnitt werden verschiedene Anwendungsmöglichkeiten von bs2fs-Dateisystemen dargestellt.

Beispiel 1: Durchsuchen von CONSLOG-Dateien

Die abgespeicherten CONSLOG-Dateien des Monats Juni im Jahr 2007 sollen nach bestimmten Inhalten durchsucht werden. Diese Suche kann sehr einfach mit dem Kommando *grep* durchgeführt werden, wenn die Dateien in einem bs2fs-Dateisystem zur Verfügung stehen.

In diesem Beispiel wird vorausgesetzt, dass der bs2fs-Container bereits eingerichtet ist.

Führen Sie die folgenden Schritte aus:

Hängen Sie die zu bearbeitenden BS2000-Dateien ein

```
# mount -F bs2fs ':V70a:$sysaudit.sys.conslog.2007-06*' /home/bs2.conslog
```

Überprüfen Sie das Ergebnis des Einhängevorgangs

```
# mount | grep 'bs2.conslog'
/home/bs2.conslog on :V70A:$SYSAUDIT.SYS.CONSLOG.2007-06* ftyp=text/nosuid on Tue Nov 27 13:
52:23 2007
```

oder

```
# df -k -F bs2fs | grep 'bs2.conslog'
:V70A:$SYSAUDIT.SYS.CONSLOG.2007-06*  2000000   331518  1668482    17% /home/bs2.conslog
```

Zeigen Sie die im bs2fs-Dateisystem bereitgestellten BS2000-Dateien an

```
# ls -l /home/bs2.conslog
Gesamt 11836
-r-x------   1 100      OTHER     151552 Jun 12 12:32 sys.conslog.2007-06-11.007.001
-r-x------   1 100      OTHER     256000 Jun 13 13:17 sys.conslog.2007-06-12.007.001
-r-x------   1 100      OTHER      75776 Jun 13 16:26 sys.conslog.2007-06-13.007.001
-r-x------   1 100      OTHER      73728 Jun 13 17:25 sys.conslog.2007-06-13.007.002
-r-x------   1 100      OTHER      77824 Jun 14 12:36 sys.conslog.2007-06-13.007.003
-r-x------   1 100      OTHER      77824 Jun 14 14:42 sys.conslog.2007-06-14.007.001
-r-x------   1 100      OTHER    5347328 Nov 14 11:12 sys.conslog.2007-06-14.007.002
#
```

Suchen Sie in den bereitgestellten Dateien nach Dateinamen mit dem Präfix *:V70A:$BACH.SEM*

```
# grep ':V70A:$BACH.SEM' /home/bs2.conslog/*
...
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MBW-000.163259  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MB0-000.163310  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MB0-000.163310  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MCH-000.170925  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MCH-000.170925  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MCL-000.170939  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MCL-000.170939  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MER-000.144635  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MER-000.144635  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0ME1-000.165231  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0ME1-000.165231  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MKI-000.133902  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MKI-000.133902  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MNC-000.130039  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: analyseresource after toupper <:V70A:$BACH.SEM*.C>
/home/bs2.conslog/sys.conslog.2007-06-14.007.002:  <C %0MNC-000.130039  %  POS1020 Message
of the POSIX kernel:WARNING: bs2_subr.c: couldbebs2fsname <:V70A:$BACH.SEM*.C>
#
```

Beispiel 2: Umwandlung von Kleinbuchstaben in Großbuchstaben

In einer BS2000-Datei sollen alle Kleinbuchstaben in Großbuchstaben umgewandelt werden. Für diese Umwandlung lässt sich das Kommando *tr* einsetzen, sofern die Dateien in einem bs2fs-Dateisystem zur Verfügung stehen.

In diesem Beispiel wird davon ausgegangen, dass die zu bearbeitende Datei wie in "Beispiel 1: Durchsuchen von CONSLOG-Dateien" gezeigt unter */home/bs2.conslog* bereitgestellt ist.

Zeigen Sie die im bs2fs-Dateisystem bereitgestellten BS2000-Dateien an

```
# ls -l /home/bs2.conslog
Gesamt 11836
-r-x------   1 100      OTHER     151552 Jun 12 12:32 sys.conslog.2007-06-11.007.001
-r-x------   1 100      OTHER     256000 Jun 13 13:17 sys.conslog.2007-06-12.007.001
-r-x------   1 100      OTHER      75776 Jun 13 16:26 sys.conslog.2007-06-13.007.001
-r-x------   1 100      OTHER      73728 Jun 13 17:25 sys.conslog.2007-06-13.007.002
-r-x------   1 100      OTHER      77824 Jun 14 12:36 sys.conslog.2007-06-13.007.003
-r-x------   1 100      OTHER      77824 Jun 14 14:42 sys.conslog.2007-06-14.007.001
-r-x------   1 100      OTHER    5347328 Nov 14 11:12 sys.conslog.2007-06-14.007.002
#
```

Wandeln Sie alle Kleinbuchstaben in der Datei *sys.conslog.2007-06-13.007.001* in Großbuchstaben um:

Schreiben Sie das Ergebnis in die Datei *conslog.out*. (Der Name der Eingabedatei wird eindeutig abgekürzt mit Wildcards angegeben.)

```
# tr '[:lower:]' '[:upper:]' </home/bs2.conslog/*13.007.001 >conslog.out
```

Überprüfen Sie das Ergebnis, indem Sie die letzten Sätze von Ein- und Ausgabedatei anzeigen:

```
# tail  /home/bs2.conslog/*13.007.001
  <C %0CYT-000.162451  %  POS1020 Message of the POSIX kernel:WARNING: vfs.c: generic mount 1
  <C %0CYT-000.162451  %  POS1020 Message of the POSIX kernel:WARNING: vfs.c: generic mount 2
  <C %0CYT-000.162451  %  POS1020 Message of the POSIX kernel:WARNING: vfs.c: generic mount 3
  <C %0CYT-000.162451  %  POS1020 Message of the POSIX kernel:WARNING: bs2_vfsops.c:
entering bs2fs_mount
  <C %0CYT-000.162451  %  POS1020 Message of the POSIX kernel:WARNING: bs2_subr.c:
couldbebs2fsname
  <0 %0CYT-000.162451  %  EXC0420 /LOGOFF PROCESSED. CPU TIME USED: 2.8544 SEC, USER ID:
TSOS, TASK ID: 1DDF00D6
  <0 %0CYU-000.162528  %  JMS0154 'TSOS' LOGGED ON FOR 'MCP0212C/STATION9'. JOB NAME
'BACHMANN'. CALLER 'TSN 0BEP'. TID 1C7E0082
  <0 %0CYU-000.162529  %  EXC0420 /LOGOFF PROCESSED. CPU TIME USED: 0.3710 SEC, USER ID:
TSOS, TASK ID: 1C7E0082
     /0B1Q-000.162609 CHANGE-CONSLOG PROCESSOR NAME: MCP0212C STATION NAME: STATIO10 AUDIT-
ID: 00000000000000000000000000000000
     TCLOG    .162609  ***2007-06-13*** 000004 **** UTC+02:00
************************************************************
# tail conslog.out
  <C %0CYT-000.162451  %  POS1020 MESSAGE OF THE POSIX KERNEL:WARNING: VFS.C: GENERIC MOUNT 1
  <C %0CYT-000.162451  %  POS1020 MESSAGE OF THE POSIX KERNEL:WARNING: VFS.C: GENERIC MOUNT 2
  <C %0CYT-000.162451  %  POS1020 MESSAGE OF THE POSIX KERNEL:WARNING: VFS.C: GENERIC MOUNT 3
  <C %0CYT-000.162451  %  POS1020 MESSAGE OF THE POSIX KERNEL:WARNING: BS2_VFSOPS.C:
ENTERING BS2FS_MOUNT
  <C %0CYT-000.162451  %  POS1020 MESSAGE OF THE POSIX KERNEL:WARNING: BS2_SUBR.C:
COULDBEBS2FSNAME
  <0 %0CYT-000.162451  %  EXC0420 /LOGOFF PROCESSED. CPU TIME USED: 2.8544 SEC, USER ID:
TSOS, TASK ID: 1DDF00D6
  <0 %0CYU-000.162528  %  JMS0154 'TSOS' LOGGED ON FOR 'MCP0212C/STATION9'. JOB NAME
'BACHMANN'. CALLER 'TSN 0BEP'. TID 1C7E0082
  <0 %0CYU-000.162529  %  EXC0420 /LOGOFF PROCESSED. CPU TIME USED: 0.3710 SEC, USER ID:
TSOS, TASK ID: 1C7E0082
     /0B1Q-000.162609 CHANGE-CONSLOG PROCESSOR NAME: MCP0212C STATION NAME: STATIO10 AUDIT-
ID: 00000000000000000000000000000000
     TCLOG    .162609  ***2007-06-13*** 000004 **** UTC+02:00
************************************************************
#
```

3 Einsatz von bs2fs-Dateisystemen

In diesem Kapitel erfahren Sie, wie Sie bs2fs-Dateisysteme verwalten und was Sie beim Arbeiten mit bs2fs-Dateisystemen beachten müssen.

3.1 Administrieren von bs2fs-Dateisystemen

Die Verwaltung von bs2fs-Dateisystemen umfasst folgende Aufgaben:

- Anlegen des bs2fs-Containers mit dem POSIX-Installationsprogramm
- Ein- und Aushängen des bs2fs-Containers
- Ein- und Aushängen von bs2fs-Dateisystemen
- Ändern von Verwaltungsdateien, wenn die Listen der automatisch bereitzustellenden bzw. einzuhängenden Ressourcen aktualisiert werden sollen
- Lokalisieren und Beseitigen von Problemen beim Einsatz von bs2fs-Dateisystemen (siehe "Diagnose und Leistungsverbesserung")

3.1.1 Einrichten und Einhängen des bs2fs-Containers

Für den Einsatz von bs2fs-Dateisystemen wird im POSIX ein spezieller Bereich benötigt, der temporär die Dateien von bs2fs-Dateisystemen aufnimmt: der bs2fs-Container. Dabei handelt es sich um ein ufs-Dateisystem, das explizit als bs2fs-Container gekennzeichnet ist. In einem POSIX-System kann es nur einen einzigen bs2fs-Container geben. Dieser nimmt alle Dateien aller eingehängten bs2fs-Dateisysteme auf.

Ein Dateisystem mit der Eigenschaft bs2fs-Container können Sie bei der Installation mit dem POSIX-Installationsprogramm anlegen.

Seine Größe sollte abhängig von der Größe der wirklich benutzten Dateien festgelegt werden. Da man davon ausgehen kann, dass nicht alle BS2000-Dateien gleichzeitig geöffnet und somit in den Container kopiert werden, könnte man eine Abschätzung für den Platzbedarf im Container z.B. so erhalten: Summe der Größen aller BS2000 Dateien, die mit bs2fs eingehängt werden multipliziert mit 0,5.

Einrichten des bs2fs-Containers mit dem POSIX-Installationsprogramm

Beim *append* eines neuen ufs-Dateisystems oder eines existierenden Dateisystems (mit

overwrite=yes) zusammen mit der Option *bs2fscontainer* erhält dieses Dateisystem intern

die Eigenschaft bs2fs-Container, die erst den Einsatz dieses Dateisystems als bs2fs-

Container ermöglicht.

Einhängen des bs2fs-Containers mit dem POSIX-Installationsprogramm

Wie jedes andere ufs-Dateisystem kann der bs2fs-Container (= ufs-Dateisystem mit der Eigenschaft bs2fscontainer) bei Angabe von *Automount=Y* in die Dateisystemhierarchie eingehängt werden. Der dabei entstehende Eintrag *Automount=Y* in der Dateisystemtabelle */etc/vfstab* bewirkt außerdem, dass das Dateisystem automatisch bei Start des POSIX-Subsystems eingehängt wird.

Einhängen der bs2fs-Containers im laufenden Betrieb mit dem mount-Kommando

Um den bs2fs-Container einzuhängen, geben Sie ein entsprechendes *mount*-Kommando mit der Option *-o bs2fscontainer* an, z. B.

```
mount -F ufs -o bs2fscontainer /dev/dsk/23 /home/bach/mount2
```

Voraussetzung für das Einhängen eines Dateisystems als bs2fs-Container ist, dass es wie unter "Einrichten des bs2fs-Containers mit dem POSIX-Installationsprogramm" beschrieben angelegt wurde. Andernfalls wird ein *mount*-Kommando mit der Option *-o bs2fscontainer* abgewiesen.

Beim Einhängen ist ein bs2fs-Container standardmäßig leer. Sollte er nicht leer sein, so wird sein Inhalt beim Einhängen gelöscht.

3.1.2 Einhängen von bs2fs-Dateisystemen

Mit dem Einhängen von bs2fs-Dateisystemen legen Sie fest, welche BS2000-Dateien an welcher Stelle in der POSIX-Dateisystemhierarchie bereitgestellt werden.

Ein bs2fs-Dateisystem können Sie erst einhängen, nachdem der bs2fs-Container eingehängt wurde.

Um ein bs2fs-Dateisystem einzuhängen, geben Sie ein entsprechendes *mount*-Kommando mit der Option *-F bs2fs* an, z. B.

```
mount -F bs2fs ':v70a:$bach.ass.*.s' /home/bach/bs2.1
```

3.1.3 Allgemeine Aspekte beim Ein- und Aushängen

Sowohl bs2fs-Dateisysteme als auch der bs2fs-Container können im laufenden Betrieb mit den Kommandos *mount* und *umount* ein- und ausgehängt werden. Allerdings muss dabei die folgende Reihenfolge eingehalten werden. Kommando-Eingaben, die von dieser Reihenfolge abweichen, führen zu Fehlern und Abbruch des entsprechenden Kommandos:

Reihenfolge beim Ein- und Aushängen

- Einhängen des bs2fs-Containers
  Das ufs-Dateisystem mit der Eigenschaft bs2fs-Container muss eingehängt werden, bevor das erste bs2fs-Dateisystem eingehängt werden kann.
- Ein- oder Aushängen von bs2fs-Dateisystemen
  bs2fs-Dateisysteme können ein- und wieder ausgehängt werden, so lange der bs2fs-Container eingehängt ist.
- Aushängen des bs2fs-Containers
  Das ufs-Dateisystem mit der Eigenschaft bs2fs-Container kann nur dann ausgehängt werden, wenn kein bs2fs-Dateisystem mehr eingehängt ist.

3.1.4 Automatisierung

bs2fs-Container

Die Verwaltung und damit auch die Automatisierung der *mount*/*umount*-Vorgänge erfolgt analog zu anderen POSIX-Dateisystemen vom Typ "ufs". Beim Einrichten des Dateisystems mit dem POSIX-Installationsprogramm (*append*-Funktion) wird automatisch ein Eintrag in der Tabelle */etc/vfstab* erstellt. Sofern `Automount=yes` vereinbart wurde, wird das Dateisystem gleich eingehängt. Der Eintrag `Automount=yes` in der Tabelle */etc/vfstab* bewirkt außerdem, dass das Dateisystem beim nächsten Start des POSIX-Subsystems automatisch eingehängt wird. Modifikationen in der Tabelle */etc/vfstab* können ebenfalls mit dem POSIX-Installationsprogramm erfolgen (*modify*-Funktion).

bs2fs-Dateisysteme

bs2fs-Dateisysteme können - ähnlich wie nfs-Dateisysteme - nicht mit dem POSIX-Installationsprogramm verwaltet werden. Zur Automatisierung der *mount*-Vorgänge muss der Administrator Systemtabellen selbst edititieren oder Scripts aufbauen.

Überblick über die verschiedenen Automatisierungsmöglichkeiten:

1. *mount*-Kommandos in Shell-Scripts eintragen und diese zu einem beliebigen Zeitpunkt ausführen
2. *mount*-Kommandos in Shell-Scripts vom Typ "rc" eintragen. Diese werden dann automatisch bei Start des POSIX-Subsystems ausgeführt.
3. Die gewünschten bs2fs-Dateisysteme im vfstab-Format (mit Automount=yes) in eine eigene Dateisystemtabelle eintragen. Diese Tabelle kann dann als Argument des Kommandos *mountall* verwendet werden.
4. Die gewünschten bs2fs-Dateisysteme (mit Automount=yes) in die Tabelle */etc/vfstab* eintragen. Diese Tabelle wird durch das Kommando *mountall* verarbeitet, wenn keine Dateisystemtabelle angegeben wird. Beim Start des POSIX-Subsystems werden die *mount*-Vorgänge automatisch ausgeführt.

Empfehlung: vfstab-gerechte Einträge können mit *mount –p* auf ein bereits erfolgreich eingehängtes bs2fs-Dateisystem erstellt werden.

Ein Beispiel hierzu finden Sie in "Einführungsbeispiel zum Schnelleinstieg in das bs2fs-Dateisystem".

Starten und Beenden von POSIX

Wenn die *mount*/*umount*-Vorgänge beim Starten und Beenden von POSIX automatisch erfolgen, hat dies den Vorteil, dass jeweils die notwendige Reihenfolge eingehalten wird.

Starten: ufs-Dateisysteme, bs2fs-Dateisysteme, nfs-Dateisysteme

Beenden: nfs-Dateisysteme, bs2fs-Dateisysteme, ufs-Dateisysteme

Wenn der Administrator die Dateisysteme explizit ein- und aushängen möchte, muss er diese Reihenfolgen beachten.

3.2 Arbeiten mit dem bs2fs-Dateisystem

In diesem Abschnitt erfahren Sie, welche Arten von Dateien und Bibliothekselementen Sie in einem bs2fs-Dateisystem bearbeiten können, welche Zugriffsrechte in einem bs2fs-Dateisystem gelten und welche Besonderheiten Sie beim Arbeiten mit bs2fs-Dateisystemen beachten müssen.

3.2.1 Unterstützte BS2000-Dateien und Bibliothekselemente

In bs2fs-Dateisystemen können Sie "einfache" DVS-Dateien sowie PLAM-Bibliotheken und deren Elemente bearbeiten.

3.2.1.1 BS2000-DVS-Dateien

Bei der Unterstützung von BS2000-DVS-Dateien in einem bs2fs-Dateisystem ist zu

unterscheiden, ob die Datei neu erstellt wird oder ob sie bereits im BS2000 vorhanden ist.

Neu erstellte DVS-Dateien

- FCBTYPE

  In einem bs2fs-Dateisystem neu erstellte DVS-Dateien sind immer vom FCBTYPE SAM mit variabler Satzlänge.

- BACL (siehe auch "Zugriffsattribute")

  Die BACL-Attribute werden automatisch im BS2000 aktiviert und zwar abhängig von dem beim *open()* angegeben mode und der *umask*-Einstellung (`mode & ~umask`).

  Z.B. ergeben sich bei einer *umask*-Einstellung `022` (Default in der Shell) und einem *mode*-Muster `777` die Schutzbitrechte `rwx r-x r-x`.

  Bei aktiviertem BACL werden Modifikationen mit /MODIFY-FILE-ATTRIBUTES bei USER-ACCESS oder ACCESS nicht mehr ausgewertet.

- Übertragungsmodus *ftyp*

  Ob die Übertragung einer SAM-Datei in den bs2fs-Container und zurück im Modus *text*, *textbin* oder *binary* erfolgt, hängt von der *ftyp*-Option des *mount*-Kommandos für das bs2fs-Dateisystem ab. Das POSIX-Kommando *ftyp* wird nicht ausgewertet und ist dem *bs2cp* vorbehalten. Siehe auch *mount*-Kommando, *-o* Option *ftyp*.

Bereits existierende DVS-Dateien

- FCBTYPE

  Bereits existierende DVS-Dateien können vom FCBTYPE SAM, ISAM und PAM sein. Der FCBTYPE bleibt auch beim Überschreiben einer solchen Datei erhalten.

  Unter Überschreiben ist zu verstehen: *creat()*, *open (O_CREAT |O_WRONLY |O_TRUNC)*, *cp*-Kommando, nicht jedoch *rename()* oder *mv*-Kommando.

  Für bereits existierende DVS-Dateien, die lediglich per /CREATE-FILE o.ä. eingerichtet wurden, ist noch kein FCBTYPE festgelegt (kein open). Diese Dateien erhalten nach dem Öffnen als bs2fs-Datei den FCBTYPE SAM.

- BACL (siehe auch "Zugriffsattribute")

  Bei schreibenden und überschreibenden Zugriffen bleiben alle ursprünglichen Schutzattribute erhalten. Wenn die BACL-Attribute zuvor nicht aktiviert waren, sind sie es hinterher auch nicht. Die *mode*-Angaben im *open()* oder *creat()* werden wie in POSIX üblich bei bereits existierenden Dateien ignoriert.

  Die BACL-Eigenschaften von bereits existierenden Dateien ohne BACL können im bs2fs-Dateisystem nur durch explizite Modifikation der Schutzattribute mit *chmod()* oder dem *chmod*-Kommando aktiviert werden.

