Deutsch

BS2000/OSD

POSIX

Grundlagen für Anwender und Systemverwalter

Benutzerhandbuch

Stand der Beschreibung:
BS2000/OSD V7.0/V8.0/V9.0

Ausgabe November 2012

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 manuals@ts.fujitsu.com senden.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2008

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 9001:2008 erfüllt.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright und Handelsmarken

Inhalt

1 Einleitung

Unter POSIX (**P**ortable **O**pen **S**ystem **I**nterface for UNI**X**) versteht man eine Reihe von Standards auf UNIX-Basis. Diese Standards gewährleisten die Kompatibilität und Interoperabilität von Anwendungen in einem heterogenen Netz. Ein heterogenes Netz besteht aus Rechnern von verschiedenen Herstellern sowie aus System- und Anwendersoftware von verschiedenen Softwareanbietern.

Der POSIX-Standard wurde vom Institute of Electrical and Electronics Engineers (IEEE) 1989 als nationaler amerikanischer Standard definiert. Anschließend wurde er vom X/OPEN-Konsortium erweitert und 1990 als internationaler Standard verabschiedet (X/OPEN Portability Guide IV).

Der X/OPEN Portability Guide IV, kurz auch XPG4-Standard genannt, umfasst 7 Bände, die u.a. Schnittstellendefinitionen zu Basisbetriebssystemen, Programmiersprachen, Datenverwaltung und Vernetzung enthalten. Das Betriebssystem BS2000/OSD unterstützt ab V2.0 die XPG4-Standards, die in den ersten beiden Bänden enthalten sind:

- Band 1: System Interfaces and Headers (ca. 350 Programmschnittstellen)
- Band 2: Commands and Utilities (ca. 200 Benutzerschnittstellen)

Zur Unterstützung dieser Schnittstellen wurde die POSIX-Funktionalität im BS2000/OSD integriert. POSIX bezeichnet also sowohl den Standard vom IEEE als auch die BS2000/OSD-Funktionalität „POSIX“. Mit POSIX wurden die Voraussetzungen für eine erfolgreiche Zertifizierung nach dem XPG4-Standard geschaffen, die in zwei Stufen erfolgte: Ende 1995 erhielt BS2000/OSD von „The Open Group“ (vormals X/OPEN) das „XPG4 Base Branding“ (XPG4) und Mitte 1997 das Branding nach „XPG4 UNIX profile“ (auch XPG4.2 oder UNIX95 genannt). Außerdem wurde BS2000/OSD mit seinem POSIX-Subsystem 1999 von „The Open Group“ als Internet Server zertifiziert.

Der Kern der POSIX-Funktionalität ist als privilegiertes BS2000-Subsystem realisiert. Dem Benutzer stehen die Bibliotheksfunktionen des XPG4-Standards über eine C-Bibliothek und eine definierte Menge von Kommandos über eine Shell (POSIX-Shell) zur Verfügung. Die C-Bibliothek ist Bestandteil des Produkts CRTE (Common Runtime Environment).

Mit POSIX lassen sich Anwendungsprogramme leicht portieren - unabhängig vom ausführenden Betriebssystem. Deshalb können XPG4-konforme Programme nach einer Neuübersetzung auch im BS2000/OSD ablaufen.

Die POSIX-Programmschnittstellen werden parallel zu den BS2000-Programmschnittstellen angeboten. Die gemischte Nutzung von BS2000- und POSIX-Programmschnittstellen in einem Programm ist mit Einschränkungen möglich.

1.1 Konzept der POSIX-Dokumentation

Für das Kennenlernen von POSIX und das Arbeiten mit dem Subsystem POSIX und der POSIX-Shell im BS2000/OSD steht Ihnen folgende Dokumentation zur Verfügung:

- Eine Einführung in das Arbeiten mit dem Subsystem POSIX erhalten Sie im vorliegenden Handbuch „POSIX - Grundlagen für Anwender und Systemverwalter“. Darüber hinaus werden die Verwaltungsaufgaben beschrieben, die im Zusammenhang mit dem Subsystem POSIX anfallen. Außerdem erfahren Sie, mit welchen BS2000-Softwareprodukten Sie das Subsystem POSIX nutzen können.
- Eine vollständige Beschreibung der POSIX-Kommandos, mit denen Sie in der POSIX-Shell arbeiten können, enthält das POSIX-Handbuch „Kommandos“ [1].
- Alle notwendigen Informationen zum Einsatz von bs2fs-Dateisystemen finden Sie im POSIX-Handbuch „BS2000-Dateisystem bs2fs“ [2].

POSIX-Dokumentation im BS2000/OSD-Umfeld

Im BS2000/OSD werden Softwareprodukte funktionell erweitert, so dass Sie auch mit diesen Produkten die POSIX-Funktionalität nutzen können.

Eine Reihe von Dienstprogrammen ermöglichen den Zugriff auf das POSIX-Dateisystem. So können Sie z.B. mit dem EDT Dateien des POSIX-Dateisystems bearbeiten.