- Übertragungsmodus ftyp

  ISAM-Dateien werden immer im Modus *text*, PAM-Dateien immer im Modus *binary* übertragen. Für SAM-Dateien ist analog zu neu erstellten SAM-Dateien die *ftyp*-Option des *mount*-Kommandos für das bs2fs-Dateisystem relevant.

  > **Vorsicht bei Kopieraktionen innerhalb der bs2fs-Welt**
  > Wenn die Dateiattribute der Quelle und des Ziels unterschiedlich sind bzgl. des Übertragungsmodus (text, binary), kann es zu unerwarteten (Daten-)Ergebnissen kommen.
  >
  > Eine Analogie zum /COPY-FILE-Kommando, wo die Zieldatei den gleichen FCBTYPE wie die Quelldatei erhält, gibt es beim Kopieren mit POSIX-bs2fs-Mitteln nicht.

Nicht unterstützte BS2000-Dateitypen

Folgende Dateien/Bibliotheken werden **nicht** in einem bs2fs-Dateisystem unter POSIX dargestellt, also beispielsweise vom *ls*-Kommando nicht ausgegeben:

- ausgelagerte (migrierte) Dateien/Bibliotheken (bitte beachten Sie auch die Hinweise zur Behandlung von migrierten Dateien in "Sammlung von bs2fs-Besonderheiten")
- Alias-Katalogeinträge
- Dateigenerationsgruppen und Dateigenerationen
- Dateien/Bibliotheken, auf die über RFA zugegriffen wird
- Band-Dateien
- Dateien auf Privatplatten

Durch Passwort geschützte Dateien sind im bs2fs-Dateisystem unter POSIX zwar sichtbar, es sind jedoch nur solche Zugriffe auf ihren Inhalt möglich, die nicht durch Passwort geschützt sind, z.B. lesender Zugriff auf eine mit Schreib-Passwort geschützte Datei.

Durch Guards geschützte Dateien sind im bs2fs-Dateisystem unter POSIX sichtbar. Zugriffe auf ihren Inhalt sind nur möglich, wenn das schützende Guard den entsprechenden Zugriff für die Benutzerkennung des Aufrufers zulässt.

Überblick über die Open-Modi von DVS-Dateien

Abhängig von ihrem FCBTYPE, der Satzlänge und der Option *ftyp* beim Einhängen des bs2fs-Dateisystems (siehe "mount - Dateisysteme und ferne Ressourcen einhängen") werden DVS-Dateien in POSIX unterschiedlich interpretiert. Einen Überblick über die Abbildung dieser Eigenschaften auf den Dateityp und die entsprechenden Open-Modi im bs2fs-Dateisystem gibt folgende Tabelle:

FCBTYPE	ftyp	Dateityp	Openmodus zum Lesen	Openmodus zum Schreiben
SAM	text (default)	Textdatei	r	w,tabexp=yes
	binary	Binärdatei	rb,type=record	wb,type=record,forg=seq
	textbin	Binäre Textdatei	r	w,tabexp=no
ISAM	-	Textdatei	r	w,tabexp=yes

PAM	-	Binärdatei	r	wb,type=record

Die Satzschlüssel von ISAM-Dateien werden beim Lesen ignoriert, beim Schreiben werden numerische Satzschlüssel (in Zeichendarstellung) erzeugt.

3.2.1.2 PLAM-Bibliotheken und Elemente

Unterstützte Elementtypen

Das bs2fs-Dateisystem unterstützt dieselben Elementtypen wie das *bs2cp*-Kommando (bzw. /COPY-POSIX-FILE), nämlich die vordefinierten Texttypen (*D*, *J*, *M*, *P*, *S*, *X*) und LLMs (*L*). Darüber hinaus werden in bs2fs-Dateisystemen auch vom Benutzer (z.B. mit openFT) definierte Typen sowie von den oben genannten Standardtypen abgeleitete Typen unterstützt.

Analog zu DVS-Dateien vom Typ SAM, hängt der Modus *text*, *textbin* oder *binary*, in dem Textelemente (alle Typen ungleich LLM) in den bs2fs-Container und zurück kopiert werden, von der *ftyp*-Option des *mount*-Kommandos für das bs2fs-Dateisystem ab.

Darstellung von PLAM-Bibliotheken und Bibliothekselementen

Eine PLAM-Bibliothek wird als Verzeichnis mit dem Namen der PLAM-Bibliothek dargestellt. Dieses Bibliotheksverzeichnis enthält standardmäßig die Unterverzeichnisse der Standardtypen *D, J*, *L*, *M*, *P*, *S* und *X,* in denen die entsprechenden Elementtypen abgelegt sind.

Zusätzlich enthält das Bibliotheksverzeichnis weitere Verzeichnisse mit dem Namen **abgeleiteter Elementtypen**, jedoch nur dann, wenn mindestens ein Element des entsprechenden Typs existiert und der abgeleitete Typ einen der Standardtypen D, J, L, M, P, S oder X als Basistyp hat.

Die Standardtypen C, F, H, R, U und SYSJ und davon abgeleitete Typen werden nicht unterstützt. Dies gilt auch für die reservierten Typen, deren Namen mit '$' oder 'SYS' beginnen.

Falls die PLAM-Bibliothek **benutzerdefinierte Typen** enthält, die keinen Standardtyp als Basistyp haben, werden diese als entsprechende Unterverzeichnisse dargestellt. Solche Typen können z.B. mit openFT erzeugt werden. PLAM-Bibliotheken lassen sich mit bs2fs-Mitteln (z.B. mit *mkdir*) nicht neu erstellen. PLAM-Bibliotheken, die nach einem bs2fs-*mount* mit anderen Mitteln (z.B. mit LMS) neu erstellt wurden, können jedoch jederzeit im bs2fs-Dateisystem bearbeitet werden. Hierzu muss nur ihr Name dem Wildcardmuster des bs2fs-*mount*-Kommandos entsprechen.

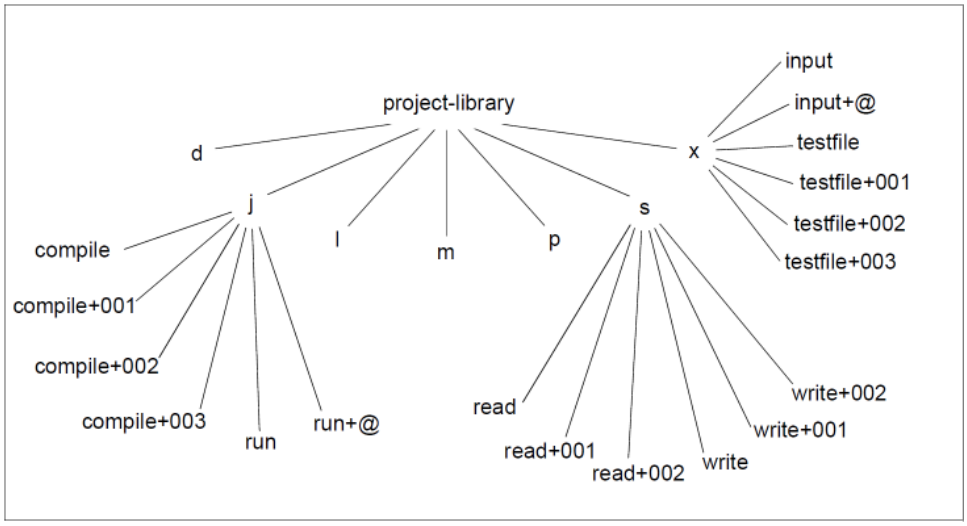
Bibliothekselemente sind als Dateien in dem Verzeichnis abgelegt, dessen Name ihrem Typ entspricht. Ihr Name ist folgendermaßen aufgebaut: *elementname* **+** *version*. Mit diesem Namen kann gezielt auf eine bestimmte Version eines Elements zugegriffen werden. Den Zugriff auf die jeweils höchste Version eines Elements ermöglicht der Name *elementname.* Hierbei handelt es sich um einen Verweis (hard link) auf die höchste Version eines Elements.

Für diesen Verweis gelten folgende Besonderheiten:

- Wird im bs2fs-Dateisystem eine neue Elementversion mit höherer Versionsbezeichnung erzeugt, verweist *elementname* automatisch auf die neue höchste Version.
- Das Löschen von *elementname* (z.B. mit `rm` *elementname*) hat folgende Wirkung:

  Zunächst wird die Datei gelöscht, die die höchste Version des Elements enthält. Falls weitere Versionen des Elements vorhanden sind, verweist *elementname* auf die neue höchste Version. Andernfalls wird auch der Verweis *elementname* gelöscht.
- Wird eine bestimmte Verson eines Elements (z.B. mit `rm` *elementname*+*ver*) gelöscht, wird auch der Verweis *elementname* gelöscht, falls keine weiteren Versionen des Elements existieren.
- Alle Versionen eines Elements können mit `rm` *elementname* `+*` gelöscht werden.

Beispiel für die Struktur einer PLAM-Bibliothek



3.2.2 bs2fs-Namenskonventionen

Beim Neuerstellen einer BS2000-DVS-Datei oder eines Bibliothekselementes mit POSIX-Mitteln (*open O_CREAT*, *creat*, *rename*) wird der Name zwar als POSIX-Dateiname angegeben, unterliegt jedoch den Regeln der Bildung von DVS-Dateinamen, Elementnamen (Namenslänge, gültige Zeichen) und weiteren bs2fs-spezifischen Regeln.

Groß-/Kleinschreibung

Die Namen können in Groß- oder Kleinschreibung oder einer Mischung aus beiden angegeben werden. Die Ausgabe der Dateinamen erfolgt jedoch immer in Kleinbuchstaben. Bei der Verwendung von wildcards in der Shell bzw. beim Kommando *find* müssen allerdings Kleinbuchstaben geschrieben werden, da sonst keine entsprechenden Dateinamen gefunden werden. Mit Großbuchstaben im Dateinamensmuster würde in der Shell die Dateinamensersetzung scheitern, da das Muster mit der Wildcardsyntax durchgereicht wird.

Die Groß-/Kleinschreibung ist für alle Objekte unterhalb des bs2fs-Einhängepunkt möglich, d.h. bei PLAM-Bibliotheken auch für die Bibliotheksverzeichnisse und Typ-Verzeichnisse.

Wildcardmuster beim mount-Kommando für das bs2fs-Dateisystem

Beim Neuanlagen einer DVS-Datei wird der Dateiname danach überprüft, ob er im Bereich des BS2000-Wildcardmusters liegt, das beim *mount*-Kommando für das bs2fs-Dateisystem angegeben wurde. Falls nicht, wird der *open* mit *errno ENAME* abgewiesen. Auch beim Suchen nach einer Datei (z.B. für lesendes *open*, *access*, *stat* u.a.) oder PLAM-Bibliothek erfolgt diese Prüfung; Nicht-Übereinstimmung wird mit *errno ENAME* zurückgewiesen.

3.2.3 Zugriffsattribute

Abbildung der BS2000-Standardattribute

Die Abbildung der BS2000-Standardattribute auf die Zugriffsrechte in POSIX können Sie folgender Tabelle entnehmen:

BS2000-Attribute		**POSIX-Standard-ACL**								
		user			**group**			**others**		
ACCESS	**USER-ACCESS**	**r**	**w**	**x**	**r**	**w**	**x**	**r**	**w**	**x**
***READ**	***OWNER-ONLY**	x	-	x	-	-	-	-	-	-
***READ**	***ALL-USERS/*SPECIAL**	x	-	x	x	-	x	x	-	x
***WRITE**	***OWNER-ONLY**	x	x	x	-	-	-	-	-	-
***WRITE**	***ALL-USERS/*SPECIAL**	x	x	x	x	x	x	x	x	x

Die mit ACCESS und USER-ACCESS festgelegten Rechte werden für eine Datei nur dann in POSIX-Zugriffsrechte abgebildet, wenn für sie keine einfache Zugriffskontrollliste (Basic Access Control List, BACL) festgelegt ist.

Folgende Attribute von Dateien und Bibliothekselementen werden nicht ins bs2fs-Dateisystem abgebildet:

- EXEC-PASSWORD, READ-PASSWORD, WRITE-PASSWORD (Datei oder Bibliothekselement)
  EXPIRATION-DATE (Datei)
  GUARD (Datei oder Bibliothekselement)

  Ist eines dieser Schutzattribute für eine Datei oder ein Element gesetzt, kann es beim Zugriffsversuch zur Abweisung mit EACCES kommen, auch wenn die gelieferte Status-Information einen Zugriff zu erlauben scheint.

- CODED-CHARACTER-SET (Datei oder Bibliothekselement)

  Ist dieses Attribut verschieden von *NONE oder *STD bzw. EDF03IRV, kann das Ergebnis einer Verarbeitung in einem bs2fs-Dateisystem ein anderes sein als das Ergebnis derselben Verarbeitung im BS2000 (etwa mit demselben C-Programm): z.B. unterschiedliche Sortierreihenfolge, verschiedene Darstellung am Bildschirm usw.

- STATE=*IN-HOLD (Bibliothekselement)

  Ist dieses Attribut gesetzt, kommt es beim Zugriffsversuch zur Abweisung mit EACCES, obwohl die gelieferte Status-Information das nicht erkennen lässt.

- ACCESS-METHOD (Bibliothekselement)

  Dieses Attribut kann bei Elementen mit SOURCE-ATTRIBUTE=*KEEP von einer entsprechenden Datei übernommen werden. Im bs2fs-Dateisystem wird es jedoch nicht berücksichtigt. Wird ein solches Element in eine Datei kopiert oder verschoben, kann eine Verarbeitung dieser Datei andere Ergebnisse haben als eine Verarbeitung des ursprünglichen Elements.

- ISAM-Schlüssel

  ISAM-Schlüssel aus BS2000-Dateien sind beim Zugriff über ein bs2fs-Dateisystem nicht sichtbar. Wird eine ISAM-Datei in einem bS2fs-Dateisystem in eine andere existierende ISAM-Datei kopiert, die dabei überschrieben wird, haben die Datensätze der beiden Dateien anschließend in der Regel unterschiedliche ISAM-Schlüssel.

- ISAM-Schlüssel (Bibliothekselement)

  ISAM-Schlüssel aus einer ISAM-Datei, die mit SOURCE-ATTRIBUTE=*KEEP in ein Bibliothekselement übernommen wurde, sind Bestandteil der jeweiligen Sätze.

- STORAGE-FORM=*DELTA (Bibliothekselement)

  Bibliothekselemente mit diesem Attribut (sogenannte Delta-Elemente) können in einem bs2fs-Dateisystem nur gelesen werden.

- Satzformat B (Bibliothekselement)

  Bibliothekselemente mit Sätzen vom Format B können nur bearbeitet werden, wenn sie vom Typ L oder einem davon abgeleiteten Typ sind. Zugriffsversuche auf Elemente anderer Typen, werden mit EMODE abgewiesen.

- Satzart ungleich 1 (Bibliothekselement)

  Datensätze mit Format A und einer Satzart ungleich 1 sind beim Lesen nicht sichtbar.

  Beim Update (O_RDWR oder O_APPEND) und beim Umbenennen innerhalb derselben Bibliothek und desselben Typs oder zwischen Typen mit demselben Basistyp bleiben sie erhalten, sofern die Quelle beim Umbenennen nicht schreibend geöffnet ist.

  Beim Kopieren oder Verschieben eines Bibliothekselements in eine Datei, eine andere Bibliothek oder beim Umbenennen in einen Typen mit anderem Basistyp oder beim Umbenennen einer schreibend geöffneten Quelle werden sie nicht mit übertragen.

BACL-Attribute

Beim Neuanlegen von bs2fs-Dateien und Bibliothekselementen sowie bei Attributänderungen mit *chmod* werden die BACL-Attribute im BS2000 aktiviert. Bei aktiviertem BACL werden Modifikationen über /MODIFY-FILE-ATTRIBUTES für USER-ACCESS oder ACCESS nicht mehr ausgewertet.

- Systeme ohne SECOS:

  An Systemen, an denen SECOS nicht aktiv ist, können die BACL-Gruppenrechte der BS2000-Datei bzw. die im bs2fs-Dateisystem abgebildeten Gruppenrechte zwar modifiziert werden, werden aber bei Datei-Zugriffen nicht ausgewertet. Es zählen alleine die Benutzerrechte und die Rechte für OTHER. Dieses Verhalten ist bei BS2000- und bs2fs-Zugriffen identisch.

- Systeme mit SECOS, Gruppe *UNIVERSAL:

  An Systemen, an denen SECOS aktiv ist, aber keine SECOS-Gruppen eingerichtet und zugewiesen wurden, sind alle Kennungen Mitglieder der Gruppe *UNIVERSAL. Das hat folgende Auswirkungen:

  - Bei Dateizugriffen im native BS2000 werden ausschließlich die Benutzerrechte und die Rechte für GROUP ausgewertet, die Rechte für OTHER nicht.
  - Bei Zugriffen im bs2fs-Dateisystem spielen die POSIX-Gruppe und die Rechte für OTHER eine Rolle:

    Bei gesetztem Schutzattribut `-rw-r----- (0640)` wird der lesende Zugriff durch eine andere Kennung abgewiesen, wenn die Kennung nicht zusätzlich derselben POSIX-Gruppe angehört. Ist dies nicht der Fall, wird für den lesenden Zugriff das Leserecht für OTHER benötigt (0644).

    Wenn die Kennung derselben POSIX-Gruppe angehört, wird nur Leserecht für GROUP benötigt.

3.2.4 Sicherheit

Mit bs2fs-Dateisystemen kann auf Dateien im BS2000 zugegriffen werden. Unberechtigte Zugriffe können Sie mit den verschiedenen Schutzmechanismen von BS2000 und POSIX abwehren. Folgende Schutzmechanismen stehen zur Verfügung:

- Benutzerverwaltung (BS2000)
- Dateizugriffsschutz (BS2000 und POSIX)

Zugriffsrechte im bs2fs-Dateisystem

Grundsätzlich hat ein Anwender im bs2fs-Dateisystem dieselben Zugriffsrechte wie im BS2000. Er kann also auf eine Datei oder ein Bibliothekselement nur die Zugriffe (Lesen, Schreiben, Ausführen) durchführen, die ihm auch im native BS2000 erlaubt wären.

In Einzelfällen können Einschränkungen beim Zugriff auftreten. Wenn eine Datei beispielsweise mit einem Passwort oder Guard geschützt ist, ist es möglich, dass einem POSIX-Anwender ein Zugriff verboten ist, der ihm als BS2000-Anwender erlaubt wäre.

Folgende Tabelle gibt einen Überblick über die für die verschiedenen Zugriffe erforderlichen Zugriffsrechte:

Zugriff	**Objekt**	**Für den Zugriff erforderliche Rechte**
Anlegen	Datei	Eigentümer (userid) des Dateisystems (falls die Datei bereits existiert, genügt das Schreibrecht für die Datei)
	Bibliothekselement	Schreib- und Leserecht für die Bibliothek **und** zusätzlich Administrationsrecht für die Bibliothek oder den Typ, falls ein solches definiert ist (falls das Bibliothekselement bereits existiert, genügt das Schreibrecht für das Bibliothekselement)
Öffnen zum Lesen	Datei	Leserecht für die Datei
	Bibliothekselement	Leserecht für die Bibliothek **und** Leserecht für das Element
Öffnen zum Schreiben	Datei	Schreib- und Leserecht für die Datei
	Bibliothekselement	Schreib- und Leserecht für die Bibliothek **und** Schreibrecht für das Element Wenn das Element nicht zum Überschreiben (O_TRUNC) geöffnet wird, ist außerdem das Leserecht für das Element erforderlich.
Löschen	Datei	Eigentümer (userid) des Dateisystems **und** Schreibrecht für die Datei
	Bibliothekselement	Schreib- und Leserecht für die Bibliothek **und** zusätzlich Administrationsrecht für die Bibliothek oder den Typ, falls ein solches definiert ist **und** Schreibrecht für das Element
Umbenennen	Datei/ Bibliothekselement	Für die Quelle sind die Rechte für Löschen und für das Ziel die Rechte für Anlegen erforderlich

Für alle anderen Zugriffe gelten die mit einfachen Zugriffskontrolllisten (Basic Access Control List, BACL) oder ACCESS und USER-ACCESS festgelegten Rechte.

> Die Kennung TSOS hat standardmäßig die gleichen Rechte wie der Eigentümer einer Datei oder einer PLAM-Bibliothek.
>
> Der Kennung SYSROOT sind beim Zugriff auf Objekte in bs2fs-Dateisystemen **keine** besonderen Rechte zugeordnet.

Anwendungsempfehlungen

Damit die Dateien und Bibliothekselemente, die in einem bs2fs-Dateisystem montiert werden, für einen POSIX-Anwender zugreifbar sind, gelten folgende Empfehlungen:

- Soll es einem Anwender erlaubt sein, eine Datei oder ein Bibliothekselement auszuführen, muss für ihn mindestens das Leserecht gesetzt sein. Um eine Datei schreibend (z.B. open mit O_WRONLY) oder ein Bibliothekselement schreibend, aber nicht überschreibend (z.B. open mit O_WRONLY ohne O_TRUNC) zu öffnen, muss er ebenfalls das Leserecht besitzen. Das Leserecht darf ihm auch nicht entzogen werden, wenn das Objekt mit einem Guard geschützt wird.
- Soll einem Anwender das Lesen, Schreiben oder Ausführen einer Datei oder eines Bibliothekselementes erlaubt sein, darf der entsprechende Zugriff nicht durch ein Passwort geschützt sein.
- Damit ein schreibender Zugriff auf eine Datei oder ein Bibliothekselement oder ein Löschen möglich ist, darf dieses Objekt nicht mit einem Verfallsdatum (EXPIRATION-DATE) geschützt sein, das größer als das aktuelle Datum ist.
- Für jeden Zugriff auf Bibliothekselemente, auf deren Eigenschaften oder auch auf das Inhaltsverzeichnis einer Bibliothek benötigt ein Anwender mindestens das Leserecht für die Bibliothek.

3.2.5 Sammlung von bs2fs-Besonderheiten

Beim Arbeiten mit Dateien in einem bs2fs-Dateisystem sind einige Besonderheiten (z.B. gegenüber dem Arbeiten mit lokalen POSIX-Dateien) zu beachten, die im Folgenden noch einmal zusammengefasst werden:

- Neu erstellte Dateien in einem bs2fs-Dateisystem

  Dateien oder Elemente, die in einem bs2fs-Dateisystem neu erstellt werden, erhalten BACL-Attribute. Somit haben die Dateiattribute ACCESS und USER-ACCESS keine Bedeutung für den Schutz.

- DVS-Wildcard beim *mount*-Kommando für das bs2fs-Dateisystem

  Neue DVS-Dateien (immer vom Typ SAM) können nur angelegt werden, wenn ihr Name der Musterangabe im *mount*-Kommando und den Syntaxregeln des BS2000 entspricht. Insbesondere ist es nicht möglich, Dateien anzulegen, deren Name mit einem Punkt (.) beginnt.

- PLAM-Bibliothek: BS2000-Datei-Lock vom Typ Lesen

  Beim Zugriff auf eine PLAM-Bibliothek im bs2fs-Dateisystem (z.B. mit *ls*), wird diese mit einem lesenden Lock ("Lock Type INPUT") im BS2000 versehen. Umbenennen, Löschen und Ändern der Dateiattribute der PLAM-Bibliothek sind dann nicht möglich. Lese-Locks werden vom bs2fs-Dämonen automatisch wieder freigegeben, wenn innerhalb von ca. 20 Sekunden keine bs2fs-Zugriffe auf die PLAM-Bibliothek mehr erfolgt sind.

  Locks vom Typ Schreiben ("Lock Type INOUT") auf eine PLAM-Bibliothek gibt der bs2fsd-Dämon dagegen sofort frei, sobald in der Bibliothek kein Element mehr für den auftraggebenden Benutzer geöffnet ist.

- Löschen von Dateien in einem bs2fs-Dateisystem erfordert Schreibrecht

  Das Löschen von Dateien in einen bs2fs-Dateisystem wird generell zurückgewiesen, wenn kein Schreibrecht existiert. In POSIX können auch Dateien ohne Schreibrecht gelöscht werden (vgl. *rm* mit Rückfrage bzw. mit Schalter *-f* oder *mv* mit Rückfrage).

- Zugriffsrechte für den privilegierten Benutzer

  Analog zu den Zugriffsrechten im BS2000 besitzt auch beim Zugriff auf Dateien im bs2fs-Dateisystem nur die Kennung TSOS standardmäßig die gleichen Rechte wie der Eigentümer einer Datei oder einer PLAM-Bibliothek. Die Kennung SYSROOT, der im POSIX ebenfalls die privilegierte uid=0 zugeordnet ist, wird bei bs2fs-Dateizugriffen wie eine beliebige andere, nicht privilegierte Kennung behandelt.

  In der POSIX/UNIX-Welt werden privilegierten Kennungen (uid=0) Zugriffe auf Dateien gewährt, die über die Zugriffsmöglichkeiten des Dateieigentümers hinausgehen, z.B. das Lesen einer Datei, obwohl die Schutzbits dem Dateieigentümer kein Lesen erlauben.

  In bs2fs-Dateisystemen stehen jedoch auch TSOS nur die Berechtigungen zur Verfügung, die der Eigentümer eines Objekts im BS2000 hat. Beim Einsatz von SECOS (z.B. GUARDS) kann der Eigentümer auch TSOS-Rechte einschränken.

- Update von Dateien im Textmodus

  Beim Update von bestehenden Sätzen in mit *ftyp=text* eingehängten SAM- oder ISAM-Dateien darf die Satzlänge nicht geändert werden. Die Verkürzung oder Verlängerung eines Satzes führt zu einem *EIO*-Error. Der *write()* wird nicht ausgeführt, und nach dem *close()* sind nur die Teile der BS2000-Datei verändert, die erfolgreich mit *write()* geschrieben wurden.

- Zeitstempel

  Im bs2fs-Dateisystem werden die mit FSTAT ermittelten Zeitstempel einer Datei bzw. eines Elements wie folgt auf die Felder der *stat*-Datenstruktur im POSIX abgebildet:

BS2000	**POSIX**
CRE-DATE/TIME	*st_ctime*
ACC-DATE/TIME	*st_atime*
CHANG-DATE/TIME	*st_mtime*

  Zu beachten ist hierbei:

  - Während eine Datei geöffnet ist, werden im bs2fs-Dateisystem die Felder *st_atime* und *st_mtime* der ufs-Schattendatei als *st_atime* und *st_mtime* der bs2fs-Datei übernommen, nach dem Schließen werden wieder die Werte ausgegeben, die FSTAT im BS2000 liefert. Dies kann dazu führen, dass die mit dem Kommando /SHOW-FILE-ATTRIBUTES angezeigten Änderungszeiten einer Datei, von denen mit *ls -l* im bs2fs-Dateisystem ermittelten Zeiten abweichen, solange die Datei geöffnet ist.
  - Die Felder CRE-DATE/TIME und *st_ctime* werden im BS2000 und POSIX/UNIX unterschiedlich interpretiert. CRE-DATE/TIME gibt im BS2000 den Zeitpunkt des Anlegens einer Datei wieder, *st_ctime* protokolliert in POSIX/UNIX den Zeitpunkt der
    letzten Änderung der Verwaltungsstrukturen wieder (z.B. aufgrund von *chmod*)

- Dateigrößen

  Solange eine Datei oder ein Bibliothekselement im bs2fs nach dem Einhängen (*mount*) nicht geöffnet war, wird bei der Ausgabe des Kommandos *ls -l* die Größe einer Datei aus der vom BS2000-FSTAT-Makro gelieferten Anzahl von PAM-Seiten mal 2048 berechnet. Erst nachdem eine Datei im bs2fs-Dateisystem geöffnet wurde, erfolgt die Anzeige byte-genau. Dieser Wert kann fehlerhaft sein, falls die Größe der Datei außerhalb des bs2fs verändert wurde.

- Behandlung von migrierten Dateien

  Die Verarbeitung von migrierten Dateien wird nicht unterstützt. Dies hat folgende Auswirkungen:
  - *ls*, *stat()*, *fstat()* zeigen die Namen migrierter Dateien nicht an.
  - Beim Versuch, eine migrierte Datei zu öffnen, wird errno 8 (`ENOENT`, "No such file or directory") zurückgeliefert.
  - Beim Versuch, eine neue Datei mit dem Namen einer migrierten Datei zu erzeugen, z.B. mit *creat()*, *rename()* oder mit den Shell-Kommmandos *cp*, *mv*, wird errno 206 (`ENXIO`, "No such device or adress") zurückgeliefert. `ENXIO` ist bei bs2fs-Zugriffen eindeutig, d.h. es handelt sich immer um den Versuch, eine Datei mit dem Namen einer migrierten Datei neu zu erzeugen.

- X-Bit für Dateiverzeichnisse

  Das x-Bit für ein Dateiverzeichnis wird im bs2fs-Dateisystem anders interpretiert als in der UNIX-Welt üblich. Zum Suchen eines Eintrags im Inhaltsverzeichnis einer PLAM-Bibliothek benötigt man im bs2fs-Dateisystem das Leserecht, nicht das Ausführungsrecht.

- Die Verzeichnisstruktur eines bs2fs-Dateisystems ist durch den *mount*-Vorgang festlegt und kann nicht verändert werden, da sie die Struktur von PLAM-Bibliotheken abbildet. Verzeichnisse in einem bs2fs-Dateisystem können daher nicht gelöscht oder umbenannt werden, und es ist auch nicht möglich, neue Verzeichnisse anzulegen. Somit lassen sich auch keine neuen PLAM-Bibliotheken mit bs2fs-Mitteln (z.B. mit *mkdir*) erstellen. Die Zugriffsrechte von Verzeichnissen können ebenfalls nicht geändert werden.

- Bibliothekselemente können mit beliebigen Namen entsprechend den Syntaxregeln für PLAM-Bibliotheken in den definierten Verzeichnissen (Typen) angelegt werden, außer in Verzeichnissen, die einen Typ abbilden, der keinen Standardtyp als Basistyp hat.

- Bibliothekselemente, die einem Typ zugeordnet sind, der keinen Standardtyp als Basistyp hat, können nur gelesen werden.

- Bibliothekselemente eines Typs mit Basistyp L können nur in Bibliothekselemente desselben Typs oder eines anderen Typs mit Basistyp L umbenannt oder kopiert werden. Ebenso können Dateien oder Bibliothekselemente eines Typs mit Basistyp ungleich L nicht in ein Element eines Typs mit Basistyp L umbenannt oder kopiert werden.

- Bibliothekselemente sind im bs2fs-Dateisystem unter einem Namen abgelegt, der sich aus dem Elementnamen und der Versionsbezeichnung zusammensetzt:
  elementname+ver
  Zusätzlich existiert ein Verweis *elementname* auf die aktuell höchste Version des Elements. Dies hat folgende Auswirkungen
  - Wird im bs2fs-Dateisystem eine neue Elementversion mit einer Versionsbezeichnung erzeugt, die höher ist als die bisher höchste, verweist *elementname* automatisch auf die neue höchste Version.
  - Das Löschen von *elementname* (z.B. mit `rm` *elementname*) hat folgende Wirkung:Zunächst wird die Datei gelöscht, die die höchste Version des Elements enthält.
    Falls weitere Versionen des Elements vorhanden sind, verweist *elementname* anschließend auf die neue höchste Version. Andernfalls wird auch der Verweis *elementname* gelöscht.
  - Wird eine bestimmte Version eines Elements (z.B. mit `rm` *elementname+ver*) gelöscht, wird auch der Verweis *elementname* gelöscht, falls keine weiteren Versionen des Elements existieren.

4 Schnittstellen zur Unterstützung von bs2fs-Dateisystemen

In diesem Kapitel finden Sie die Beschreibungen der Schnittstellen zur Verwaltung von bs2fs-Dateisystemen und Hinweise zu Shell-Kommandos und POSIX-Tools sowie zu C-Programmschnittstellen.

4.1 Administrationsschnittstellen

POSIX unterstützt die Verwaltung von bs2fs-Dateisystemen mit:

- Kommandos
- Dämonen
- Verwaltungsdateien

Zur Verwaltung von bs2fs-Dateisystemen benötigen Sie die POSIX-Root-Berechtigung (BS2000-Kennung TSOS oder SYSROOT).

4.1.1 Kommandos

Dieser Abschnitt beschreibt die für den Einsatz von bs2fs-Dateisystemen relevanten Kommandos in alphabetischer Reihenfolge. Die Darstellungsmittel, die in der Kommandosyntax verwendet werden, finden Sie in "Darstellungsmittel".

Kommando	Funktion
mount	Dateisysteme einhängen
mountall	Mehrere Dateisysteme einhängen
show_pubset_export	vom EXPORT-PUBSET betroffene Dateisysteme anzeigen
start_bs2fsd	Kopierdämönen starten
umount	Dateisysteme aushängen
umountall	Mehrere Dateisysteme aushängen

Tabelle 1: bs2fs-Kommandos

4.1.2 mount - Dateisysteme und ferne Ressourcen einhängen

(mount a filesystem)

mount (Format 2 und Format 3) hängt ein ufs-Dateisystem in die Dateisystemhierarchie an der Pfadnamenposition *einhängepunkt* ein, die bereits vorhanden sein muss. Hat *einhängepunkt* vor der mount-Operation bereits einen Inhalt, bleibt dieser verdeckt, bis das Dateisystem wieder ausgehängt wird.

mount (Format 4 und Format 5) hängt ein bs2fs-Dateisystem an einer bestimmten Stelle im POSIX-Dateisystem ein. Unter einem bs2fs-Dateisystem versteht man eine auswählbare Menge von Dateien im BS2000, die transparent in POSIX zur Verfügung gestellt werden, so dass auf sie mit POSIX-Mitteln (Kommandos, Programmschnittstellen) zugegriffen werden kann. Die Auswahl der Dateien erfolgt über Benutzer- und Katalogkennung sowie Wildcard-Symbole.

Außerdem kann mit *mount* (Format 1) eine Liste aller eingehängten Dateisysteme ausgegeben werden.

Syntax

```
Format 1:  mount[ -v | -p]

Format 2:  mount[ -F ufs][ -V][ -r]
               [ -o spez_optionen] { ressource | einhängepunkt}

Format 3:  mount[ -F ufs][ -V][ -r]
               [ -o spez_optionen] ressource einhängepunkt

Format 4:  mount[ -F bs2fs][ -V][ -r]
               [ -o spez_optionen] { ressource | einhängepunkt }

Format 5:  mount[ -F bs2fs][ -V][ -r]
               [ -o spez_optionen] ressource einhängepunkt
```

Liste der eingehängten Dateisysteme ausgeben

```
Format 1:  mount[ -v | -p]
```

Keine Option angegeben

mount gibt eine Liste aller eingehängten Dateisysteme aus (siehe Beispiel).

-v

gibt die Ausgaben in einer neuen Darstellung aus. Bei der neuen Ausgabe werden dateisystemtyp und Optionen zusätzlich zu den Angaben der alten Ausgabe angezeigt.

Die Felder *einhängepunkt* und *ressource* sind vertauscht (siehe Beispiel).

-p

Gibt die Liste der eingehängten Dateisysteme im Format von */etc/vfstab* aus.

ufs-Dateisysteme einhängen

```
Format 2:  mount[ -F ufs][ -V][ -r]
               [ -o spez_optionen] { ressource | einhängepunkt}

Format 3:  mount[ -F ufs][ -V][ -r]
               [ -o spez_optionen] ressource einhängepunkt
```

Format 2 und Format 3 werden gemeinsam beschrieben, da sie sich nur durch die (optionale) Angabe von *ressource* und *einhängepunkt* unterscheiden.

Format 2 kann nur verwendet werden, wenn in der Datei */etc/vfstab* bereits ein Eintrag für das entsprechende Dateisystem existiert. Aus diesem wird dann die fehlende Angabe für *ressource* oder *einhängepunkt* ergänzt.

Formate 2 und 3 dürfen nur vom POSIX-Verwalter eingegeben werden.

Keine Option angegeben

mount gibt eine Liste aller eingehängten Dateisysteme aus.

-F ufs

Gibt *ufs* als Dateisystem-Typ an.

-V

Gibt die gesamte Kommandozeile auf dem Bildschirm aus, führt das Kommando jedoch nicht aus. Die Kommandozeile wird mit den vom Benutzer eingegebenen Optionen und Argumenten sowie aus */etc/vfstab* abgeleiteten Werte erstellt. Diese Option sollte verwendet werden, um die Kommandozeile einer allgemeinen Prüfung und einer Gültigkeitsprüfung zu unterziehen.

-r

Einhängen des Dateisystems mit Lesezugriff.

-o

Angeben *ufs*-dateisystemspezifischer Optionen. Werden mehrere Optionen angegeben, werden sie durch Komma getrennt. Werden ungültige Optionen angegeben, wird eine Warnung ausgegeben, und die ungültigen Optionen werden ignoriert.

Folgende Optionen sind verfügbar:

f

Täuscht einen */etc/mntab*-Eintrag vor, hängt aber keine Dateisysteme ein. Parameter werden nicht überprüft.

n

Einhängen des Dateisystems, ohne einen Eintrag in */etc/mnttab* zu stellen.

journal

Die Funktion "Journaling" wird nicht mehr unterstützt, d.h. die Option ist wirkungslos.

rw | ro

Lese-/Schreib- oder Lesezugriff. Standardwert ist *rw*.

nosuid

Standardmäßig wird das Dateisystem so eingehängt, dass das s-Bit für Benutzer gesetzt wird. Durch Angeben von *nosuid* wird der Standardwert außer Kraft gesetzt, und das Dateisystem wird ohne Setzen des s-Bits für Benutzer eingehängt.

remount

Wird zusammen mit *rw* verwendet. Ein mit Lesezugriff eingehängtes Dateisystem kann mit Lese-/Schreibzugriff neu eingehängt werden. Diese Option schlägt fehl, wenn das Dateisystem aktuell nicht oder mit *rw* eingehängt ist.

bs2fscontainer

Vereinbart das einzuhängende Dateisystem als bs2fs-Container, d.h. als Dateisystem, das temporär Dateien von bs2fs-Dateisystemen aufnimmt.

Diese Option darf nur für ein einziges *ufs*-Dateisystem angegeben werden. Jedes weitere *mount*-Kommando mit dieser Option wird abgewiesen.

Die Optionen *-r*, *-o ro*, *-o journal* und *-o* remount dürfen nicht gemeinsam mit der Option *bs2fscontainer* angegeben werden.

Beim Einsatz des POSIX-Installationsprogramm kann diese Option über die Optionszeile eingegeben werden.

Diese Option kann nur für ein Dateisystem angegeben werden, das beim Einrichten mit dem POSIX-Installationsprogramm als bs2fs-Container gekennzeichnet wurde. Bei der *append*-Funktion eines neu zu erstellenden oder zu überschreibenden ufs-Dateisystems muss dazu in der Optionszeile die Option *bs2fscontainer* angegeben werden. Ist dies nicht der Fall, so wird das Einhängen mit Fehler abgebrochen und das Dateisystem bleibt samt Inhalt erhalten.

Das ufs-Dateisystem, das als bs2fs-Container dient, wird als leeres Dateisystem erwartet. Falls es nicht leer ist, wird sein Inhalt bei Ausführung des *mount*-Kommandos mit der Option *–o bs2fscontainer* gelöscht.

Nach einem erfolgreichen mount werden zwei bs2fsd-Kopierdämonen automatisch gestartet.

`ressource`

Gibt das Dateisystem an, das eingehängt werden soll.

`einhängepunkt`

Gibt an, wo die Ressource lokal eingehängt werden soll. Es muss ein absoluter Pfadname angegeben werden. Handelt es sich bei *einhängepunkt* um einen symbolischen Verweis, wird das Dateisystem in das Verzeichnis, auf das sich der symbolische Verweis bezieht, eingehängt und nicht zusätzlich zu dem symbolischen Verweis.

bs2fs-Dateisysteme einhängen

```
Format 4:  mount[ -F bs2fs][ -V][ -r]
               [ -o spez_optionen] { ressource | einhängepunkt }

Format 5:  mount[ -F bs2fs][ -V][ -r]
               [ -o spez_optionen] ressource einhängepunkt
```

Format 4 und Format 5 werden gemeinsam beschrieben, da sie sich nur durch die (optionale) Angabe von *ressource* und *einhängepunkt* unterscheiden.

Format 4 kann nur verwendet werden, wenn in der Datei */etc/vfstab* bereits ein Eintrag für das entsprechende Dateisystem existiert. Aus diesem wird dann die fehlende Angabe für *ressource* oder *einhängepunkt* ergänzt (siehe Hinweis).

Formate 4 und 5 dürfen nur vom POSIX-Verwalter eingegeben werden.

Voraussetzung für die Eingabe eines *mount*-Kommandos des Formats 4 oder 5 ist, dass bereits ein bs2fs-Container eingehängt ist.

Keine Option angegeben

mount gibt eine Liste aller eingehängten Dateisysteme aus.

`-F bs2fs`

Gibt *bs2fs* als Dateisystem-Typ an.