Durch die Erweiterung des CRTE gemäß dem XPG4-Standard können Sie mit den C-Bibliotheksfunktionen unabhängig vom ausführenden Betriebssystem portable C-Programme schreiben.

Als Grundlage für den Zugriff auf die POSIX-Funktionalität aus anderen Softwareprodukten wird das vorliegende Handbuch „POSIX - Grundlagen für Anwender und Systemverwalter“ vorausgesetzt.

1.2 Zielsetzung und Zielgruppen des Handbuchs

Dieses Handbuch wendet sich an:

- DV-Organisatoren, die einen Überblick über POSIX gewinnen möchten.
- Nichtprivilegierte BS2000-Benutzer, die mit POSIX arbeiten wollen. Grundlegende Kenntnisse von UNIX-Betriebssystemen sind von Vorteil.
- Benutzer von Workstations, die bisher überwiegend mit UNIX-Systemen gearbeitet haben und nun POSIX nutzen möchten. BS2000-Grundkenntnisse sind erforderlich.
- BS2000-Systemverwalter und POSIX-Verwalter. Gute Kenntnisse des Betriebssystems BS2000/OSD und von UNIX-Systemen sind erforderlich.

Grundbegriffe von UNIX-Betriebssystemen sind in diesem Handbuch für BS2000-Benutzer mit aufgenommen.

1.3 Konzept des Handbuchs

Dieses Handbuch enthält Kapitel, die für alle Benutzer wichtig sind, und Kapitel, die nur für BS2000-Systemverwalter, POSIX-Verwalter oder BS2000-Gruppenverwalter von Bedeutung sind.

Die Kapitel 1 bis 4 wenden sich an alle Benutzer:

- Einleitung
- Einführung in POSIX
- Arbeiten mit POSIX
- BS2000-Softwareprodukte im Umfeld von POSIX

Die Kapitel 5 bis 8 wenden sich an BS2000-Systemverwalter, POSIX-Verwalter und BS2000-Gruppenverwalter:

- POSIX installieren
- POSIX-Subsystem und POSIX-Lader
- Dateisysteme verwalten und überwachen
- POSIX-Benutzer verwalten

Im Referenzteil dieses Handbuchs (Kapitel 9 und 10) finden Sie:

- BS2000-Kommandos für POSIX
- Privilegien bei POSIX
- Kommandoumfang der POSIX-Shell
- Dämonen von POSIX
- Dateiverzeichnisse, die bei einer Erstinstallation angelegt werden
- Gerätedateien, die bei einer Erstinstallation angelegt werden
- Verwaltungsdateien, die bei einer Erstinstallation angelegt werden
- Tuning-Maßnahmen
- POSIX-Protokolldateien
- Lokale Uhrzeit in POSIX

Im Anschluss an den Referenzteil finden Sie verschiedene Verzeichnisse, die Ihnen das Arbeiten mit diesem Handbuch erleichtern.

Readme-Datei

Funktionelle Änderungen der aktuellen Produktversion und Nachträge zu diesem Handbuch entnehmen Sie bitte ggf. der produktspezifischen Readme-Datei.

Readme-Dateien stehen Ihnen online bei dem jeweiligen Produkt zusätzlich zu den Produkthandbüchern unter *http://manuals.ts.fujitsu.com* zur Verfügung. Alternativ finden Sie Readme-Dateien auch auf der Softbook-DVD.

Informationen unter BS2000/OSD

Wenn für eine Produktversion eine Readme-Datei existiert, finden Sie im BS2000-System die folgende Datei:

`SYSRME.<product>.<version>.<lang>`

Diese Datei enthält eine kurze Information zur Readme-Datei in deutscher oder englischer Sprache (<lang>=D/E). Die Information können Sie am Bildschirm mit dem Kommando `/SHOW-FILE` oder mit einem Editor ansehen.
Das Kommando `/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=<product>` zeigt, unter welcher Benutzerkennung die Dateien des Produkts abgelegt sind.

Ergänzende Produkt-Informationen

Aktuelle Informationen, Versions-, Hardware-Abhängigkeiten und Hinweise für Installation und Einsatz einer Produktversion enthält die zugehörige Freigabemitteilung. Solche Freigabemitteilungen finden Sie online unter *http://manuals.ts.fujitsu.com*.

1.4 Änderungen gegenüber dem Vorgänger-Handbuch

Die Ausgabe dieses Handbuches enthält gegenüber der letzten Ausgabe (Bestellnummer: U22795-J-Z125-6) folgende Änderungen:

Das neue Shell-Kommando *edtu* ermöglicht es, den EDT im Unicode-Modus aufzurufen.

Die Konsistenzprüfung bestimmter oder aller Dateisysteme kann auch beim Starten des Subsystems POSIX erzwungen werden, siehe Abschnitt „POSIX-Subsystem starten“ auf Seite 154.

Ab BS2000/OSD V9.0 kann einzelnen oder allen Benutzerkennungen eine Standard-Jobklasse für POSIX-Tasks zugewiesen werden, siehe Abschnitt „POSIX-Standard-Jobklassen“ auf Seite 194.

Die Syntax des Kommandos ADD-POSIX-USER (siehe Seite 195) wurde erweitert.

Der syslog-Dämon ermöglicht die Protokollierung von Systemnachrichten in eine oder mehrere Protokolldateien, siehe Abschnitt „POSIX-Protokolldateien“ auf Seite 282.

Die Beschreibung der Installation wurde neu strukturiert (siehe Seite 115).

Die Mechanismen zur Ermittlung der lokalen Uhrzeit werden detailliert beschrieben (siehe Seite 292).

1.5 Darstellungsmittel

In den Kommando-/Anweisungsformaten werden bestimmte Zeichen und Darstellungsformen verwendet. Ihre Bedeutung finden Sie im Handbuch „Kommandos“ zu BS2000/OSD [28].

Folgende Konventionen gelten für die Darstellungsmittel im Text:

- Im Fließtext ist nicht zwischen Konstanten und Variablen unterschieden. Alle Elemente der Syntax, Teile aus Datenstrukturen sowie Dateinamen, Pfadnamen und Kommandos sind dort in *kursiver* Schrift dargestellt.
- In Anwendungsbeispielen sind Eingaben in das System in Schreibmaschinenschrift dargestellt. Alle Eingabezeilen werden bei Zeichenterminals mit der Taste ↵ abgeschlossen, bei Blockterminals mit EM DUE. Deshalb sind die Tasten am Ende der Eingabezeilen weggelassen.
  Manche Eingaben sind terminalabhängig, d.h. sie unterscheiden sich bei Block- und Zeichenterminals (siehe dazu auch den Abschnitt „Unterstützung von Terminals“ auf Seite 30).

  Ausgaben des Betriebssystems sind in Schreibmaschinenschrift dargestellt.
- Literaturhinweise sind im Text durch Kurztitel angegeben, die in Anführungszeichen stehen. Die vollständigen Titel finden Sie zusammen mit einer Kurzbeschreibung im Literaturverzeichnis.
- Verweise innerhalb dieses Handbuchs geben die betreffende Seite im Handbuch an und je nach Bedarf auch den Abschnitt bzw. das Kapitel. Verweise auf Themen, die in einem anderen Handbuch beschrieben sind, enthalten nur den Kurztitel dieses Handbuchs. Über das Stichwortverzeichnis können Sie in dem genannten Handbuch dann die entsprechende Stelle im Text finden.

Dieses Symbol steht vor Warnungen, die Sie im Interesse der System- und Betriebssicherheit unbedingt beachten müssen.

Dieses Symbol kennzeichnet wichtige Hinweise, die Sie unbedingt beachten sollten.

2 Einführung in POSIX

Dieses Kapitel wendet sich an alle Leser, die sich einen Überblick über POSIX verschaffen wollen. Es informiert Sie über die Rolle von POSIX im BS2000/OSD, das POSIX-Dateisystem, das Subsystem POSIX und über das Sicherheitskonzept von POSIX.

2.1 POSIX im BS2000/OSD

Mit POSIX setzt BS2000/OSD seine Strategie der Öffnung im Sinne der „Open Systems Direction“ konsequent fort.

Dieser Abschnitt gibt Ihnen einen allgemeinen Überblick über

- offene Systeme
- die Vorteile des POSIX-Standards
- die Bestandteile von POSIX
- die Hardware-Voraussetzungen für POSIX
- die Unterstützung von Terminals
- die BS2000-Softwareprodukte, die an POSIX angepasst wurden

Neue Anforderungen an die Informationstechnologie

Viele Jahrzehnte waren die meisten Unternehmen stark hierarchisch strukturiert. In den letzten Jahren wurden flache Organisationsstrukturen eingeführt. „Lean Management“ sorgt für kürzere Wege und schnellere, flexiblere Entscheidungen.

Diese Entwicklung hat auch neue Anforderungen an die Informationstechnologie gestellt. Heute gehören leistungsfähige, kostengünstige PCs und Workstations zur Grundausstattung eines Arbeitsplatzes. Der zunehmende Wunsch der Anwender nach übergreifenden Lösungen erfordert es, dass diese PCs und Workstations mit den vorhandenen Hostsystemen zu einem optimalen Gesamtsystem kombiniert werden. Alle Systeme müssen miteinander kommunizieren können, um die gemeinsamen Ressourcen zu nutzen.

Die Vernetzung heterogener Systeme ist also eine Grundforderung an die Informationstechnologie. Die Wirtschaftlichkeit und Anwenderfreundlichkeit von PCs und Workstations müssen mit der hohen Rechnerleistung, Speicherkapazität, Verfügbarkeit, Datenkonsistenz und Sicherheit von Hostsystemen verbunden werden.