`-V`

Gibt die gesamte Kommandozeile auf dem Bildschirm aus, führt das Kommando jedoch nicht aus. Die Kommandozeile wird mit den vom Benutzer eingegebenen Optionen und Argumenten sowie aus */etc/vfstab* abgeleiteten Werte erstellt. Diese Option sollte verwendet werden, um die Kommandozeile einer allgemeinen Prüfung und einer Gültigkeitsprüfung zu unterziehen.

`-r`

Einhängen des Dateisystems mit Lesezugriff.

`-o`

Angeben *bs2fs*-dateisystemspezifischer Optionen. Werden mehrere Optionen angegeben, werden sie durch Komma getrennt. Werden ungültige Optionen angegeben, wird eine Warnung ausgegeben, und die ungültigen Optionen werden ignoriert.

Folgende Optionen sind verfügbar:

`rw | ro`

Lese-/Schreib- oder Lesezugriff. Standardwert ist *rw*.

nosuid

Das Dateisystem wird ohne Setzen des s-Bits für Benutzer eingehängt. Für bs2fs-Dateisysteme ist diese Option standardmäßig aktiviert und kann nicht deaktiviert werden.

remount

Wird zusammen mit *rw* verwendet. Ein mit Lesezugriff eingehängtes Dateisystem kann mit Lese-/Schreibzugriff neu eingehängt werden. Diese Option schlägt fehl, wenn das Dateisystem aktuell nicht oder mit *rw* eingehängt ist.

ftyp={**text**|**binary**|**textbin**}

Die Wirkung dieser Option entspricht der des Kommandos *ftyp* beim Kopieren von Dateien mit dem *bs2cp*-Kommando. Die Option legt fest, ob BS2000-SAM-Dateien und textartige PLAM-Bibliothekselemente (Elementtyp ungleich L) in POSIX als Text- oder Binärdateien interpretiert werden. PAM-Dateien werden stets als Binärdateien, ISAM-Dateien werden stets als Textdateien interpretiert.

Diese Option sollte nur einmal angegeben werden. Bei mehrfacher Angabe gilt die Angabe mit der höchsten Priorität, wobei *ftyp=textbin* die höchste, *ftyp=text* die nächsthöchste und *ftyp=binary* die niedrigste Priorität besitzt.

Default ist *ftyp=text*.

ftyp=text

SAM-Dateien und textartige Bibliothekselemente werden als Textdateien interpretiert. Beim Schreiben in eine bs2fs-Datei werden Zeilenende-Zeichen (X'15') in einen Satzwechsel umgewandelt und Tabulatorzeichen (X'05') in die entsprechende Anzahl Leerzeichen.

ftyp=binary

SAM-Dateien und textartige Bibliothekselemente werden als Binärdateien interpretiert. Es erfolgt eine 1:1-Übertragung ohne Interpretation und Umsetzung von Daten (Satzwechsel/Zeilenende-Zeichen, Tabulator/Leerzeichen, usw.).

ftyp=textbin

SAM-Dateien und textartige Bibliothekselemente werden als binäre Textdateien interpretiert. Beim Schreiben in eine bs2fs-Datei werden nur Zeilenende-Zeichen (X'15') in einen Satzwechsel umgewandelt. Tabulatoren (X'05') werden nicht in Leerzeichen umgesetzt.

`ressource`

Legt fest, welche BS2000-Dateien gemounted werden sollen. Die Option ist in folgender Syntax anzugeben:

```
:cat:$user.filename-with-wild
```

Sie kann in Groß- oder Kleinschreibung oder auch in gemischter Form angegeben werden. Sonderzeichen der POSIX-Shell wie '$' oder '*' müssen explizit entwertet werden.

`cat`

Katalog-Kennung

`user`

BS2000-Benutzerkennung

`filename-with-wild`

BS2000-Dateiname mit Wildcard-Symbolen:

`*`

Ersetzt eine beliebige, auch leere Zeichenfolge.

`/`

Ersetzt genau ein beliebiges Zeichen.

`.` (Punkt) am Ende

Teilqualifizierte Angabe eines Namens.
Entspricht implizit der Zeichenfolge "./*", d.h. nach dem Punkt folgt mindestens ein beliebiges Zeichen.

`<`s_x`:`s_y`>`

Ersetzt eine Zeichenfolge, für die gilt:

- sie ist mindestens so lang wie die kürzeste Zeichenfolge (s_x oder s_y)
- sie ist höchstens so lang wie die längste Zeichenfolge (s_x oder s_y)
- sie liegt in der alphabetischen Sortierung zwischen s_x und s_y; Zahlen werden hinter Buchstaben sortiert (A...Z, 0...9)
- s_x darf auch die leere Zeichenfolge sein, die in der alphabetischen Sortierung an erster Stelle steht
- s_y darf auch die leere Zeichenfolge sein, die an dieser Stelle für die Zeichenfolge mit der höchst möglichen Codierung steht (enthält nur die Zeichen X'FF')

`<`s_1`,...>`

Ersetzt alle Zeichenfolgen, auf die eine der mit s angegebenen Zeichenkombinationen zutrifft. s kann auch die leere Zeichenfolge sein.

Jede Zeichenfolge s kann auch eine Bereichsangabe "s_x`:`s_y " sein.

`-s`

Ersetzt alle Zeichenfolgen, die der angegebenen Zeichenfolge s nicht entsprechen. Das Minuszeichen darf nur am Beginn der Zeichenfolge stehen.

Bei der mit *ressource* beschriebenen Dateimenge kann es sich sowohl um bereits existierende als auch um neu zu erstellende Dateien handeln. Beim Neuerstellen einer Datei muss der gewünschte Dateiname dem Wildcardmuster des entsprechenden *mount*-Kommandos entsprechen.

`einhängepunkt`

Gibt an, wo die Ressource lokal eingehängt werden soll. Es muss ein absoluter Pfadname angegeben werden. Handelt es sich bei *einhängepunkt* um einen symbolischen Verweis, wird das Dateisystem in das Verzeichnis, auf das sich der symbolische Verweis bezieht, eingehängt und nicht zusätzlich zu dem symbolischen Verweis.

Hinweis

Wenn für das betreffende Dateisystem ein Eintrag in der Datei */etc/vfstab* existiert, kann eine der Optionen *ressource* oder *einhängepunkt* entfallen (Format 2 bzw. 4). Beim Einsatz von bs2fs-Dateisystem ist in diesem Fall Folgendes zu beachten:

- Ist die Option *einhängepunkt* angegeben, dann kann eindeutig ein Eintrag in */etc/vfstab* identifiziert werden, und das entsprechende Dateisystem wird eingehängt.
- Ist nur die Option *ressource* angegeben so können für ein bs2fs-Dateisystem mehrere passende Einträge in der Datei */etc/vfstab* enthalten sein, da dieses Dateisystem parallel an mehreren Stellen eingehängt sein kann. In diesem Falle wird nur das **erste** in der Datei */etc/vfstab* eingetragene bs2fs-Dateisystem eingehängt. Als passende Einträge werden nur Einträge mit identischer Wildcard-Zeichenfolge erkannt. Einträge mit unterschiedlicher Wildcard-Zeichenfolge werden auch dann nicht berücksichtigt, wenn sie dieselbe Menge von Dateien definieren.

Datei

/etc/mnttab

Tabelle der eingehängten Dateisysteme

/etc/dfs/fstypes

standardmäßiger verteilter Dateisystem-Typ

/etc/vfstab

Tabelle der automatisch eingehängten Dateisysteme

/etc/mnttab

Tabelle der eingehängten Dateisysteme

Beispiel

Einhängen des bs2fs-Containers und eines bs2fs-Dateisystems. Das Beispiel wird unter der POSIX-Verwalterkennung durchgeführt.

```
# mount –F ufs –o bs2fscontainer /bs2fscont
# mount -F bs2fs ':V70a:$sysaudit.sys.conslog.2007-06*'  /home/bs2.conslog
# mount
/ on /dev/root read/write/setuid on Tue Nov 27 11:31:04 2007
...
/bs2fscont on /dev/dsk/23 bs2fscontainer/setuid/read/write/noquota on Tue Nov 27
11:35:23 2007
/home/bs2.conslog on :V70A:$SYSAUDIT.SYS.CONSLOG.2007-06* ftyp=text/nosuid on Tue
Nov 27 13:52:23 2007
#
```

4.1.3 mountall - Mehrere Dateisysteme einhängen

(mount file systems)

mountall dient zum Einhängen von Dateisystemen entsprechend einer *dateisystemtabelle* (*/etc/vfstab* ist die Standarddateisystemtabelle). Der Gerätedateiname "-" liest aus der Standardeingabe. Ist der Bindestrich angegeben, muss die Standardeingabe dasselbe Format haben wie */etc/vfstab*.

Bevor die einzelnen Dateisysteme eingehängt werden, wird mit *fsck* eine Plausibilitätsprüfung durchgeführt, um festzustellen, ob das System einhängbar erscheint (nicht bei Dateisystemen vom Typ *bs2fs* oder *nfs)*. Ist das Dateisystem nicht einhängbar, wird es mit *fsck* korrigiert, bevor versucht wird, es einzuhängen.

Ist **nur** der *dateisystemtyp* angegeben, beschränkt sich die Wirkung von *mountall* auf Dateisysteme des angegebenen Typs.

Die Dateisysteme werden in der Reihenfolge *ufs* - *bs2fs* - *nfs* eingehängt. Somit ist gewährleistet, dass beim Einhängen der bs2fs-Dateisysteme das dafür erforderliche bs2fscontainer-Dateisystem im ufs bereits eingehängt ist.

Syntax

mountall[**-F** dateisystemtyp] [**-l** | **-r** | **-b**][dateisystemtabelle]

Keine Option angegeben

mountall hängt alle Dateisysteme ein, bei denen das Feld *automnt* in der *dateisystemtabelle* auf *yes* gesetzt ist.

Optionen

-F dateisystemtyp

Angabe des Dateisystem-Typs, der eingehängt werden soll.

-l

Begrenzt den Vorgang auf lokale Dateisysteme (*ufs* und *bs2fs*).

-r

Begrenzt den Vorgang auf ferne Dateisystem-Typen (*nfs*).

-b

Begrenzt den Vorgang auf bs2fs-Dateisysteme.

dateisystemtabelle

wird *dateisystemtabelle* nicht angegeben, bezieht sich *mountall* auf */etc/vfstab*.

Hinweis

Wenn die Option *-F* in Kombination mit einer oder mehrerer der Optionen *-l*, *-r* und *-b* angegeben wird und die Optionen miteinander verträglich sind, so haben die Optionen *-l*, *-r* und *-b* Vorrang. Beispielsweise haben *mountall -F bs2fs -l* und *mountall -F ufs -l* dieselbe Wirkung wie *mountall -l* : Es werden alle lokalen Dateisysteme (also alle ufs- und bs2fs-Dateisysteme) eingehängt. Ebenso führen die Angaben *mountall -F bs2fs* und *mountall -b* zum gleichen Ergebnis: alle bs2fs-Dateisysteme werden eingehängt.

Fehler

Wenn die Dateisysteme einhängbar und fehlerfrei sind, wird keine Meldung ausgegeben. Fehler- und Warnmeldungen stammen von *fsck* und *mount* bzw. von *mountall* bei falscher Syntaxangabe.

Datei

/etc/vfstab

Standarddateisystemtabelle

4.1.4 show_pubset_export - Vom Export eines Pubsets betroffene Dateisysteme anzeigen

(show file system affected by pubset export)

Das Kommando liefert dem Systemadministrator die Information, welche Dateisysteme in POSIX vom Export eines Pubsets betroffen sind und somit vor der Ausführung des Kommando EXPORT-PUBSET ausgehängt werden müssen.

Diese Information ist insbesondere beim Einsatz von bs2fs-Dateisystemen hilfreich. Dabei genügt es nämlich nicht, wie bei ufs- und NFS-Dateisystemen, zu prüfen, ob der Einhängepunkt eines Dateisystems auf dem betroffenen Pubset liegt.

Bei bs2fs-Dateisystemen ist zusätzlich auch die Lage der mittels bs2fs eingehängten BS2000-Dateien relevant. Außerdem spielt der Einhängepunkt des bs2fs-Containers eine Rolle. Liegt dieser auf dem zu exportierenden Pubset, so sind alle eingehängten bs2fs-Dateisysteme vom Export betroffen, unabhängig von ihrer Lage.

Abhängig vom Dateisystemtyp ist ein Dateisystem dann vom EXPORT-PUBSET betroffen, wenn die in der folgenden Liste aufgeführten Objekte auf dem zu exportierenden Pubset liegen:

ufs:

Einhängepunkt oder Behälterdatei

nfs:

Einhängepunkt

bs2fs:

Einhängepunkt oder eingehängte BS2000-Dateien oder Einhängepunkt bzw. Behälterdatei des bs2fs-Containers (vgl. ufs)

Syntax

```
show_pubset_export cat-id
```

`cat-id`

Katalog-Kennung des Pubsets, das geprüft werden soll (ohne einschließende Doppelpunkte ":"). Die Angabe kann in Groß- oder Kleinschreibung oder gemischt erfolgen, die Prüfung wird mit der in Großbuchstaben umgewandelten *cat-id* durchgeführt.

Datei

Die folgenden Dateien werden zur Ermittlung der betroffenen Dateisysteme nach der angegebenen Katalog-Kennung durchsucht:

/etc/mnttab

Tabelle aller eingehängten Dateisysteme

/etc/partitions

Tabelle aller möglichen Partitions

Falls in dieser Datei die Angabe der Katalogkennung fehlt, so wird die Default-Id über das BS2000 ermittelt und für die Prüfung verwendet.

SYSSSI.POSIX-BC.<version>

SYSSSI-Datei von POSIX aus dem BS2000
Aus dieser Datei wird der BS2000-Dateiname der Behälterdatei des root-Dateisystems ermittelt (Parameter ROOTFSNAME), da dieser Name nicht in der Datei */etc/partitions* eingetragen ist.

Beispiel

Die vom Export des Pubsets DATA betroffenen Dateisysteme werden ermittelt. Die Behälterdatei des unter dem Home-Verzeichnis */home/bach* eingehängten ufs-Dateisystems liegt auf dem Pubset DATA.

```
# show_export DaTa
the nfs filesystems on pubset DATA
nfs filesystem PGTR0157:/home4 mounted on /home/bach/nfs4
nfs filesystem PGTR0157:/home5 mounted on /home/bach/nfs5
the bs2fs filesystems on pubset DATA
bs2fs filesystem :DATA:$BACH.ASS.* mounted on /home/bach/bs2/mount.ass
bs2fs filesystem :V70A:$BACH.CCC.* mounted on /home/bach/bs2/mount.ccc
bs2fs filesystem :DATA:$BACH.PLAM* mounted on /home/bach/bs2/mount.plam
the ufs filesystems on pubset DATA
ufs filesystem /dev/dsk/4 mounted on /home/froede
ufs filesystem /dev/dsk/5 mounted on /home/bach
#
```

4.1.5 start_bs2fsd - Kopierdämonen starten

(start copy daemons)

Mit *start_bs2fsd* kann der Systemadministrator zusätzliche Kopierdämonen *bs2fsd* starten.

Syntax

```
start_bs2fsd [number]
```

Keine Option angegeben

start_bs2fsd informiert über die Anzahl der aktuell laufenden Kopierdämonen.

`number`

Gibt an wie viele Kopierdämonen zusätzlich zu den bereits laufenden Kopierdämonen gestartet werden sollen. Es können maximal 8 Kopierdämonen laufen.

Beispiel

Prüfung, wie viele Dämonen aktuell laufen

```
# start_bs2fsd
/sbin/start_bs2fsd: 2 bs2fs daemons are running
```

Start eines zusätzlichen Dämonen und erneute Prüfung

```
# start_bs2fsd 1
/sbin/start_bs2fsd: start additional daemon 1 of 1
# start_bs2fsd
/sbin/start_bs2fsd: 3 bs2fs daemons are running
```

4.1.6 umount - Dateisysteme und ferne Ressourcen aushängen

(unmount a filesystem)

Das Kommando *umount* hängt ein über *mount* eingehängtes Dateisystem wieder aus. Der Eintrag des Dateisystems wird aus der Tabelle */etc/mnttab* gelöscht.

Syntax

```
umount[ -V] {ressource | einhängepunkt}
```

-V

Gibt die gesamte Kommandozeile auf dem Bildschirm aus, führt das Kommando jedoch nicht aus. Die Kommandozeile wird mit den vom Benutzer eingegebenen Optionen und Argumenten sowie aus */etc/vfstab* abgeleiteten Werte erstellt. Diese Option sollte verwendet werden, um die Kommandozeile einer allgemeinen Prüfung und einer Gültigkeitsprüfung zu unterziehen.

`ressource`

Gibt die Ressource an, die ausgehängt werden soll. Format wie bei *mount*.

Für bs2fs-Dateisysteme muss die Option entsprechend dem Eintrag in den internen Tabellen in GROSSBUCHSTABEN angegeben werden. Sonderzeichen der POSIX-Shell wie '$' oder '*' müssen explizit entwertet werden.

Wird bei NFS-Ressourcen durch den Namen des bereitstellenden Servers ersetzt, auf den ein Doppelpunkt und der Pfadname der Ressource folgen müssen.

`einhängepunkt`

Gibt an, wo die *ressource* lokal ausgehängt werden soll. Es muss ein absoluter Pfadname angegeben werden.

Hinweis

Für das Aushängen von bs2fs-Dateisystemen wird grundsätzlich die Angabe der Option *einhängepunkt* empfohlen. Ist ein bs2fs-Dateisystem mehrfach an verschiedenen Stellen im POSIX-Dateisystem gemounted (d. h. identische *ressource*-Angabe bei *mount*) und bei *umount* wird nur *ressource* angegeben, dann wird nur das zuletzt eingehängte Dateisystem ausgehängt. Bei Angabe von *einhängepunkt* wird dagegen immer das entsprechende Dateisystem ausgehängt.

Das Kommando *umount* wird abgewiesen, wenn es sich auf ein mit der Option *bs2fscontainer* gemountetes Dateisystem bezieht und noch mindestens ein bs2fs-Dateisystem eingehängt ist. Bei einem erfolgreichen *umount* des bs2fs-Containers werden die bs2fsd-Kopierdämonen automatisch beendet.

Datei

/etc/mnttab

Tabelle der eingehängten Dateisysteme

/etc/vfstab

Tabelle der automatisch eingehängten Dateisysteme

4.1.7 umountall - Aushängen mehrerer Dateisysteme

(unmount filesystems)

umountall bewirkt das Aushängen aller eingehängten Dateisysteme außer *root*, */proc*, */var* und */usr*. Ist **nur** der *dateisystemtyp* angegeben, beschränkt sich die Wirkung von *umountall* auf Dateisysteme des angegebenen Typs. Die Dateisysteme werden in der Reihenfolge *nfs* - *bs2fs* - *ufs* ausgehängt. Somit ist gewährleistet, dass das für bs2fs-Dateisysteme erforderliche bs2fscontainer-Dateisystem im ufs erst ausgehängt wird, wenn es nicht mehr benötigt wird, d.h. wenn keine bs2fs-Dateisysteme mehr eingehängt sind.

Syntax

```
umountall[ -F dateisystemtyp][ -k][ -l | -r| -b]
```

-F

Angabe des Dateisystem-Typs, der ausgehängt werden soll.

-k

Sendet das Signal SIGKILL an Prozesse, die Dateien im Dateisystem geöffnet haben.

-l

Begrenzt den Vorgang auf lokale Dateisysteme (*ufs* und *bs2fs*).

-r

Begrenzt den Vorgang auf ferne Dateisystem-Typen (*nfs*).

-b

Begrenzt den Vorgang auf bs2fs-Dateisysteme.

Fehler

Wenn die Dateisysteme aushängbar sind, wird keine Meldung ausgegeben. Fehler- und Warnmeldungen stammen von *fsck* und *mount*.

Hinweis

Wenn die Option *-F* in Kombination mit einer oder mehrerer der Optionen *-l*, *-r* und *-b* angegeben wird und die Optionen miteinander verträglich sind, so haben die Optionen *-l*, *-r* und *-b* Vorrang. Beispielsweise haben *umountall -F bs2fs -l* und *umountall -F ufs -l* dieselbe Wirkung wie *umountall -l* : Es werden alle lokalen Dateisysteme (also alle ufs- und bs2fs-Dateisysteme) ausgehängt. Ebenso führen die Angaben *umountall -F bs2fs* und *umountall -b* zum gleichen Ergebnis: alle bs2fs-Dateisysteme werden ausgehängt.

Datei

/etc/mnttab

Tabelle der eingehängten Dateisysteme

/etc/vfstab

Tabelle der automatisch eingehängten Dateisysteme

4.1.8 Kopierdämon bs2fsd

Dieser Abschnitt beschreibt den Kopierdämon *bs2fsd*.