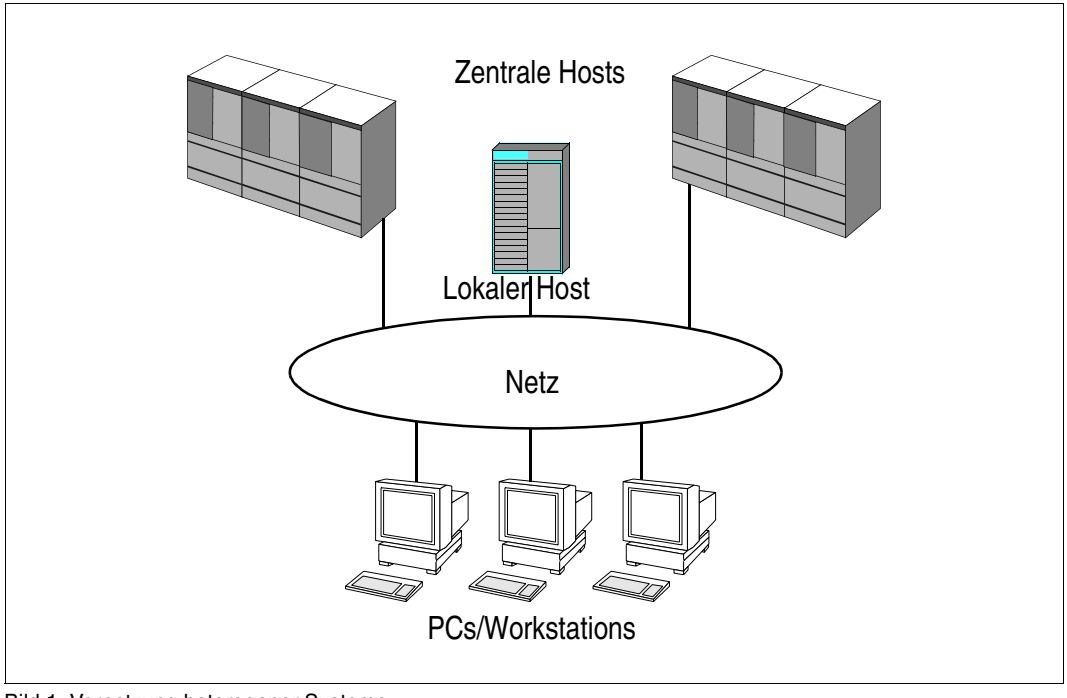

Bild 1: Vernetzung heterogener Systeme

Die Kommunikation zwischen heterogenen Systemen und die bessere Nutzung dieser Systeme ist nur möglich, wenn Standards für Betriebssystem-Schnittstellen festgelegt und eingehalten werden.

2.1.1 Eine Welt offener Systeme

Im Auftrag des Institute of Electrical and Electronic Engineers (IEEE) wurden umfangreiche Standards für portierbare Betriebssystem-Schnittstellen entwickelt. Diese Standards wurden unter dem Begriff „POSIX“ zusammengefasst. Durch POSIX werden proprietäre Systeme zu offenen Systemen. In offenen Systemen können Anwendungen über Systemgrenzen hinweg übertragen werden (Portabilität) und mit anderen Anwendungen zusammen- arbeiten (Interoperabilität).

2.1.1.1 Offenes BS2000 - BS2000/OSD

Die Offenheit des Produkt- und Systemangebots als Schwerpunkt der strategischen Geschäftsausrichtung wurde in der „Open Systems Direction“ (OSD) festgelegt. Deshalb wird auch das Betriebssystem BS2000 verstärkt auf eine offene Systemwelt ausgerichtet. Diese Ausrichtung macht auch der neue Name „BS2000/OSD“ sichtbar.

BS2000/OSD ist eine kompatible Erweiterung des Betriebssystems BS2000. Deshalb laufen bestehende BS2000-Anwendungen in BS2000/OSD wie bisher ab; alle Dienste des BS2000 sind nach wie vor verfügbar.

BS2000/OSD stellt neue Kommandos zur Verfügung, so dass BS2000-Dateien für den Zugriff von POSIX-Anwendungen kopiert werden können. Benutzer (oder auch Programme, die fertige Prozeduren oder Shell-Skripts aufrufen) können BS2000-Dateien in ein hierarchisches POSIX-Dateisystem kopieren.

Viele Standard-Softwarepakete sind bereits heute auf BS2000/OSD verfügbar. So das relationale Datenbanksystem ORACLE und die betriebswirtschaftliche Anwendung R/2. Bei Bedarf lassen sich weitere Softwarepakete aus der offenen Welt kostengünstig portieren.

2.1.1.2 Offenheit durch Client-Server-Architekturen

In Client-Server-Architekturen werden verschiedene Rechnerwelten systemübergreifend integriert. Dezentrale, intelligente Rechner werden mit zentralen Mainframes verbunden. Dadurch ist eine verteilte Verarbeitung möglich.

Innerhalb dieses homogenen Ganzen stellen die Server verschiedene Dienste zur Verfügung, die von den Clients genutzt werden. Server-Funktionen werden überwiegend von Mainframe- und UNIX-Systemen ausgeübt. Client-Funktionen haben vor allem PCs und Workstations, aber auch UNIX-Systeme. Client- und Server-Systeme können beliebig kombiniert werden.

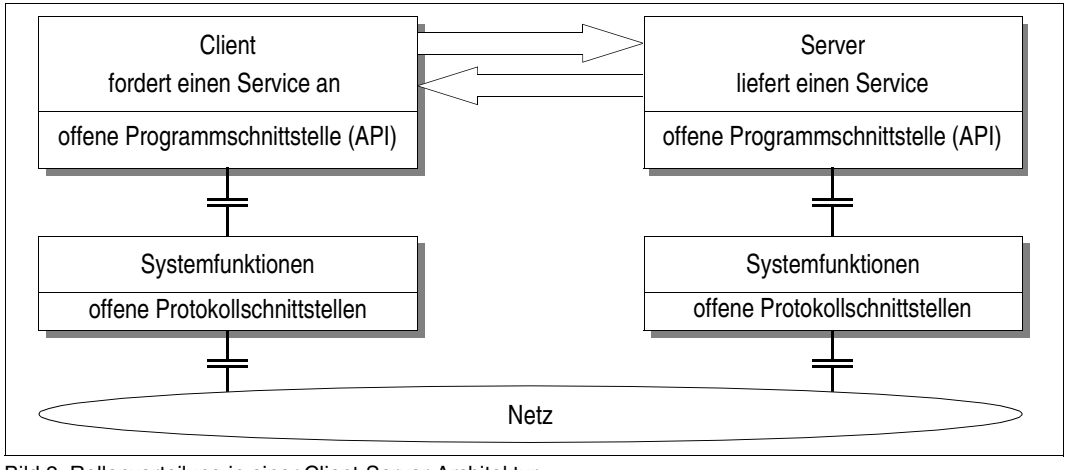

Bild 2: Rollenverteilung in einer Client-Server-Architektur

In einer flexiblen Architektur kann derselbe Rechner für bestimmte Dienste als Client und für andere Dienste als Server eingesetzt werden. So lassen sich die Stärken der verschiedenen Rechner optimal nutzen:

- Auf PCs laufen bevorzugt Standardanwendungen unter MS Windows ab, besonders im Bereich der Textverarbeitung, Tabellenkalkulation und der Geschäftsgrafik.
- Bei den Workstations stehen Anwendungen im Vordergrund, die eine hohe Grafikleistung erbringen, wie z.B. CAD (Computer Aided Design).
- BS2000-Mainframes eignen sich besonders als unternehmensweite Server durch ihre sehr große Rechnerleistung, den Einsatz von großen Massenspeichern, die hohe Sicherheit und den hohen Automatisierungsgrad der Administration.

BS2000-Server arbeiten mit anderen Servern im Netz zusammen, unabhängig davon, ob es sich um ein BS2000- oder UNIX-System handelt. So können zum Beispiel abteilungsinterne Daten von verschiedenen UNIX-Systemen verwaltet werden, während für übergreifende Daten ein zentraler BS2000-Server mit Hochleistungsperipherie zuständig ist.

Für den Anwender haben Client-Server-Architekturen mehrere Vorteile:

- Die Flexibilität in der Ablauforganisation wird erhöht.
- Informationen sind leicht und überall verfügbar.
- Unter den verschiedenen Systemen wird eine optimale Lastverteilung erreicht.
- Das Rechnernetz kann der jeweiligen Betriebsgröße genau angepasst werden.

2.1.1.3 BS2000/OSD bringt die UNIX-Systeme und die BS2000-Welt zusammen

Workstation-Benutzer können über die POSIX-Schnittstelle die Ressourcen und die Leistungsstärke des BS2000 nutzen, ohne die BS2000-proprietären Schnittstellen kennen zu müssen. Im Verbund mit BS2000/OSD wird der Plattenspeicher einer Workstation um den Host-Plattenspeicher erweitert, damit auch der Workstation-Benutzer große Datenmengen verarbeiten kann. Beispielsweise kann ein Entwickler seine Anwendungen auf einer Workstation entwickeln und anschließend seine Programme im BS2000 übersetzen, testen, korrigieren und ablaufen lassen.

Die UNIX-Plattformen und die BS2000-Welt können unabhängig voneinander existieren, wobei sie sich aber dieselben Prozess- und Speicherressourcen teilen. Die Benutzer können sich das Beste aus beiden Welten aussuchen: Die Standardschnittstelle und Portabilität der UNIX-Systeme und eine Vielzahl von Diensten des BS2000-Teils.