Die Darstellungsmittel, die in der Kommandosyntax verwendet werden, finden Sie in "Darstellungsmittel".

Der Kopierdämon *bs2fsd* überträgt BS2000-Dateien und Elemente von PLAM-Bibliotheken aus dem BS2000 in den entsprechenden Bereich des bs2fs-Containers und umgekehrt.

Die Übertragung einer Datei oder eines Bibliothekselements vom BS2000 in den bs2fs-Container erfolgt automatisch beim ersten Zugriff auf die Datei im bs2fs-Dateisystem (erster *open*), bei schreibendem Zugriff wird nach dem Abschluss der Bearbeitung im POSIX (letzter *close*) die Rückübertragung ins BS2000 durchgeführt.

Falls die Datei bzw. das Bibliothekselement zum Schreiben geöffnet wird, bleibt sie/es vom Beginn der Übertragung aus dem BS2000 bis zum Abschluss der Rückübertragung ins BS2000 im BS2000 geöffnet. Sie/es kann daher während der Bearbeitung im POSIX von anderen Anwendern im BS2000 nicht verändert werden. Dadurch ist ein konsistenter Zustand der Datei gewährleistet.

Standardmäßig werden beim Einhängen des bs2fs-Containers 2 *bs2fsd*-Dämonen gestartet. Der Systemadministrator hat jedoch die Möglichkeit, bei Bedarf (z. B. viele bs2fs-Dateisysteme, viele Dateien, große Dateien) mit dem Kommando *start_bs2fsd* weitere Kopierdämonen zu starten.

Die Dämonen laufen als POSIX-Hintergrundprozesse weitgehend in TU unter der Benutzerkennung TSOS ab. Nach ihrem Start "schlafen" sie, bis sie durch eine Anforderung "geweckt" werden.

Die Dämonen können Sie überwachen, indem Sie z.B. folgendes POSIX-Kommando eingeben:

```
ps -ef | grep bs2fsd
```

Außerdem können Sie mit dem Kommando *start_bs2fsd* ermitteln, wieviele Kopierdämonen im Moment laufen.

Beim Aushängen des bs2fs-Containers werden alle Kopierdämonen automatisch beendet.

4.1.9 Verwaltungsdateien

Dieser Abschnitt beschreibt die Verwaltungsdateien.

Für den Einsatz von bs2fs-Dateisystemen werden, wie für die Dateisystemverwaltung in POSIX üblich, die Verwaltungsdateien */etc/mnttab* und */etc/vfstab* benutzt.

Die Datei */etc/mnttab* wird von POSIX beim Start automatisch angelegt und bei jedem *mount*-/*umount*-Vorgang aktualisiert.

Die Einträge für ufs-Dateisysteme und systemeigene Dateisysteme in der Datei */etc/vfstab* werden vom POSIX-Installationsprogramm erstellt. Einträge für nfs- und bs2fs-Dateisysteme können vom Administrator (root-Berechtigung) mit einem Editor bearbeitet werden.

In der folgenden Übersicht sind die Verwaltungsdateien aufgelistet:

Datei	**Funktion**
/etc/mnttab	Tabelle der eingehängten Dateisysteme
/etc/vfstab	Tabelle der definierten Dateisysteme

Tabelle 2: Verwaltungsdateien

4.1.10 /etc/mnttab - Tabelle der eingehängten Dateisysteme

Die Datei */etc/mnttab* enthält Angaben über alle am lokalen Rechner eingehängten Dateisysteme. Diese Datei enthält Informationen, die durch das Kommando *mount* erzeugt werden.

Jede Zeile enthält folgende Informationen, die durch eine beliebige Anzahl von Leerzeichen und/oder Tabulatoren getrennt sind:

Aufbau

resource	mountp	fstype	spec-options	time

`resource`

Absoluter Pfadname des eingehängten Dateisystems bzw. bei bs2fs-Dateisystemen eingehängte BS2000-Dateien in Wildcard-Syntax.

Für bs2fs-Dateisysteme weicht der Eintrag folgendermaßen von der Angabe beim *mount*-Kommando ab:

- Er ist vollständig in Großbuchstaben umgewandelt
- Zu entwertende Zeichen werden ohne den zugehörigen Entwerter dargestellt

Beispiel:

`mount –F bs2fs –o ftyp=text :v70a:\$bach.sys\* /home/bach/bs2.1`

erzeugt folgenden Eintrag in */etc/mnttab*:

`:V70A:$BACH.SYS*` `/home/bach/bs2.1 bs2fs ftyp=text ...`

`mountp`

Absoluter Pfadname des Einhängepunkts.

`fstype`

Dateisystem-Typ.

`spec-options`

Optionen, wie sie beim *mount*-Kommando angegeben wurden.

`time`

Einhängezeitpunkt, angegeben in Sekunden seit 1.1.1970

Einträge in der Datei */etc/mnttab* werden wieder gelöscht, wenn die Kommandos *umount* oder *umountall* für entsprechende Dateisysteme oder Dateisystemtypen ausgeführt werden.

Beispiel

Geben Sie in der POSIX-Shell ein: `cat /etc/mnttab`

```
/dev/root            /                  ufs    rw,suid                          1196069614
/proc                /proc              proc   rw                               1196069614
/dev/fd              /dev/fd            fdfs   rw                               1196069614
/dev/dsk/3           /var               ufs    suid,rw,noquota                  1196069614
/dev/dsk/4           /home/froede       ufs    suid,rw,noquota                  1196069615
/dev/dsk/10          /home/gast         ufs    suid,rw,noquota                  1196069615
/dev/dsk/5           /home/bach         ufs    suid,rw,noquota                  1196069621
/dev/dsk/2           /home/bach/mount99 ufs    suid,rw,noquota                  1196069621
/dev/dsk/23          /bs2fscont         ufs    bs2fscontainer,suid,rw,noquota 1196070061
:V70A:$BACH.ASS.*.S  /home/bach/bs2.1   bs2fs  ftyp=binary                      1196084245
:V70A:$BACH.CCC.*.C  /home/bs2.2        bs2fs  ftyp=text                        1196084250
:V70A:$BACH.PLAMLIB* /home/bach/bs2.2   bs2fs  ftyp=textbin                     1196084255
:V70A:$BACH.SEM*.C   /home/bs2000       bs2fs  ftyp=text                        1196084261
```

4.1.11 /etc/vfstab - Tabelle der definierten Dateisysteme

Die Datei */etc/vfstab* beschreibt jedes auf dem lokalen Rechner definierte Dateisystem. Die Datei können Sie mit einem Editor bearbeiten.

Die Dateisysteme, die in der Datei */etc/vfstab* mit *yes* in der Spalte *automnt* eingetragen sind, werden beim POSIX-Start bzw. durch das Kommando *mountall* automatisch eingehängt.

Außerdem werden die Einträge in der Datei benutzt, um bei der Ausführung eines *mount*-Kommandos ggf. fehlende Angaben für *ressource* oder *einhängepunkt* sowie *mount*-Optionen zu ergänzen.

Für ufs-Dateisysteme, die mit dem POSIX-Installationsprogramm definiert werden, werden automatisch entsprechende Einträge in der Datei */etc/vfstab* erzeugt. Für alle anderen Dateisysteme (z.B. bs2fs- oder nfs-Dateisysteme) sind die Einträge bei Bedarf manuell zu erstellen.

Im Gegensatz zur Datei */etc/mnttab* hat die Ausführung der Kommandos *mount* und *umount* auf die Datei */etc/vfstab* keine Auswirkungen. Entsprechende Einträge bleiben erhalten.

Die Felder in der Tabelle sind durch Tabulatoren und/oder Leerzeichen getrennt. Ein Bindestrich (–) kennzeichnet einen leeren Eintrag im Feld. Die Tabelle enthält folgende Felder:

Aufbau

special	fsckdev	mountp	fstype	fsckpass	automnt	mntopts

`special`

Beschreibt die einzuhängende Ressource.

Bei (manuellen) Einträgen für bs2fs-Dateisysteme ist Folgendes zu beachten:

- Buchstaben dürfen nur in Großschreibung angegeben werden
- Sonderzeichen dürfen nicht entwertet werden, ebenso ist das Einschließen der Zeichenfolge in Hochkommata nicht zulässig

`fsckdev`

Name des blockorientierten Geräts bzw. der Ressource des zeichenorientierten Geräts.

`mountp`

Einhängepunkt: absoluter Pfadname des Verzeichnisses, in dem die Ressource eingehängt werden soll.

`fstype`

Dateisystem-Typ.

`fsckpass`

Ist die für mehrere *fsck*-Kommandos zu verwendende Durchlaufnummer.

`automnt`

Gibt an, ob die Ressource automatisch beim Start von POSIX bzw. durch das Kommando *mountall* eingehängt werden soll (*yes*) oder nicht (*no*).

`mntopts`

Liste durch Kommata getrennter Optionen für das Einhängen des Dateisystems. Die Optionen entsprechen den *spez_optionen* des Kommandos *mount*.

Beispiel

Geben Sie in der POSIX-Shell ein: `cat /etc/vfstab`

```
/dev/root              /dev/rroot     /                      ufs    1   yes  -
/proc                  -              /proc                  proc   -   no   -
/dev/fd                -              /dev/fd                fdfs   -   no   -
/dev/dsk/3             /dev/rdsk/3    /var                   ufs    1   yes  -
172.25.86.64:/home2/froede/SHARE  -   /home/froede/RETSINA   nfs    -   no   soft
PGOB0004:/home2/froede/SHARE      -   /home/froede/PGOB0004  nfs    -   no   soft
/dev/dsk/4             /dev/rdsk/4    /home/froede           ufs    1   yes  -
/dev/dsk/10            /dev/rdsk/10   /home/gast             ufs    1   yes  -
/dev/dsk/13            /dev/rdsk/13   /mnt/ascii             ufs    1   no   -
/dev/dsk/8             /dev/rdsk/8    /mnt/dat1              ufs    1   no   -
/dev/dsk/23            /dev/rdsk/23   /bs2fscont             ufs    1   no   -
/dev/dsk/24            /dev/rdsk/24   /home/bach/mount3      ufs    1   no   -
/dev/dsk/25            /dev/rdsk/25   /home/bach/mountxxx    ufs    1   no   -
/dev/dsk/26            /dev/rdsk/26   /home/bach/mountyyy    ufs    1   no   -
/dev/dsk/5             /dev/rdsk/5    /home/bach             ufs    1   yes  -
/dev/dsk/2             /dev/rdsk/2    /home/bach/mount99     ufs    1   yes  -o
/dev/dsk/6             /dev/rdsk/6    /suderlan              ufs    1   no   -
:V70A:$BACH.ASS.*.S    -              /home/bach/bs2.1       bs2fs  -   yes  ftyp=binary
:V70A:$BACH.CCC.*.C    -              /home/bs2.2            bs2fs  -   yes  ftyp=text
:V70A:$BACH.PLAMLIB*   -              /home/bach/bs2.2       bs2fs  -   yes  ftyp=textbin
:V70A:$BACH.SEM*.C     -              /home/bs2000           bs2fs  -   yes  -
```

4.2 Hinweise zu Shell-Kommandos und POSIX-Tools

In diesem Abschnitt sind Besonderheiten beim Zugriff auf Dateien von bs2fs-Dateisystemen über Shell-Kommandos und POSIX-Tools beschrieben.

4.2.1 df - Anzahl der freien und belegten Plattenblöcke und I-Nodes ausgeben

(report free disk space)

Im Kommando *df* wird implizit die Schnittstelle *statvfs()* (64-Bit-Variante) benutzt. Im Falle eines bs2fs-Dateisystems sind die ausgegebenen Daten dann entsprechend zu interpretieren (siehe auch *statvfs()* in "fstatvfs, statvfs - Dateisystem-Informationen"). Für den Dateisystemtyp bs2fs werden alle 3 Formate des Kommandos *df* unterstützt.

Syntax

```
Format 1:  df[ -F FSType] -P[ -lV][ -k | -h | -H][ file]
Format 2:  df[ -F FSType][ -beglntVv][ -k | -h | -H][ -o ufs_options]...[ file]
Format 3:  df -c[ file]...
```

bs2fs-spezifische Optionen und Regeln des df-Kommandos:

`-F bs2fs`

identifiziert ein bs2fs-Dateisystem

`datei`

Es kann ein Pfadname oder die Ressource angegeben werden.

Pfadname:

bs2fs-Einhängepunkt, einfache bs2fs-Datei, PLAM-Bibliotheksverzeichnis, Typverzeichnis, Bibliothekselement

Ressource:

Ressource des *mount*-Kommandos für das bs2fs-Dateisystem gemäß dem *mnttab*-Eintrag in Großbuchstaben (z.B. 'V70A:$BACH.ASS.*.S').

4.2.2 dumpfs - Interne Dateisystem-Information ausgeben

(dump filesystem)

Das Kommando *dumpfs* wird für bs2fs-Dateisysteme nicht unterstützt.

4.2.3 fsck - Konsistenzprüfung des Dateisystems und Korrektur im Dialog mit dem Benutzer

(filesystem check)

Das Kommando *fsck* wird für bs2fs-Dateisysteme nicht unterstützt.

4.2.4 fsexpand - Existierende Dateisysteme vergrößern

(expand existing file systems)

Das Kommando *fsexpand* wird für bs2fs-Dateisysteme nicht unterstützt.

4.2.5 pathchk - Pfadnamen überprüfen

(check pathnames)

Bei Dateien oder Verzeichnissen in einem bs2fs-Dateisystem wird eine Fehlermeldung ausgegeben, wenn die Namenslänge nicht den Regeln eines bs2fs-Dateisystems entspricht.

4.2.6 pdbl/posdbl - Benutzerspezifischen/globalen Programm-Cache einrichten und verwalten

(set up and manage user-specific/global program cache)

Das Vorladen von im bs2fs-Dateisystem liegenden Executables (LMS-Element, Typ L oder einfache "binary"-DVS-Datei) mit posdbl oder pdbl (-b Schalter) wird unterstützt.

Syntax (nur relevante Schalter)

```
pdbl{ -s[ sid]| -u} -b pfad

posdbl -b pfad
```

Beispiel für ein LLM-Element

```
# posdbl -s
POSIX-DBL:              linker ON       loader ON
POSIX-DBL:              holder of global cache: TSN=031F PID=19
Cache  POSIX@DBL        CREATED: 04/10/08 12:24:46
                        SIZE: 32 MB     ENTRIES: 16
                        FREE PAGES: 7112


# posdbl -l
 RM                 39 Apr 10 12:25:30 $TSOS.SINLIB.POSIX-BC.070.SHELL
+IN@RLOGIND        113 Apr 11 11:16:47 $TSOS.SINLIB.POSIX-BC.070.ROOT
+HD                 35 Apr 10 12:25:08 $TSOS.SINLIB.POSIX-BC.070.ROOT
 PS                 49 Apr 11 11:33:22 $TSOS.SINLIB.POSIX-BC.070.SHELL
 SH                243 Apr 11 12:20:29 $TSOS.SINLIB.POSIX-BC.070.SHELL
 MAN                37 Apr 11 12:14:39 $TSOS.SINLIB.POSIX-SH.070
 WHO                46 Apr 10 15:25:43 $TSOS.SINLIB.POSIX-SH.070
 MORE              107 Apr 11 12:14:39 $TSOS.SINLIB.POSIX-SH.070
 SORT               53 Apr 11 12:20:32 $TSOS.SINLIB.POSIX-BC.070.SHELL
 CRON               59 Apr 10 12:25:31 $TSOS.SINLIB.POSIX-SH.070
 GREP               41 Apr 11 12:02:40 $TSOS.SINLIB.POSIX-BC.070.SHELL
+STTY               39 Apr 11 11:17:06 $TSOS.SINLIB.POSIX-BC.070.ROOT
 EXPR               40 Apr 10 12:24:57 $TSOS.SINLIB.POSIX-BC.070.SHELL
 UNAME              35 Apr 11 11:17:00 $TSOS.SINLIB.POSIX-BC.070.SHELL
+POSDBL             44 Apr 11 12:38:53 $TSOS.SINLIB.POSIX-BC.070.ROOT
+IN@RSHD            97 Apr 10 12:25:11 $TSOS.SINLIB.POSIX-BC.070.INET

# posdbl -b /home/bach/bs2/plamlib.4/l/usp
```

```
# posdbl -l
 RM                 39 Apr 10 12:25:30 $TSOS.SINLIB.POSIX-BC.070.SHELL
+IN@RLOGIND        113 Apr 11 11:16:47 $TSOS.SINLIB.POSIX-BC.070.ROOT
+HD                 35 Apr 10 12:25:08 $TSOS.SINLIB.POSIX-BC.070.ROOT
 PS                 49 Apr 11 11:33:22 $TSOS.SINLIB.POSIX-BC.070.SHELL
 SH                243 Apr 11 12:20:29 $TSOS.SINLIB.POSIX-BC.070.SHELL
+usp                59 Apr 11 12:41:02 /home/bach/bs2/plamlib.4/l/usp
 MAN                37 Apr 11 12:14:39 $TSOS.SINLIB.POSIX-SH.070
 WHO                46 Apr 10 15:25:43 $TSOS.SINLIB.POSIX-SH.070
 MORE              107 Apr 11 12:14:39 $TSOS.SINLIB.POSIX-SH.070
 SORT               53 Apr 11 12:20:32 $TSOS.SINLIB.POSIX-BC.070.SHELL
 CRON               59 Apr 10 12:25:31 $TSOS.SINLIB.POSIX-SH.070
 GREP               41 Apr 11 12:02:40 $TSOS.SINLIB.POSIX-BC.070.SHELL
+STTY               39 Apr 11 11:17:06 $TSOS.SINLIB.POSIX-BC.070.ROOT
 EXPR               40 Apr 10 12:24:57 $TSOS.SINLIB.POSIX-BC.070.SHELL
 UNAME              35 Apr 11 11:17:00 $TSOS.SINLIB.POSIX-BC.070.SHELL
+POSDBL             44 Apr 11 12:41:18 $TSOS.SINLIB.POSIX-BC.070.ROOT
+IN@RSHD            97 Apr 10 12:25:11 $TSOS.SINLIB.POSIX-BC.070.INET
```

4.3 Hinweise zu C-Programmschnittstellen

In diesem Abschnitt sind Besonderheiten beim Zugriff auf Dateien von bs2fs-Dateisystemen über Programmschnittstellen beschrieben.

Über die Schnittstelle *sysfs* (siehe Handbuch "C-Bibliotheksfunktionen für POSIX-Anwendungen" und "sysfs - Information über Dateisystemtyp abfragen") können Sie ermitteln, ob in einer vorhandenen POSIX-Installation bs2fs-Dateisysteme unterstützt werden.

Zur Prüfung, ob ein bs2fs-Dateisystem eingehängt ist, können Sie das Kommando *mount* verwenden oder die *mount*-Tabelle */etc/mnttab* durchsuchen (siehe "/etc/mnttab - Tabelle der eingehängten Dateisysteme"). Diese Tabelle enthält pro eingehängtem Dateisystem einen Eintrag der Daten dieses Dateisystems.

4.3.1 errnos

Auf Dateien von bs2fs-Dateisystemen können Sie mit denselben Programmschnittstellen zugreifen wie auf Dateien von ufs-Dateisystemen. Allerdings kann die Variable *errno* mitunter im Fehlerfall zusätzliche Werte oder Werte mit unterschiedlicher Bedeutung enthalten. Die Bedeutung der entsprechenden Fehlernummern beim Zugriff auf bs2fs-Dateisysteme finden Sie in der folgenden Tabelle:

Fehlernummer	**Bedeutung**
EAGAIN	Auf die BS2000-Datei kann vorübergehend nicht zugegriffen werden, z.B. weil die Datei von einem anderen Prozess geöffnet ist, oder weil die Verbindung zum Pubset unterbrochen ist.
EDOM	Interner Schnittstellenfehler
EFAULT	Interner Verarbeitungsfehler oder ungültige Adresse (z.B. Pufferadresse)
EIO	DMS oder LMSUP oder PLAM meldete I/O-Fehler
EMACRO	DMS oder LMSUP oder PLAM meldet Verarbeitungsfehler
ENAME	Ein Bestandteil des Pfadnamens ist kein gültiger BS2000-Dateiname, PLAM-Typname oder Elementname (letzterer ggf. einschließlich gültiger Versionsbezeichnung)
ENODEV	auf den Datenträger oder das Gerät kann nicht zugegriffen werden
ENOEXEC	Ein zu bearbeitendes LLM-Element besitzt keine LLM-Struktur.
ENONET	Pubset ist nicht verfügbar, möglicherweise wegen Problem im MSCF-Verbund
ENOSTR	Es wird versucht, auf eine BS2000-Datei mit nicht unterstütztem Dateityp zuzugreifen
ENOSUBSYS	Es konnte keine Verbindung zu LMSUP oder PLAM hergestellt werden
ENOSYS	Operation wird vom bs2fs-Dateisystem nicht unterstützt
ENXIO	Datei wurde verdrängt (migriert)
EOPR	Operation wird für die angegebene Datei nicht unterstützt
ESRCH	Es existiert kein bs2fs-Dämon, der die Zugriffe auf das BS2000-Objekt ausführen könnte. Maßnahme: Systemverwalter verständigen

Fehlerwerte wie *EIO*, *ENOEXEC* oder *ENOSPC* können auch bei *close* gesetzt werden, wenn die Datei schreibend geöffnet war und nicht ordnungsgemäß in die BS2000-Datei oder das Element zurückgeschrieben werden konnte. In diesen Fällen bleibt die Kopie im bs2fs-Container erhalten.