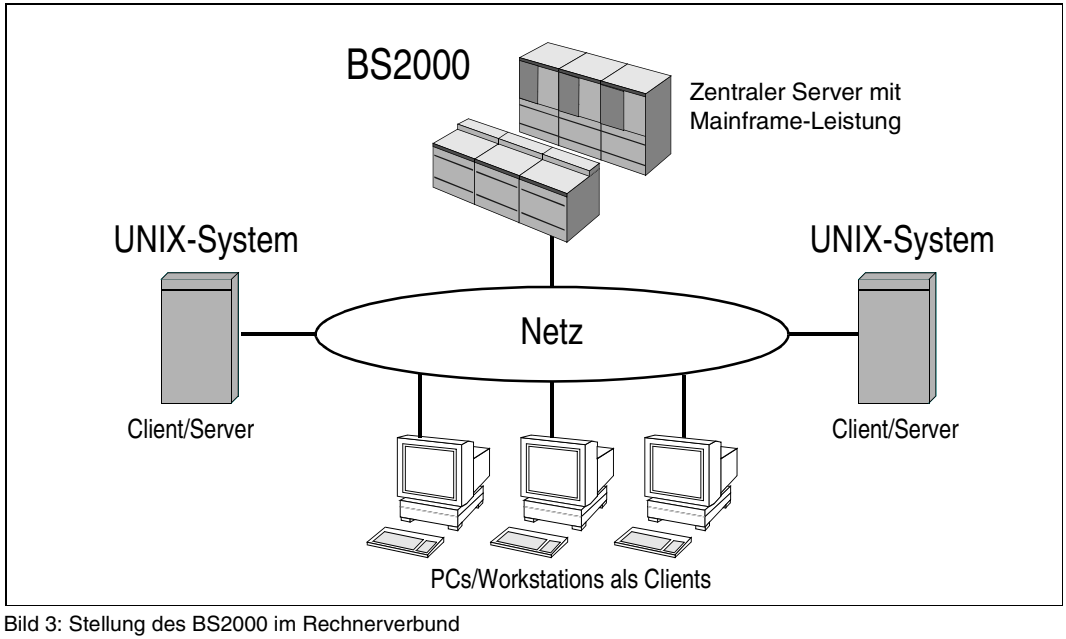

Bild 3: Stellung des BS2000 im Rechnerverbund

2.1.2 Vorteile des POSIX-Standards

Der POSIX-Standard bietet Ihnen folgende Vorteile:

- Portabilität von Anwendungsprogrammen
- Interoperabilität von Anwendungsprogrammen
- Arbeiten mit hierarchischen Dateisystemen
- BS2000/OSD als Server
- Verteilte Datenhaltung
- Verteilte Verarbeitung
- Gemeinsame Entwicklungstools

Im folgenden sind diese Vorteile näher erläutert.

Portabilität von Anwendungsprogrammen

Anwendungsprogramme, die gemäß den POSIX-Schnittstellen geschrieben sind, können auf allen XPG4-konformen Betriebssystemen und Hardware-Plattformen ablaufen. Portable Anwendungsprogramme können im BS2000 ebenso problemlos ablaufen wie zum Beispiel auf einer UNIX-Plattform.

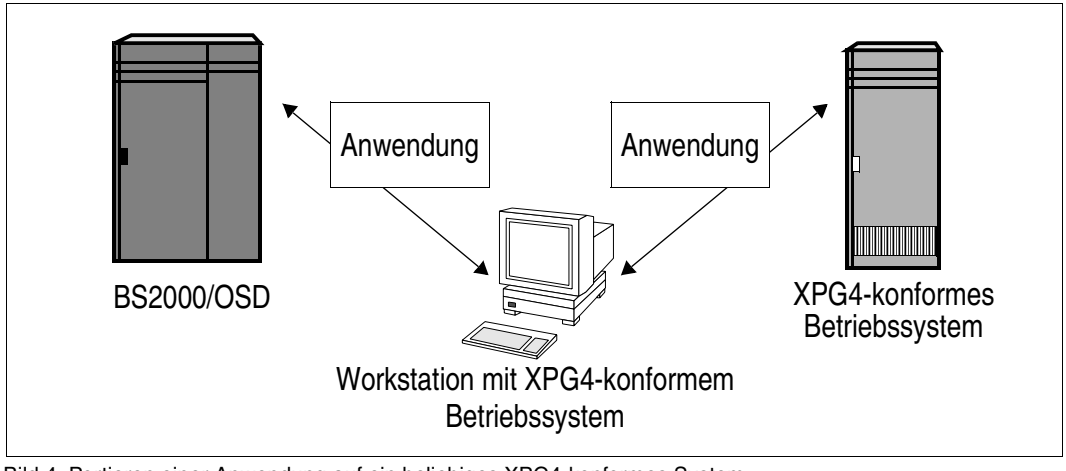

Bild 4: Portieren einer Anwendung auf ein beliebiges XPG4-konformes System

Daten und Dateien von Anwendungsprogrammen im ASCII-Code müssen vor dem Einsatz in POSIX in EBCDIC konvertiert werden (siehe Abschnitt „Kopieren und Konvertieren von Dateien“ auf Seite 37).

Interoperabilität von Anwendungsprogrammen

Anwendungsprogramme, die unter verschiedenen XPG4-konformen Betriebssystemen ablaufen, können Daten untereinander austauschen, falls die Dateiformate übereinstimmen (siehe Abschnitt „Kopieren und Konvertieren von Dateien“ auf Seite 37).