4.3.2 fstat, stat - bs2fs-Dateistruktur

Die stat-Stuktur der Systemcalls *stat()* bzw. *fstat()* liefert alle wichtigen Informationen zu einer POSIX-Datei. Diese Informationen werden u.a. auch vom POSIX-Kommando *ls* ausgewertet und aufbereitet.

Im Falle von bs2fs-Dateien gibt es einige Besonderheiten, die in der folgenden Tabelle aufgelistet sind.

In einigen Fällen ist die Information unterschiedlich, je nachdem ob die Datei gerade geöffnet oder geschlossen ist. Im geöffneten Zustand befindet sich die Datei als "Schatten-Datei" im bs2fs-Container.

Mit *stat()* lassen sich die Informationen sowohl für geschlossene als auch für geöffnete Objekte abrufen, mit *fstat()* nur für geöffnete Objekte.

	Information für geschlossenes bs2fs-Objekt (stat)	**Information für geöffnetes bs2fs-Objekt (stat oder fstat)**
st_ino	Eindeutige Nummer zwischen einem bs2fs-*mount* und *umount*	
st_dev	Eindeutige Nummer des bs2fs-Dateisystems, in dem die mit *st_ino* gekennzeichnete Datei liegt.	
st_nlink	Bei DVS-Dateien: "normalerweise" 1 Bei Bibliothekselementen mit höchster Version: 2 bei anderen Bibliothekselementen:1	
st_size	DVS-Datei: letzte beschriebene PAM-Seite in Byte Bibliothekselement: Größe PAM-Seiten in Byte	Größe der Schattendatei im Container
st_uid	bs2-POSIX-user	
st_gid	bs2-POSIX-group	
st_atime **st_mtime** **st_ctime**	Eigenschaften der BS2000-Datei bzw. des Bibliothekselements Bei Bibliothekselementen hat *st_atime* den gleichen Wert wie bei der Bibliothek oder einen elementspezifischen Wert, falls LMS-aktiviert	Eigenschaften der Schattendatei im Container
st_mode	Eigenschaften der BS2000-Datei bzw. des Bibliothekselements	

Erläuterung zu den Zeitstempeln

st_atime

Beim Öffnen zum Lesen erfolgt der Zeitstempel-Update der BS2000-Datei im Rahmen des BS2000-Close, der sofort nach dem ggf. notwendigen Kopieren der Schattendatei in den Container erfolgt. Während der lesenden Zugriffe im Container erfolgt der Zeitstempel-Update nach jedem lesenden Zugriff auf die Schattendatei. Die Zeitstempel der BS2000- und der Schattendatei laufen dann auseinander, d.h. die BS2000-Datei hat prinzipiell einen "älteren" atime-Stand als die Schattendatei. (Solange die Datei geöffnet ist, sieht die Anwendung immer nur den Zeitstempel der Schattendatei. Nach dem *close()* sieht sie dagegen wieder den "älteren" Zeitstempel der BS2000-Datei.)

Für Bibliothekselemente wird üblicherweise kein Zugriffszeitstempel geführt, das Feld *st_atime* enthält dann den Zugriffszeitstempel der Bibliothek. Nur wenn für die Bibliothek mit der LMS-Anweisung //MODIFY-LIBRARY-ATTRIBUTES mit dem Operanden ACCES-DATE=*KEEP oder mit der entsprechenden LMSUP-Funktion das Führen von Zugriffszeitstempeln festgelegt wurde, werden ab diesem Zeitpunkt die Zugriffszeitstempel für deren Elemente bei Zugriffen aktualisiert und können dann ins Feld *st_atime* übernommen werden.

st_ctime

Nach *chmod()* wird das Feld *st_ctime* des Dateistatus nicht verändert; *st_ctime* repräsentiert wie bei BS2000-Dateien den Zeitpunkt der Erzeugung (CREATION TIME bei FSTAT).

4.3.3 fstatvfs, statvfs - Dateisystem-Informationen

Die Programmschnittstellen *fstatvfs* und *statvfs* liefern Informationen über das Dateisystem, in dem eine Datei liegt. Diese Datei wird bei *fstatvfs* über einen Dateideskriptor (filedescriptor) und bei *statvfs* über ihren Namen spezifiziert.

In beiden Fällen erfolgt die Ausgabe in dem Format, das in der Struktur *statvfs_t* im Header *sys/statvfs.h* definiert ist.

```
typedef struct statvfs {
  unsigned long f_bsize;         /* Blockgröße des Dateisystems */
  unsigned long f_frsize;        /* Fragmentgröße */
  unsigned long f_blocks;        /* # Blöcke auf Dateisystem mit Größe f_frsize */
  unsigned long f_bfree;         /* # freie Blöcke mit Größe f_frsize */
  unsigned long f_bavail;        /* # verfügbare freie Blöcke für Nicht-Superuser */
  unsigned long f_files;         /* # Dateiknoten (inodes) */
  unsigned long f_ffree;         /* # freie Dateiknoten (inodes) */
  unsigned long f_favail;        /* # verfügbare freie Inodes für Nicht-Superuser */
  unsigned long f_fsid;          /* Dateisystem-Id (Nummer) */
  char f_basetype[FSTYPSZ];      /* Name des Ziel-Dateisystems, Null-terminiert */
  unsigned long f_flag;          /* Bitmaske der Optionen von f_flag */
  unsigned long f_namemax;       /* maximale Länge der Dateinamen */
  char f_fstr[32);               /* dateisystemspezifische Zeichenkette */
  unsigned long f_filler[16];    /* reserviert für zukünftige Versionen */
} statvfs_t;
```

Dieses Format ist allerdings auf ufs-Dateisysteme zugeschnitten. Bei einem bs2fs-Dateisystem muss der Anwender diese Information entsprechend der folgenden Tabelle interpretieren. Die Daten werden mit Hilfe der BS2000-Schnittstelle $SJINFO erzeugt.

`unsigned long f_bsize`

Wird auf 2048 gesetzt (PAMBLOCKSIZE).

`unsigned long f_frsize`

Wird auf 2048 gesetzt (PAMBLOCKSIZE).

`unsigned long f_blocks` *(*)*

Diese Daten werden aus den entsprechenden Werten der BS2000-Schnittstelle $SJINFO gewonnen. Allerdings lassen sich nicht alle BS2000-Werte auf ein exaktes Äquivalent abbilden, das für ufs-Dateisystemen verwendet wird.

`unsigned long f_bfree` *(*)*

`unsigned long f_bavail` *(*)*

`unsigned long f_files` *(*)*

`unsigned long f_ffree` *(*)*

`unsigned long f_favail` *(*)*

`unsigned long f_fsid`

Enthält den Index, der von der Schnittstelle *sysfs* verwendet wird.

`char f_basetype[FSTYPSZ]`

Enthält den String "bs2fs" zur Kennzeichnung.

`unsigned long f_flag`

Nur die Kennzeichnung ST_RDONLY ist von Bedeutung.

`unsigned long f_namemax`

Wird abhängig vom Katalog- und Benutzerkennung auf 38 - 41 gesetzt. Dieser Wert gilt nur für DVS-Dateien bzw. PLAM-Bibliotheken. Bei Bibliothekselementen oder Typen enthält die Variable den Wert für die übergeordnete PLAM-Bibliothek. Die Beschränkungen für Namen von Elementen und Elementtypen können nur mit *pathconf* ermittelt werden. Innerhalb der Beschränkung von "Elementname+Version" auf 89 Zeichen ist zu beachten, dass der Bestandteil Elementname auf 64 Zeichen und der Bestandteil Version auf 24 Zeichen begrenzt ist.

`char f_fstr[32]`

(*) unsigned long long bei *fstatvfs64()* oder *statvfs64()*

4.3.4 sysfs - Information über Dateisystemtyp abfragen

Die Programmschnittstelle *sysfs* liefert Informationen über die in einem System konfigurierten Dateisystemtypen.

Abhängig vom Funktionsargument liefert *sysfs*

- den Index der dem Dateisystemtyp mit dem angegebenen Namen im System entspricht
- den Namen der dem Dateisystemtyp mit dem angegebenen Index entspricht
- die maximale, im System konfigurierte Anzahl der Dateisystemtypen

Mit dieser Schnittstelle kann geprüft werden, ob ein System fähig ist, bs2fs-Dateien zu verarbeiten. Der Argument für den Dateisystemtyp bs2fs lautet "bs2fs".

Beispiel

```
if (sysfs(GETFSIND, "bs2fs") < 0) {
    /* bs2fs NOT SUPPORTED */
}
```

4.3.5 Weitere Besonderheiten und Einschränkungen

Bei folgenden Programmschnittstellen existieren Einschränkungen beim Zugriff auf Dateien von bs2fs-Dateisystemen:

chmod()

Die Funktion wird nur auf Datei- und Elementebene akzeptiert. Andernfalls wird sie mit EISDIR zurückgewiesen. Die Bits S_ISUID ("set user id on execution") und S_ISGID ("set group id on execution") sind für BS2000-Dateien und Bibliothekselemente nicht definiert und werden ignoriert (falls bei chmod angegeben). Das Feld st_ctime des Dateistatus wird nicht verändert, es repräsentiert wie bei BS2000-Dateien den Zeitpunkt der Erzeugung (creation time, vgl. *fstat*).

chown(), fchown()

Die Funktionen werden mit ENOSYS zurückgewiesen, falls versucht wird, die Benutzer- und/oder Gruppennummer zu ändern.

close()

Nach Fehlern bei *close()* (z.B. mit den errno-Werten EIO, ENOEXEC, ENOSPC) bleibt die Schattendatei erhalten. Sie kann mit Unterstützung des POSIX-Verwalters zur Rettung der Daten genutzt werden.

creat(), open()

Die Funktionen werden nur auf Datei- und Elementebene akzeptiert. Andernfalls werden sie mit EISDIR zurückgewiesen. Eine Datei oder ein Element kann nicht gleichzeitig im BS2000 und in einem bs2fs-Dateisystem schreibend geöffnet sein. Ist eine Datei oder ein Element in einem bs2fs-Dateisystem schreibend geöffnet, kann sie bzw. es auch nicht in einem anderen bs2fs-Dateisystem geöffnet werden. Wird eine Datei nicht vom Eigentümer (userid des Dateisystems), sondern von einem Prozess mit TSOS-Privileg erzeugt, wird sie trotzdem der uid und gid des Eigentümers des Dateisystems zugeordnet. Existiert eine (im bs2fs-Dateisystem nicht sichtbare) Dateigenerationsgruppe mit dem angegebenen Dateinamen, wird der Aufruf mit ENOSTR zurück gewiesen.

Ein Bibliothekselement darf nicht nur der Eigentümer erzeugen, sondern jeder Benutzer, der Schreibrecht und ggf. das Administrationsrecht für die Bibliothek oder den Typ besitzt. Ist jedoch der Benutzer, der ein Element erzeugt, nicht der Eigentümer, werden die im mode-Argument angegebenen Schutzattribute ignoriert, und es werden Standardschutzattribute gesetzt.

creat(), open() mit O_EXCL:

Existiert eine im bs2fs-Dateisystem nicht sichtbare Datei, also eine mit nicht unterstützten Eigenschaften (STORAGE-LEVEL != S0, SUPPORT != PUBLIC), wird der Aufruf mit ENXIO zurückgewiesen. Die Datei bleibt dennoch (z.B. für das Kommando *ls*) unsichtbar, ihre Existenz kann aber mit "bs2cmd fstat ..." überprüft werden.

link()

Die Funktion wird nicht unterstützt. Der Aufruf wird mit ENOSYS zurückgewiesen.

mkdir(), rmdir()

Die Funktionen werden immer mit ENOSYS zurückgewiesen.

open() mit O_RDWR unter ftyp=text oder ftyp=textbin

Bei Textdateien (*ftyp=text* oder *ftyp=textbin*) vom Typ SAM erzwingt CRTE im Zusammenspiel mit SAM wegen *fopen(...,"r+")* im bs2fs-Dämon, dass die bestehende Satzstruktur erhalten bleibt, d.h. es muss in dem nach *open()* vorhandenen Dateibereich jedes Newline unverändert an der Stelle bleiben, an der es steht, und es darf kein Newline eingefügt werden. Am Ende der bisherigen Datei dürfen neue Sätze angefügt und verändert werden.

`readlink()`

Die Funktion wird immer mit `ENOSYS` zurückgewiesen.

`remove(), unlink()`

Die Funktionen werden nur auf Datei- und Elementebene akzeptiert. Andernfalls werden sie mit `EISDIR` zurückgewiesen.

`rename()`

Die Funktion wird nur auf Datei- und Elementebene akzeptiert. Andernfalls wird sie mit `EISDIR` zurückgewiesen. Es ist jedoch möglich, Dateien in Elemente umzuwandeln oder umgekehrt, z.B. mit dem *cp*- oder *mv*-Kommando. Elemente des Basistyps L können jedoch nicht in Dateien oder Textelemente umbenannt werden, Dateien und Elemente der Basistypen D, J, M, P, S, X können jedoch nicht in Elemente des Basistyps L umbenannt werden. Das Umbenennen eines Elements des Basistyps X in ein Text-Element (Basistyp D, J, M, P oder S) kann den Inhalt verfälschen. Ist das Ziel parallel im BS2000 oder einem anderen bs2fs-Dateisystem schreibend geöffnet, wird *rename()* mit `EAGAIN` abgewiesen.

`rmdir()`

Die Funktion wird immer mit `ENOSYS` zurückgewiesen.

`fstat(), stat()`

Beziehen sich *stat()* bzw. *fstat()* auf eine Datei in einem bs2fs-Dateisystem, so hängt die gelieferte Information davon ab, ob die Datei geöffnet ist oder nicht. Ist die Datei geöffnet, dann wird die Dateigröße bytegenau geliefert, und die Zeitstempel werden in Mikrosekundengenauigkeit geliefert, so wie es für POSIX-Dateien üblich ist. Ist die Datei nicht geöffnet, dann liefert *stat()* die Größe der Datei so wie sie im BS2000 angegeben wird (also in Vielfachen von 2K-Blöcken), und die Zeitstempel sind nur sekundengenau, die Felder mit den Mikrosekundenanteilen sind dann immer 0.

`symlink()`

Die Funktion wird immer mit `ENOSYS` zurückgewiesen.

`utime(), utimes()`

Die Funktionen werden nur auf Datei- und Elementebene akzeptiert. Andernfalls werden sie mit `EISDIR` zurückgewiesen. Das Feld `st_ctime` des Dateistatus wird nicht verändert (vgl. *chmod*). Das Führen der Zugriffszeitstempel (Access-Time `st_atime`) von Elementen ist eine spezielle Bibliothekseigenschaft, die mit LMS oder LMSUP gesetzt werden kann; standardmäßig werden für Elemente keine Zugriffszeitstempel erfasst; *stat()* gibt dann den Zugriffszeitstempel der Bibliothek aus. Zu welchem Zeitpunkt die Zeitstempel (Access-Time `st_atime`, Modification-Time `st_mtime`) für Verzeichnisse (Root-Verzeichnis eines bs2fs-Dateisystems, Verzeichnisse auf Bibliotheks- und Typebene) aktualisiert werden, ist undefiniert. Es kann nur der aktuelle Zeitstempel gesetzt werden. Weicht der angegebene Zeitwert um mehr als +/-3600 Sek vom aktuellen Zeitstempel ab, wird der Aufruf ignoriert.

5 Zugriff auf bs2fs-Dateisysteme über NFS

Dieses Kapitel beschreibt die Maßnahmen, die erforderlich sind, damit über NFS auf bs2fs-Dateisysteme zugegriffen werden kann, und welche Besonderheiten bei diesem Zugriff zu beachten sind.

5.1 Freigeben von bs2fs-Dateien

Die Freigabe von bs2fs-Dateien erfolgt mit dem NFS-Kommando *share*.

Syntax

```
share -F nfs [-o spez_optionen] [-d beschreibung] pfadname
```

Hierbei gelten im Vergleich zur Freigabe von ufs-Dateien folgende Besonderheiten:

Der angegebene *pfadname* muss das Root-Verzeichnis des bs2fs-Dateisystems sein. Er darf nicht auf ein Unterverzeichnis in diesem Dateisystem (PLAM-Bibliothek oder Elementtyp in einer PLAM-Bibliothek) verweisen.

Alle Client-Rechner, die Zugriff auf das freigegebene bs2fs-Dateisystem erhalten sollen, müssen explizit mit den Optionen **rw=** *client*[:*client*]... (für Lese-/Schreibzugriff) oder **ro=** *client*[:*client*]... (für Lesezugriff) angegeben werden.

Folgende spez_optionen wurden zusätzlich für die Freigabe von bs2fs-Dateien eingeführt:

bs2anon=`bs2000_uid`

Legt die BS2000-Benutzerkennung fest, unter der "anonyme" Zugriffe erfolgen sollen. Client-Prozesse mit einer UID ungleich der POSIX-UID des bs2fs-Eigentümers haben nur dann Zugriff auf das bs2fs-Dateisystem, wenn die Option *bs2anon* verwendet wird. Mit dieser Option kann der POSIX-Administrator genau eine BS2000-Kennung vereinbaren, mit deren Rechten diese Client-Prozesse auf die Dateien des jeweiligen bs2fs-Dateisystems zugreifen.

bs2conv

Legt fest, dass eine Zeichensatzkonvertierung stattfinden soll. Die Option hat keine Argumente. Ist diese Option angegeben, werden Dateiinhalte beim Lesen von EBCDIC.DF.04-1 nach ISO 8859-1 und beim Schreiben umgekehrt konvertiert.

> **i** Bei der Freigabe von bs2fs-Dateisysteme, die mit ftyp=text oder ftyp=textbin eingehängt sind, sollte diese Option immer angegeben werden, außer für Clients, die EBCDIC-Dateien verarbeiten. Soll ein Client ASCII-Dateien ohne Konvertierung auf einem bs2fs-Dateisystem verarbeiten, ist dieses mit ftyp=bin einzuhängen. Andernfalls entstehen Konflikte zwischen ASCII-Newlines (0x0A) und EBCDIC-Newlines (0x15).

bs2nameconv

Legt fest, dass eine Dateinamenkonvertierung stattfinden soll. Die Option hat keine Argumente.

Ist diese Option angegeben, werden bestimmte Dateinamen konvertiert (siehe "Konvertierung von Dateinamen").

5.2 Einhängen von freigegebenen bs2fs-Dateien

Das Einhängen von freigegebenen bs2fs-Dateisystemen erfolgt am NFS-Client mit den Mitteln, die das Client-Betriebssystems zur Verfügung stellt. Das ist in der Regel das *mount*-Kommando. Details sind in der NFS-Dokumentation des betreffenden Client-Betriebssystems nachzulesen, bei BS2000-Clients siehe Handbuch "NFS".

Hier gibt es keine Besonderheiten gegenüber dem Einhängevorgang von ufs-Dateien. Beachten Sie jedoch die Hinweise im "Empfehlungen für das Einhängen von bs2fs- Dateisystemen an NFS-Clients".

5.3 Sicherheitsaspekte und Zugriffsrechte

Damit den NFS-Clients nicht unbeabsichtigt Zugriffsrechte gewährt werden, die die Sicherheitskonzepte des BS2000 unterlaufen, gelten bei der Freigabe von bs2fs-Dateien folgende Restriktionen im Vergleich zur Freigabe von ufs-Dateien:

- Das Freigeben von bs2fs-Dateien mit dem *share*-Kommando ist nur unter der BS2000-Benutzerkennung TSOS erlaubt.
- Die Administration muss alle Client-Rechner, die Zugriff auf das freigegebene bs2fs-Dateisystem erhalten sollen, explizit mit den Optionen **rw=**... (für Lese-/Schreibzugriff) oder **ro=**... (für Lesezugriff) festlegen. Nur diesen Clients wird der Zugriff gestattet. Steht ein Client-Rechner in beiden Listen, erhält er nur lesenden Zugriff; steht er in keiner der Listen, wird ihm kein Zugriff auf das freigegebene bs2fs-Dateisystem gewährt.
- Client-Prozesse, die unter der UID ablaufen, die mit der POSIX-UID des Eigentümers (BS2000-Kennung) der bs2fs-Dateien übereinstimmt, erhalten Zugriff auf die freigegebenen bs2fs-Dateien mit den Rechten ihres Eigentümers.