Bereitstellung hierarchischer Dateisysteme

Mit dem POSIX-Dateisystem wird das BS2000 um ein hierarchisch strukturiertes Dateisystem erweitert. Ein POSIX-Dateisystem ist eine Behälterdatei (Container) im BS2000 mit der Struktur eines UNIX-Dateisystems (UFS). Das POSIX-Dateisystem besteht aus Dateien (POSIX-Dateien) und Dateiverzeichnissen (näheres siehe Abschnitt „POSIX-Dateisystem“ auf Seite 33). POSIX-Benutzer können POSIX-Dateien erzeugen und bearbeiten.

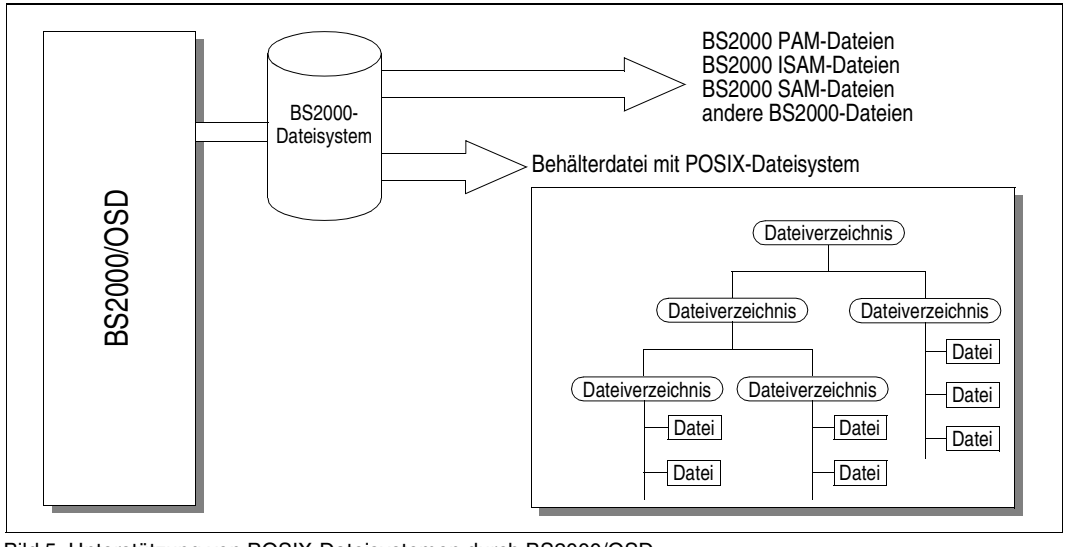

Bild 5: Unterstützung von POSIX-Dateisystemen durch BS2000/OSD

Mit dem Softwareprodukt NFS (**N**etwork **F**ile **S**ystem) lassen sich lokale POSIX-Dateisysteme in ferne Rechner einhängen und ferne UFS-Dateisysteme in ein lokales POSIX-Dateisystem.

Den unmittelbaren Zugriff auf BS2000-Dateien aus POSIX ermöglichen bs2fs-Dateisysteme (siehe Handbuch „POSIX (BS2000/OSD) – BS2000-Dateisystem bs2fs“ [1].

BS2000/OSD als Server

BS2000/OSD kann als reiner Datenserver eingesetzt werden. Dabei befinden sich die Daten (Datenbanken und Dateien) auf einem BS2000-Rechner. Die Anwendungen sind auf einem anderen Rechner abgelegt. Dies ist bei einer geringen Zahl von Datenzugriffen pro Transaktion sinnvoll.

Beim Einsatz als Server für Anwendungen und Daten befinden sich die Anwendungen und die Daten auf demselben BS2000-Rechner. Dies ist dann vorteilhaft, wenn pro Transaktion häufig auf die Daten zugegriffen wird.

Als File-Server bietet BS2000/OSD die Kapazität, Zugriffsgeschwindigkeit und Zugriffssicherheit seiner Speichersubsysteme.

Als Backup-Server kann BS2000/OSD Datenbestände aus dem Netz auf seinen Speichermedien hinterlegen und die dort verfügbaren Sicherungsmechanismen von HSMS nutzen (siehe Handbuch „HSMS (BS2000/OSD)“ [21]).

Als Print-Server stellt BS2000/OSD seine Drucker über ein verteiltes Spool- und Drucksystem zur Verfügung (siehe SPOOL-Handbücher [32] und [33]). Dadurch können die Benutzer von Arbeitsplätzen auf UNIX-Systemen ihre Druckaufträge an BS2000-Druckern schnell und kostengünstig ausdrucken.

POSIX und das World Wide Web (WWW)

Für den Anschluss des BS2000 an das World Wide Web stehen folgende Produkte zur Verfügung:

- APACHE (BS2000/OSD) ist der Web-Server auf BS2000/OSD. APACHE (BS2000/OSD) ist eine Portierung des World Wide Web-Servers Apache der Apache Group.
- Die Kommunikationssoftware openNet Server repräsentiert das openNetworking im BS2000/OSD und gliedert sich wiederum in Produkte, wie z. B. DCAM und BCAM.
- interNet Services umfasst die Produkte FTP, TELNET, DNS, NTP, OPENSSH, SMTP-Server und IMAP-/POP3-Server. Diese Produkte sind Portierungen von entsprechenden Internet-Standardprodukten, die an die spezifischen Gegebenheiten von BS2000/OSD angepasst wurden.
- JENV ist die BS2000/OSD-Umgebung für JAVA.