  Für Client-Prozesse, die unter einer anderen UID ablaufen, kann mit der Option *bs2anon* des *share*-Kommandos eine BS2000-Kennung festgelegt werden, unter der diese Prozesse auf das freigegebene bs2fs-Dateisystem zugreifen können. Ist die Option nicht angegeben, erhalten diese Prozesse keinen Zugriff auf das bs2fs-Dateisystem.

  NFS-Clients können damit entweder unter der BS2000-Kennung des Eigentümers der freigegebenen bs2fs-Dateien zugreifen oder – sofern beim *share*-Kommando angegeben – unter der für "anonyme" Zugriffe zugewiesenen BS2000-Kennung.

  Die Vertrauenswürdigkeit der zugelassenen Clients muss der Administrator des NFS-Servers (BS2000) verifizieren – wie bei anderen NFS-Freigaben eigentlich auch. Dabei muss der Client-Administrator (root) vertrauenswürdig sein, denn er weist einem User seine UID zu. Auch Client-User mit einer UID, die der POSIX-UID des Eigentümers der bs2fs-Dateien entspricht, müssen im Rahmen der ihnen zugestandenen Zugriffsrechte vertrauenswürdig sein.

5.4 Konvertierung von Dateinamen

Der Namensraum von BS2000-Dateien und PLAM-Bibliothekselementen ist in der Regel kleiner als der des Client-Systems. Auf dem Client-System kann es daher Anwendungen geben, die mit den am Server geltenden Einschränkungen des Namensraumes nicht oder nur schwer zurechtkommen. Beispielsweise legen Editoren Sicherungskopien oder Zwischensicherungen unter Dateinamen ab, die auf dem bs2fs-Dateisystem am Server nicht zulässig sind.

NFS-Clients erzeugen beim Löschen von Dateien, die noch geöffnet sind, vorübergehend Dateien, deren Namen aus ".nfs" gefolgt von unterschiedlich vielen Hexadezimalziffern bestehen. Deshalb werden diese Dateinamen grundsätzlich auf folgende Weise konvertiert:

Dateiname am NFS-Client	**Dateiname im bs2fs-Dateisystem**
`.nfs[:xdigit:][:xdigit:]*`	`@nfs[:xdigit:][:xdigit:]*`

Mit der Option *bs2nameconv* des *share*-Kommandos kann bei der Freigabe eines bs2fs-Dateisystems festgelegt werden, dass zusätzliche Konvertierungen durchgeführt werden:

Dateiname am NFS-Client	**Dateiname im bs2fs-Dateisystem**	**Beschreibung des Dateinamen am NFS-Client und der Konvertierungsregel**
`\.[^.].*`	`@[^.].*`	Dateinamen mit mindestens zwei Zeichen, wobei das erste Zeichen ein Punkt ist und das zweite Zeichen kein Punkt: Der führende Punkt (.) wird in ein at-Zeichen (@) umgewandelt.
`..*~.*`	`..*#.*`	Dateinamen, die außer als erstes Zeichen die Tilde enthalten: Jede Tilde (~) wird in ein Doppelkreuz (#) umgewandelt.

5.5 Weitere Besonderheiten beim Arbeiten mit bs2fs-Dateien am NFS-Client

Bei der Verarbeitung von bs2fs-Dateisystemen, die an NFS-Clients eingehängt sind, müssen folgende Besonderheiten beachtet werden:

- Beim Erzeugen, Kopieren oder Umbenennen von Dateien oder Bibliothekselementen müssen die entsprechenden Namensregeln des BS2000 eingehalten werden. Ansonsten wird der Anwendung beim *open()* die Fehlernummer EINVAL ("invalid argument") zurück gemeldet. Das kann zum Beispiel zu Problemen führen bei:
  - Editoren, die automatisch Kopien der Originaldatei erzeugen oder Zwischenstände sichern (.*filename*.swp, *filename*~, ...)
  - Datei-Browsern, die Miniaturansichten (Thumbnails) von Dateien generieren und temporär im betreffenden Verzeichnis ablegen

  In der Regel ist das in den Editoren oder Browsern konfigurierbar, indem entweder die jeweilige Funktion deaktiviert wird oder ein Ablageort außerhalb des bs2fs-Dateisystems gewählt wird.
- Das Schließen einer bs2fs-Datei am NFS-Server wird aus Gründen der internen Performance-Optimierung geringfügig verzögert (2 Sekunden), sofern im bs2fs-Container kein Platzmangel herrscht. Dadurch ist eine Datei am NFS-Server noch 2 Sekunden lang geöffnet und aus Sicht des BS2000 auch noch so lange gesperrt, obwohl sie aus Sicht des NFS-Clients bereits geschlossen zu sein scheint.
- Der erste *open()* der bs2fs-Datei findet in der Regel statt, wenn die Client-Anwendung die Funktion *access()* aufruft. Das kann dazu führen, dass für den NFS-Client die Antwortzeit der NFS-Funktion *ACCESS* unerwartet lang ist. Insbesondere ist die Antwortzeit auch von der Größe der bs2fs-Datei abhängig (siehe auch "Empfehlungen für das Einhängen von bs2fs-Dateisystemen an NFS-Clients").
- Auf UNIX-Dateisystemen ist der Verweiszähler (hard link count) von Verzeichnissen gleich der Anzahl der Unterverzeichnisse plus zwei (wegen der Pseudo-Unterverzeichnisse "." und ".."). In bs2fs-Dateisystemen können der Einhängepunkt sowie PLAM-Bibliotheken Unterverzeichnisse enthalten. Für sie wird erst beim Lesen des Verzeichnisinhalts (Funktion *readdir*; wird z.B. vom ls-Kommando benutzt) der Verweiszähler auf diesen Wert gesetzt. Zuvor steht er auf 2 oder auf dem Wert, den der letzte Aufruf der *readdir*-Funktion geliefert hatte. Dies führt bei einigen UNIX-Anwendungen zu Problemen. Das *find*-Kommando auf LINUX-Clients (GNU-Variante) gibt z. B. unter Umständen eine Warnung aus, dass der Verweiszähler eines Verzeichnisses falsch sei und deswegen interne Optimierungsalgorithmen nicht korrekt funktionieren, und schlägt vor, diese Optimierung zu deaktivieren.
- Die Dateityp-Erkennung auf UNIX-Systemen erfolgt in der Regel durch eine heuristische Untersuchung eines kleinen Teils des Dateiinhalts. Dazu wird die Datei geöffnet, eine bestimmte Datenmenge gelesen und analysiert und die Datei wieder geschlossen. Bei bs2fs-Dateien muss hierzu zuerst die gesamte BS2000-Datei in den bs2fs-Container am NFS-Server kopiert werden, was umso länger dauert je größer die Datei ist. Das kann am NFS-Client zu sehr langen Antwortzeiten führen, die von der Gesamtgröße der Dateien in diesem Verzeichnis abhängen.

5.6 Empfehlungen für das Einhängen von bs2fs-Dateisystemen an NFS-Clients

An den NFS-Clients, an denen bs2fs-Dateisysteme eingehängt werden, können folgende Maßnahmen erforderlich sein:

- Ausschalten des Attribut-Caches

  Da die Größe einer bs2fs-Datei sich beim ersten *open()* verändern kann, sollten die Dateiattribute am NFS-Client nicht im Attribut-Cache abgelegt werden. Dazu bieten NFS-Clients z.B. die *mount*-Option *noac*.

- Erhöhte Antwortzeiten erlauben

  Wegen der speziell bei der NFS-Funktion *ACCESS* unerwartet hohen Antwortzeiten sind am NFS-Client die Timeout-Werte hinreichend hoch einzustellen. Dazu bieten NFS-Clients z.B. die *mount*-Option *timeo=n*. Je nach Größe der bs2fs-Dateien ist hier das Zehnfache vom Standardwert oder sogar noch mehr einzustellen.

  Sollen relativ große bs2fs-Dateien verarbeitet werden, kann es erforderlich sein, einen sog. "hard-mount" durchzuführen (in der Regel durch Weglassen der *mount*-Option *soft*). Dadurch werden Dateizugriffe nach Zeitüberschreitungen generell wiederholt.

- Funktion *READDIRPLUS* deaktivieren

  Die Verwendung der NFS-Funktion *READDIRPLUS* anstelle von *READDIR* soll die Anzahl der Netzwerk-IOs verringern und damit die Performance steigern. An einigen NFS-Clients sind tatsächlich jedoch unter Umständen Performance-Einbußen festzustellen. Sie bieten z.B. eine *mount*-Option *nordirplus* an, um die Nutzung von *READDIRPLUS* zu deaktivieren.

- Ein-/Ausgabe-Pufferung deaktivieren

  Einige NFS-Clients benutzen eine lokale Zwischenspeicherung der Ein-/Ausgabe-Daten, um die Anzahl der Netzwerk-IOs zu verringern. Das kann aber zur Verschleierung von eventuellen Ein-/Ausgabe-Fehlern auf bs2fs-Dateien führen, sodass die Anwendung am NFS-Client zu Unrecht von einer erfolgreichen Ein-/Ausgabe ausgeht. Deshalb sollte die Ein-/Ausgabe-Pufferung am NFS-Client deaktiviert werden. Dazu bieten NFS-Clients z.B. die *mount*-Option *forcedirectio* an.

5.7 Beispiele

Für die Beispiele in diesem Abschnitt gelten folgende Voraussetzungen: Am NFS-Server existieren die BS2000-Kennungen MORITZ (POSIX-UID=110), GAST (POSIX-UID=100) und TSOS (POSIX-UID=0).

Beispiel 1

Die Dateien `:DATA:$MORITZ.*.LIB` werden als bs2fs-Dateisystem eingehängt. Dieses Dateisystem wird für die Clients *client01* und *client02* zum Lesen und Schreiben und für den Client *client03* nur zum Lesen freigegeben. An den Clients wird einem User mit der UID 110 der Zugriff mit den Rechten der BS2000-Kennung *MORITZ* erlaubt. Jedem anderen User wird der Zugriff verweigert, weil die Option *bs2anon* nicht angegeben ist:

```
# mount -F bs2fs -o rw,ftyp=binary ':DATA:$MORITZ.*.LIB' /bs2mnt/moritz_lib
# share -F nfs -o rw=client01:client02,ro=client03       /bs2mnt/moritz_lib
```

Beispiel 2

Die Dateien `:DATA:$ MORITZ.*.SRC` werden als BS2000-Dateisystem eingehängt. Dieses Dateisystem wird für die Clients *client04* zum Lesen und Schreiben und für den Client *client05* nur zum Lesen freigegeben, wobei an den Clients dem User mit UID=110 der Zugriff mit den Rechten der BS2000-Kennung *MORITZ* und allen anderen der Zugriff mit den Rechten der BS2000-Kennung *GAST* erlaubt wird:

```
# mount -F bs2fs -o rw,ftyp=text ':DATA:$MORITZ.*.SRC' /bs2mnt/moritz_src
# share -F nfs -o rw=client04,ro=client05,bs2anon=GAST /bs2mnt/moritz_src
```

Beispiel 3

Die Dateien `:HOME:$TSOS.SYSDAT.BCAM.*` werden für den Super-User (UID=0) am Client *client06* zum Lesen und Schreiben freigegeben, andere User an diesem Client erhalten keinen Zugriff:

```
# mount -F bs2fs -o rw,ftyp=text ':HOME:$TSOS.SYSDAT.BCAM.*' /bs2mnt/bcam.dat
# share -F nfs -o rw=client06 /bs2mnt/bcam.dat
```

Beispiel 4

Die Dateien `:HOME:$SYSAUDIT.*CONSLOG*` werden für jeden beliebigen User am Client *client07* zum Lesen freigegeben, wobei die Dateiinhalte von EBCDIC.DF.04-1 nach ISO 8859-1 konvertiert werden. D.h. die Dateien enthalten EBCDIC-Texte und werden an den NFS-Clients als ASCII-Texte ausgegeben:

```
# mount -F bs2fs -o ro,ftyp=text ':HOME:$SYSAUDIT.*CONSLOG*' /bs2mnt/conslog
# share -F nfs -o ro=client07,bs2anon=SYSAUDIT,bs2conv       /bs2mnt/conslog
```

Beispiel 5

Die Dateien `:HOME:$MAX.*` werden für jeden User am Client *client08*, sofern die Zugriffsrechte es erlauben, zum Lesen freigegeben. Dabei werden die Dateiinhalte von EBCDIC.DF.04-1 nach ISO 8859-1 konvertiert, d.h. die Dateien enthalten EBCDIC-Texte und werden an den NFS-Clients als ASCII-Texte ausgegeben. Außerdem werden Dateinamen mit bestimmten Sonderzeichen konvertiert:

```
# mount -F bs2fs -o ro,ftyp=text ':HOME:$MAX.*'   /bs2mnt/max
# share -F nfs -o ro=client08,bs2conv,bs2nameconv /bs2mnt/max
```

6 Diagnose und Leistungsverbesserung

Dieses Kapitel gibt einen Einblick in interne Abläufe und beschreibt ein Diagnosehilfsmittel und Maßnahmen zur Leistungsverbesserung.

Die in diesem Kapitel enthaltenen technischen Einzelheiten sollen einem erfahrenen Systemverwalter die Diagnose erleichtern.

6.1 Der Ein- und Aushängevorgang im Überblick

Dieser Abschnitt beschreibt die internen Abläufe beim Ein- und Aushängen des bs2fs-Containers und von bs2fs-Dateisystemen.

Einhängen des bs2fs-Containers

Das Einhängen des ufs-Dateisystems mit der Eigenschaft bs2fs-Container (*mount -o bs2fscontainer...*) ist die Voraussetzung für das Einhängen von bs2fs-Dateisystemen. Der bs2fs-Container ist das physikalische Medium, in dem die Dateien von bs2fs-Dateisystemen temporär für die Dauer ihrer Bearbeitung abgelegt sind. Beim Einhängen des bs2fs-Containers wird erwartet, dass er leer ist. Sollte dies aufgrund eines Fehlers in einer vorangegangenen Session nicht der Fall sein, wird der Inhalt beim Einhängen gelöscht.

Ein ufs-Dateisystem kann nur dann als bs2fs-Container eingehängt werden, wenn es beim Anlegen (POSIX-Installationsprogramm) entsprechend gekennzeichnet wurde. Damit wird verhindert, dass ein nicht für die Nutzung als bs2fs-Container vorgesehenes ufs-Dateisystem versehentlich als bs2fs-Container eingehängt wird, wodurch sein Inhalt gelöscht würde. Als zusätzlicher Schutz wird empfohlen, den bs2fs-Container nach Möglichkeit automatisch einzuhängen.

Für die Anwender soll der bs2fs-Container unsichtbar sein. Zugriffe auf den Inhalt des bs2fs-Containers sollen ihnen nur indirekt möglich sein, d.h. durch Zugriff auf die Dateien in den eingehängten bs2fs-Dateisystemen. Diese werden intern auf die Dateien im bs2fs-Container umgelenkt.

Der direkte Zugriff auf den bs2fs-Container sollte nur durch den Systemverwalter und ausschließlich zu Diagnosezwecken erfolgen.

Aushängen des bs2fs-Containers

Der bs2fs-Container kann nur ausgehängt werden, wenn keine bs2fs-Dateisysteme mehr eingehängt sind.

Einhängen eines bs2fs-Dateisystems

Beim Einhängen eines bs2fs-Dateisystems wird im bs2fs-Container ein Verzeichnis mit folgendem Namen angelegt: *catid***.***userid***.***nummer*, wobei:

catid

Katalogkennung aus dem Operanden *ressource* des *mount*-Kommandos

userid

Benutzerkennung aus dem Operanden *ressource* des *mount*-Kommandos

nummer

Nummer, mit der die *mount*-Vorgänge für bs2fs-Dateisysteme innerhalb einer Session in chronologischer Reihenfolge nummeriert werden.

Dieses Verzeichnis dient dazu, Abbilder von Objekten (Datei, Bibliothek, Elementtyp oder Bibliothekselement) eines bs2fs-Dateisystems aufzunehmen, wenn diese mit *open()* oder *opendir()* zur Bearbeitung geöffnet werden. Diese Abbilder im bs2fs-Container sind grundsätzlich einfache Dateien, auch wenn das entsprechende Objekt im bs2fs-Dateisystem als Verzeichnis dargestellt wird. Die eingehängten BS2000-Dateien werden somit physikalisch nicht am Einhängepunkt des bs2fs-Dateisystems abgelegt, sondern liegen ("unsichtbar" für den Anwender) in einem für dieses bs2fs-Dateisystem reservierten Verzeichnis des bs2fs-Containers.

Außerdem existieren noch Verzeichnisse auf der obersten Ebene eines bs2fs-Dateisystems. Deren bei *open* oder *opendir* erzeugte Abbilddateien stellen jeweils prozesslokale Ansichten dieser Verzeichnisse dar und heißen %*userid*%*pid*, wobei *userid* die BS2000-Benutzerkennung (8 Zeichen mit Blanks) des Aufrufers von *open* oder *opendir* ist und *pid* dessen Prozess-Id, konkret z.B. "`%BACH  %00123`".

Folgende Tabelle zeigt, wie die Namen der Objekte im bs2fs-Dateisystem auf Dateinamen im bs2fs-Container abgebildet werden:

Typ im BS2000	**Name im bs2fs-Dateisystem**	**Typ im bs2fs-Dateisystem**	**Name der Datei im bs2fs-Container**
DVS-Datei	<datei>	Datei	<datei>
PLAM-Bibliothek	<bibl>	Verzeichnis	<bibl>%
Elementtyp	<bibl>/<typ>	Verzeichnis	<bibl>%<typ>%
Bibliothekselement	<bibl>/<typ>/<elem>	Datei	<bibl>%<typ>%<elem>+<ver>

wobei

<datei> Dateiname

<bibl> Bibliotheksname

<typ> Elementtyp

<elem> Elementname

<ver> Elementversion

Folgende Tabelle zeigt diese Abbildung noch einmal für konkrete Beispiele:

Objekt, das mit open() oder opendir() geöffnet wird	**Name im bs2fs-Dateisystem**	**Dateiname im bs2fs-Container**
DVS-Datei SOURCEFILE	sourcefile	sourcefile
PLAM-Bibliothek MYPLAMLIB	myplamlib	myplamlib%
Unterverzeichnis vom Typ S der PLAM-Bibliothek MYPLAMLIB	myplamlib/s	myplamlib%s%
Element SOURCE1 vom Typ S der PLAM-Bibliothek MYPLAMLIB (wobei die höchste Elementversion 001 ist)	myplamlib/s/source1	myplamlib%s%source1+001

Die Abbild-Dateien im bs2fs-Container werden beim ersten *open()* bzw. *opendir()* auf das entsprechende Objekt angelegt und beim letzten *close()* bzw. *closedir()* wieder gelöscht. Für Dateien oder Verzeichnisse, auf die auf andere Weise zugegriffen wird, existieren keine Abbild-Dateien im bs2fs-Container.

Aushängen eines bs2fs-Dateisystems

Beim Aushängen eines bs2fs-Dateisystems wird auch das entsprechende Verzeichnis im bs2fs-Container gelöscht.

6.2 Diagnose

Im Fehlerfall wird ggf. die Information benötigt, welchem Dateisystem im bs2fs-Container abgelegte Dateien entsprechen. Zur Ermittlung dieser Zuordnung steht das Kommando *get_container_index* zur Verfügung.

6.3 get_container_index Zuordnung von bs2fs-Container-Index zu bs2fs-Dateisystem

Dieses Kommando liefert dem Administrator die Zuordnung vom Index eines Verzeichnisses im bs2fs-Container zum entsprechenden bs2fs-Dateisystem.

Syntax

```
get_container_index [mountpoint]
```

Keine Option angegeben

Ausgabe erfolgt für alle eingehängten bs2fs-Dateisysteme.

mountpoint

Die Ausgabe erfolgt nur für das Dateisystem, zu dem die angegebene Datei gehört.

Beispiel 1

Auflisten der Information über alle eingehängten bs2fs-Dateisysteme

```
# get_container_index
index = 3 for /home/bach/bs2.2 (:V70A:$BACH.PLAMLIB*)
index = 2 for /home/bs2.2 (:V70A:$BACH.CCC.*.C)
index = 1 for /home/bach/bs2.1 (:V70A:$BACH.ASS.*.S)
```

Beispiel 2

Auflisten der Information über ein bestimmtes bs2fs-Dateisystem

```
# get_container_index /home/bs2.2
index = 2 for /home/bs2.2
```

6.4 Behebung von Fehlerfällen

Wenn eine Datei in einem bs2fs-Dateisystem geändert wurde und nach Abschluss der Verarbeitung nicht wieder zurück ins BS2000 kopiert werden kann, dann wird die betreffende ufs-Datei innerhalb des bs2fs-Containers in ein spezielles bs2fs_lost+found-Verzeichnis verschoben. Existiert dort schon eine gleichnamige Datei aus einem früheren, fehlgeschlagenen Zurückschreibeversuch, dann wird diese ohne Rückfrage überschrieben.

Der Anwender kann sich mit dem Kommando *bs2fs_recover* seine solcherart verschobenen Dateien auflisten lassen und diese dann, z.B. unter einem modifizierten Namen, ins BS2000 kopieren und/oder im bs2fs-Container löschen.

Alternativ kann diese Aufgabe auch stellvertretend vom Systemverwalter übernommen werden.

Damit die in den bs2fs_lost+found-Bereich verschobenen Dateien nicht unbemerkt bleiben, wird beim Einhängen des bs2fs-Containers eine Meldung auf der Konsole ausgegeben, die über die Benutzerkennungen informiert, zu denen Dateien in diesem Bereich existieren. Der bs2fs_lost+found-Bereich wird beim Einhängen natürlich nicht wie der Rest des bs2fs-Containers gelöscht.

Mit dem Kommando *bs2fs_recover* kann auf einfache Weise geprüft werden, ob zu einer bestimmten Kennung Dateien im bs2fs_lost+found-Bereich vorliegen. Bei intensiver Nutzung des bs2fs-Dateisystems empfiehlt es sich, eine automatisierte Überprüfung mit diesem Kommando z.B. im Logon-Skript (*$HOME/.profile* oder */etc/profile*) einzurichten.

Um die Wirkungsweise des Kommandos und der damit zusammenhängenden Mechanismen kennenzulernen, können Sie Fehler beim Zurückschreiben der bs2fs-Dateien mit der POSIX-Datei */etc/.bs2fsdSimulateCloseError* simulieren. Existiert diese Datei (die keinen Inhalt benötigt), dann wird die ufs-Datei unabhängig vom Erfolgsstatus des Zurückschreibens ins BS2000 in den bs2fs_lost+found-Bereich verschoben. Wird die Datei */etc/.bs2fsdSimulateCloseError* entfernt oder umbenannt, dann ist die Simulation wieder aufgehoben.

6.5 Recovery für bs2fs_lost+found-Bereich durchführen

Dieses Kommando bietet die folgenden Funktionen:

- Auflisten der Dateien im bs2fs_lost+found-Bereich
- Ausgabe der Anzahl dieser Dateien
- Generieren eines Shell-Skripts, das die Dateien aus dem bs2fs_lost+found-Bereich ins BS2000 kopiert und/oder sie aus diesem Bereich löscht
- Direktes Kopieren und/oder Löschen von Dateien im bs2fs_lost+found-Bereich, entweder im interaktiven Modus mit Rückfrage vor jeder Aktion oder in einem automatischen Modus mit Voreinstellungen.

Syntax

Format 1:

```
bs2fs_recover [-l] [-m level]
              [-a after] [-b before] [-u user|*ALL] [dateiauswahl]
```

Format 2:

```
bs2fs_recover {-g|-x} [-d] [-w] [-c] [-f n|y] [-p prefix] [-s suufix]
              [-a after] [-b before] [-u user|*ALL] [dateiauswahl]
```

Hauptoptionen

Die Hauptoptionen legen fest, welche Funktion ausgeführt werden soll. Es darf nur eine Hauptoption angegeben werden.

Keine Hauptoption angegeben

Alle ausgewählten Dateien werden aufgelistet, das entspricht der Hauptoption *-l*.

`-l`

Die Anzahl oder eine Liste der ausgewählten Dateien wird auf die Standardausgabe ausgegeben. Die Option *-m* legt den Umfang der Ausgabe fest.

`-g`

Es wird ein Shell-Skript generiert, das bei seiner Ausführung die durch die Optionen *-d* und *-w* spezifizierten Aktionen durchführt. Das generierte Shell-Skript wird auf die Standardausgabe ausgegeben.

`-x`

Die mit den Optionen *-d* und *-w* spezifizierten Aktionen werden unmittelbar ausgeführt. Bei den jeweiligen Aktionen erfolgt eine Ausgabe standardmäßig nur im Fehlerfall. Möchte man den Fortschritt der Abarbeitung sehen, sollte die -v Option mindestens einmal angegeben werden, damit der jeweilige Dateiname vor der Ausführung der Aktionen für diese Datei aufgelistet wird.

Zusatzoptionen

Die Zusatzoptionen dienen zur Modifikation der jeweiligen Aktion bzw. zur Dateiauswahl.

-m level

Die Option steuert den Umfang der Ausgabe bei der Option *-l*. Wenn die Option *-m* nicht angegeben ist, wird *-m 1* angenommen.

Die Angabe von *level* ist obligatorisch. Folgende Angaben sind möglich:

0

Die Anzahl der gefundenen Dateien wird in einer Zeile der Form "*Anzahl* file(s)" auf die Standardausgabe ausgegeben. Zusätzlich liefert der Ende-Status in diesem Fall die Information, ob mindestens eine Datei gefunden wurde (Ende-Status = 0) oder nicht (Ende-Status = 1). Damit kann auf einfache Weise geprüft werden, ob sich eigene Dateien im bs2fs_lost+found-Bereich befinden.

1

Eine Liste der Dateien wird auf die Standardausgabe ausgegeben. Sie enthält Datum und Uhrzeit der letzten Modifikation, Dateigröße und den Namen der zugehörigen BS2000-Datei bzw. die vollständige Spezifikation des entsprechenden Bibliothekselements (*Bibliotheksname*(*Elementname*, *Typ*,*Version*)). Dem Namen wird die laufende Nummer des bs2fs-Mounts durch ein ‚/' abgetrennt vorangestellt.

2

Ausgabe wie bei *-m 1* mit folgenden zusätzlichen Informationen:

Dateiverarbeitungsmodus des bs2fs-Mounts T, TB, B (für ftyp text, textbin oder binary) und absoluter ufs-Dateiname im bs2fs-Container (Bereich bs2fs_lost+found).

3

Ausgabe wie bei *-m 2* mit folgenden zusätzlichen Informationen:

Zugriffsmethode (PAM, SAM, ISAM oder PLAM), Blockformat, Blocklänge, Satzformat und Satzlänge.

Existieren bestimmte Informationen wie z.B. Satzformat bei PAM-Dateien nicht, wird stattdessen ein ‚-' ausgegeben. Diese Informationen können Sie verwenden, um eigene Skripte für das Kopieren und Löschen der Dateien zu schreiben.

Beispiel für eine Ausgabe mit -m 3:

```
2012-03-20 09:39:06 8 3/:90GB:$TSOS.TSOSDAT T /grossercont
/bs2fs_lost+found/TSOS/:90GB:3:TD01SV00000:tsosdat SAM DATA (STD,1) V -
```

-d

In Verbindung mit den Hauptoptionen *-g* bzw. *-x:* Die ausgewählten Dateien werden aus dem bs2fs-Container-Bereich bs2fs_lost+found gelöscht. Diese Option kann auch gemeinsam mit der Option *-w* eingesetzt werden.

-w

In Verbindung mit den Hauptoptionen *-g* bzw. *-x:* Die ausgewählten Dateien werden ins BS2000 kopiert. Diese Option kann auch gemeinsam mit der Option *-d* eingesetzt werden.

-c

Unterdrückt die Verwendung der bisherigen Catid und Benutzerkennung beim Kopieren ins BS2000 (Option *-w*). Diese Option kann (gegebenenfalls zusammen mit der Option *-p*) dazu benutzt werden, die Sicherung unter einer anderen Catid/Benutzerkennung durchzuführen, als beim bs2fs-mount spezifiziert wurde.

-f n|y

Legt fest, ob bereits existierende BS2000-Dateien und LMS-Elemente beim Kopieren ins BS2000 (Option *-w*) überschrieben werden.

Abhängig von der Hauptoption hat die Option *-f* folgende Auswirkungen:

Hauptoption *-g* (Generieren eines Shell-Skripts):

Wenn die Option *-w* angegeben ist, wird im Skript generell die Variable *OV* gesetzt. Diese steuert während der Ausführung des generierten Skripts das Verhalten beim Kopieren ins BS2000.

Ohne Angabe der Option *-f* oder bei Angabe von *-f n* erhält die Variable *OV* den Wert "NR"; somit werden BS2000-Dateien beim Kopieren nicht überschrieben. Bei Angabe von *-f y* (*OV*="Y") werden sie überschrieben.

Bei gleichzeitiger Angabe der Optionen *-w* und *-d* werden Dateien nur gelöscht, wenn der vorangegangene Kopiervorgang erfolgreich war, entweder weil die Datei im BS2000 noch nicht existierte oder weil sie erfolgreich überschrieben werden konnte.

Wenn das Skript nur zum Löschen (Option *-d*) generiert wird, wird die Variable *OV* nicht gesetzt. Die Dateien im Lost+Found-Bereich werden dann unabhängig von der Option *-f* ohne Abfrage gelöscht.

Hauptoption *-x* (Unmittelbares Löschen oder Kopieren)

Mit dem Gebrauch der Option *-f* wird entschieden, ob vor jeder Aktion eine Rückfrage erfolgen soll (interaktiver Modus) oder nicht (automatisierter Modus).

Ohne Angabe der Option *-f* werden folgende Abfragen durchgeführt:

1. Soll die Datei kopiert werden (bei Option *-w*)
2. Soll die Datei überschrieben werden (bei Option *-w*, wenn die Ziel-Datei bereits existiert)
3. Soll die Datei aus dem Lost+Found-Bereich gelöscht werden (bei Option *-d*).

Falls eine dieser Abfragen mit "nein" beantwortet wird, finden keine weiteren Abfragen und Aktionen für dieselbe Datei statt. Insbesondere wird eine Datei nicht gelöscht, wenn zuvor das Kopieren oder Überschreiben abgelehnt wurde.

Bei Angabe der Option *-f y* oder *-f n* finden weder beim Kopieren (*-w*) noch beim Löschen (*-d*) Abfragen statt. Beim Kopieren werden bereits existierende Dateien abhängig von der Option *-f* überschrieben (*-f y*) oder nicht (*-f n*).

-p prefix

Gibt eine Zeichenkette an, die dem BS2000-Dateinamen beim Kopieren ins BS2000 (Option *-w*) vorangestellt wird. Enthält *prefix* eine Catid und/oder eine Benutzerkennung, dann muss gleichzeitig die Option *-c* angegeben werden, damit ein gültiger BS2000-Dateiname gebildet werden kann.

-s suffix

Gibt eine Zeichenkette an, die beim Kopieren ins BS2000 (Option *-w*) an den BS2000-Dateinamen angehängt wird.

-a `after`

Die Dateiauswahl wird auf Dateien beschränkt, deren letzte Modifikation nach dem mit *after* spezifizierten Zeitpunkt erfolgte.

Die Zeit wird wie folgt angegeben: `[[YY]yy]MMDDhhmm[.SS]`

`[YY]yy`

Jahr (zwei- oder vierstellig)
Eine zweistellige Jahresangabe wird folgendermaßen ergänzt: *yy* > 68 ergibt 19*yy, yy* <= 68 ergibt 20*yy.*
Ist kein Jahr angegeben, dann wird das aktuelle Jahr angenommen.

`MM`

Monat zweistellig (01 bis 12)

`DD`

Tag zweistellig (01 bis 31)

`hh`

Stunde zweistellig (00 bis 23)

`mm`

Minute zweistellig (00 bis 59)

`SS`

Sekunde zweistellig (00 bis 61)
Die Werte 60 und 61 sind für Schaltsekunden vorgesehen.
Ist Sekunde nicht angegeben, wird 0 angenommen.

-b `before`

Die Dateiauswahl wird auf Dateien beschränkt, deren letzte Modifikation vor dem mit *before* spezifizierten Zeitpunkt erfolgte. Das Format der Zeitangabe ist bei der Option *-a* beschrieben.

-u `user`

Die Dateiauswahl wird auf die Dateien der angegebenen Benutzerkennung beschränkt. Die Benutzerkennung kann beliebig in Groß- und Kleinbuchstaben angegeben werden. In der Regel sind aufgrund der Standardattribute der Unterverzeichnisse im Lost+Found-Bereich (`drwx --- ---`) für nicht privilegierte Kennungen nur die eigenen Dateien sicht- und bearbeitbar.

***-u** user* nicht angegeben:

Die BS2000-Benutzerkennung des Aufrufers wird verwendet.

-u *ALL (beliebige Groß-/Kleinschreibung)

Es werden alle BS2000-Kennungen bearbeitet, für die sich Dateien im Lost+Found-Bereich befinden. Diese Angabe ist in der Regel nur für die Systemverwaltung (Kennung TSOS) relevant, da nur diese die notwendigen Rechte zum Kopieren und Löschen von Dateien in fremden Kennungen besitzt.

-v

Es werden zusätzliche Informationen über die Aktionen des Tools ausgegeben. Diese Option kann auch mehrfach (bis zu drei Mal) angegeben werden, um den Detaillierungsgrad der Informationen zu erhöhen.

`dateiauswahl`

Mit *dateiauswahl* wird die Auswahl der im Lost+Found-Verzeichnis befindlichen Dateien gesteuert (zusätzlich zur Auswahl mit der Option *-u*).

Es können Muster-Ausdrücke verwendet werden, wie sie auch die Shell für die Erzeugung von Dateinamen anbietet. Diese Ausdrücke werden dann auf die BS2000-Dateinamen und LMS-Element-Spezifikationen angewendet, wie sie das Tool bei Nutzung der Option *-l* ausgibt.

Beispiel:

Von folgender Ausgabe mit *-l -m1* ist der fettgedruckte Dateiname für den Musterausdruck relevant.

```
2012-03-20 09:39:06 8 3/:90GB:$TSOS.TSOSDAT
```

Der komplette Dateiname (ohne Musterausdruck) müsste in diesem Fall auch das Präfix 3/ enthalten.

dateiauswahl nicht angegeben:

Alle vorhandenen Dateien des aufrufenden Benutzers.

6.6 Maßnahmen zur Leistungsverbesserung

Prüfen Sie, ob bei der Auswahl der BS2000-Dateien, die in einem bs2fs-Dateisystem bereitgestellt werden sollen, folgende Punkte beachtet wurden:

- Die Musterangabe in der Option *ressource* sollte so gewählt werden, dass möglichst nur die BS2000-Dateien eingehängt werden, die tatsächlich benötigt werden, da die Anzahl der eingehängten Dateien die Performance beeinflusst.
- Ist es syntaktisch nicht oder nur sehr schwer möglich, die benötigten von den nicht benötigten Dateien abzugrenzen, sollte vor dem ersten Montieren eine der folgenden Umgehungsmöglichkeiten in Betracht gezogen werden: Auslagerung der benötigten Dateien auf ein anderes Pubset (unter derselben oder einer anderen Benutzerkennung) oder eine sinnvolle Umbenennung dieser BS2000-Dateien.
- Es sollte vermieden werden, dass dieselben BS2000-Dateien mehrmals an verschiedenen Stellen eingehängt werden. Dies tritt ein, wenn mehrere *mount*-Vorgän-ge mit identischen oder überschneidenden *ressource*-Angaben ausgeführt werden. Wenn eine solche Datei in einem der beteiligten bs2fs-Dateisysteme für schreibende Zugriffe geöffnet ist, kann sie nicht gleichzeitig in einem anderen bs2fs-Dateisystem geöffnet werden.
- Starten Sie ggf. zusätzliche Kopierdämonen mit dem Kommando *start_bs2fsd*.

7 Fachwörter

bs2fs-Container

Ein Dateisystem vom Typ ufs, das einzig dazu dient, temporär Dateien von bs2fs-Dateisystemen aufzunehmen. Diese Dateien werden dann in Verzeichnissen (pro eingehängtem bs2fs-Dateisystem ein Verzeichnis) im bs2fs-Container abgelegt.

bs2fs-Dateisystem

Auswählbare Menge von Dateien im BS2000, die im POSIX als Dateisystem zur Verfügung gestellt werden, so dass auf sie mit POSIX-Mitteln (Kommandos, Programmschnittstellen) zugegriffen werden kann. Die Auswahl der Dateien erfolgt über Benutzer- und Katalogkennung sowie Wildcard-Symbole.

Dämon

Dämonen sind Systemprozesse, die permanent und meist im Hintergrund laufen und die allgemeine Aufgaben durchführen.

Dateibaum

Gesamte Hierarchie der Dateien auf einem UNIX-Rechner bzw. in POSIX; sie wird Dateibaum genannt, da sie gemäß einer Baumstruktur geordnet ist. Die Wurzel des Dateibaums ist das Root-Dateiverzeichnis (/). Alle anderen Dateiverzeichnisse sind Zweige, die vom Root-Dateiverzeichnis ausgehen. Die Dateien sind die Blätter des Baumes. Jede Datei ist über genau einen Pfad des Dateibaums erreichbar.

Dateisystem

Ein Dateisystem ist eine hierarchische Sammlung von Dateiverzeichnissen und Dateien, die sich physikalisch auf demselben Speichermedium befinden, z.B. auf einer Partition oder in einer Behälterdatei.

Der Begriff wird für Organisationsformen von Dateien benutzt: wie z.B. UNIX-Dateisystem, POSIX-Dateisystem, hierarchisches Dateisystem, andere BS2000-Dateisysteme (z.B. DMS, Data Management System).

Dateisystem-Typ

Art eines Dateisystems im Dateibaum von POSIX oder eines UNIX-Rechners. Die bekanntesten Typen sind:

Typ *bs2fs*: BS2000-Dateien, die unter POSIX zur Verfügung gestellt werden.
Typ *ufs*: lokale Dateisysteme mit Benutzerdaten.
Typ *nfs*: Dateisysteme, die sich physikalisch auf fernen Rechnern befinden.

Weitere Typen sind z.B *fdfs*, *proc*, *bfs*, *rfs* oder *s5*.

Alle in einem System verwendeten Typen von Dateisystemen müssen bei der Konfiguration des Systems bekannt sein, nachträglich kann kein neuer Dateisystemtyp hinzugefügt werden. Information über die in einem System verwendbaren Dateisystemtypen liefert die Schnittstelle sysfs().

Dateiverzeichnis

Ein Dateiverzeichnis wird verwendet, um Dateien und untergeordnete Dateiverzeichnisse eines hierarchischen Dateisystems zu gruppieren und zu organisieren.

einhängen

Mit dem Kommando *mount* werden lokale Ressourcen in einem lokalen Dateisystem zugänglich gemacht; wird auch montieren genannt.

Kopierdämon

Systemprozess, der das Kopieren vom BS2000 in den bs2fs-Container und zurück durchführt.

POSIX-Dateisystem

Dateisystem auf einem BS2000-Rechner mit POSIX. Das POSIX-Dateisystem entspricht einem UNIX-Dateisystem.

Root-Dateiverzeichnis

Das Dateisystem, an dem der Dateibaum beginnt. Es wird auch Wurzel-Dateiverzeichnis genannt und durch den Schrägstrich (/) dargestellt.

transparenter Dateizugriff

Der Benutzer kann auf bs2fs-Dateien wie auf ufs-Dateien (mit den beschriebenen wenigen Einschränkungen) zugreifen, mit den gleichen Zugriffsfunktionen oder Kommandos.

ufs-Dateisystem

Das ufs-Dateisystem ist das lokale Standard-Dateisystem in POSIX. Es ist realisiert durch eine BS2000-Datei (vom Typ PAM), in der die Dateien des Dateisystems wie auf einer UNIX-Platte abgelegt sind.

8 Literatur

Die Handbücher finden Sie im Internet unter http://manuals.ts.fujitsu.com. Handbücher, die auch in gedruckter Form vorliegen, können Sie unter http://manualshop.ts.fujitsu.com bestellen.

[1] **POSIX**
Grundlagen für Anwender und Systemverwalter
Benutzerhandbuch

[2] **POSIX**
Kommandos
Benutzerhandbuch

[3] **C/C++**
C-Bibliotheksfunktionen für POSIX-Anwendungen
Referenzhandbuch

[4] **NFS**
NFS-Benutzer und NFS-Verwalter
Benutzerhandbuch

[5] **BS2000**
Einführung in die Systembetreuung
Benutzerhandbuch

[6] **BS2000**
Kommandos
Benutzerhandbuch

[7] **EDT**
Anweisungen
Benutzerhandbuch

[8] **EDT**
Unicode-Modus Anweisungen
Benutzerhandbuch