Auch BS2000/OSD-Anwendungen können sich mit ihren Daten im Internet präsentieren. Über ein Common Gateway Interface (CGI), das durch den WWW-Server im BS2000/OSD verfügbar ist, wird der direkte Zugriff auf die Anwendungen realisiert. Diese Anwendungen können mit oder ohne Transaktionssteuerung durch openUTM ablaufen und auf beliebige BS2000/OSD-Datenhaltungen zugreifen.

Die Sicherheitsfunktionen von BS2000/OSD gewährleisten, dass nicht alle Daten jedem frei zugänglich sind.

BS2000/OSD-Anwendungen ins WWW bringen

Bestehende BS2000/OSD-Anwendungen können mit geringem Aufwand WWW-fähig gemacht werden. Dabei werden mit dem Produkt WebTransactions alphanumerische Oberflächen (Masken) in HTML-Formate umgesetzt und den WWW-Browsern zur Ausgabe übergeben. Weiterführende Informationen dazu finden Sie in den WebTransactions-Handbüchern [38] und [39].

Die Zahl der angebotenen Host-Anbindungen wird ständig erweitert. Spezialanbindungen werden mit der nötigen Einbindungstechnik momentan als Projektleistung von Fujitsu angeboten.

Einen Überblick über die Dokumentation zu WebTransactions finden Sie im Internet unter *http://manuals.ts.fujitsu.com*.
Wählen Sie: *Software* > *openSEAS* > *WebTransactions* und dann die jeweilige Produktgruppe bzw. das Produkt.

Verteilte Datenhaltung

Bei der verteilten Datenhaltung können Sie sowohl mit lokalen als auch mit fernen Daten arbeiten. Dadurch können Sie Datenbestände an die kostengünstigste Stelle in einem Rechnernetz legen.

Sie können von einer Workstation auf BS2000-Dateien zugreifen, nachdem die BS2000-Dateien in ein POSIX-Dateisystem kopiert wurden. Sie können aber auch BS2000-Dateien in ein POSIX-Dateisystem kopieren und auf eine Workstation laden.

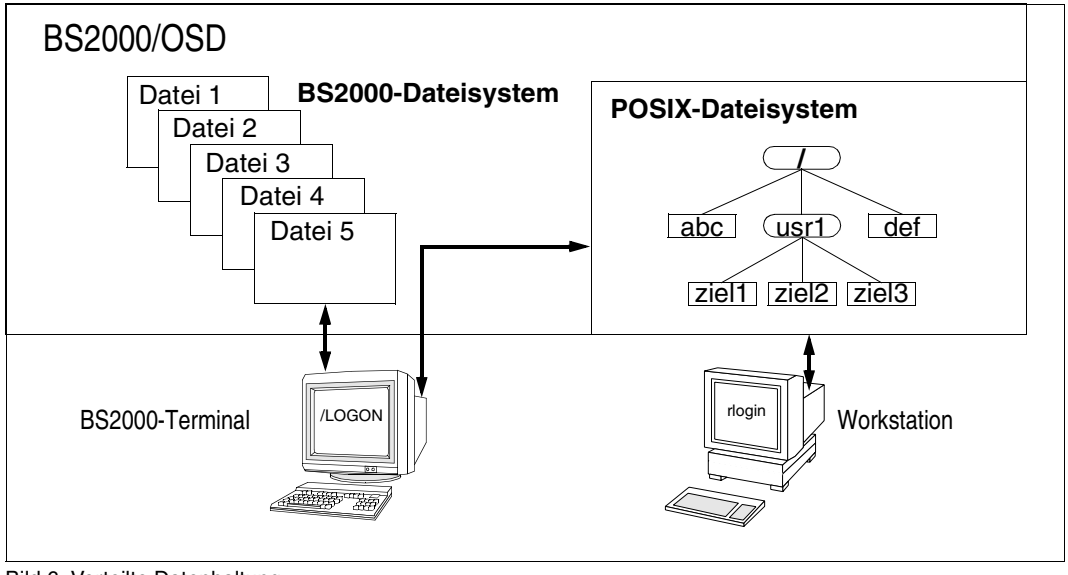

Bild 6: Verteilte Datenhaltung

Verteilte Verarbeitung

Mit der verteilten Verarbeitung können Sie die Rechnerkapazitäten optimal nutzen. Bei dieser Form der Client-Server-Architektur findet die Verarbeitung am jeweils geeignetesten Ort statt. Beispielsweise können Eingabeprüfungen und Kalkulationen auf einer Workstation ablaufen, während Datenbankzugriffe und rechenintensive Auswertungen auf einem BS2000-Rechner laufen.

Gemeinsame Entwicklungstools

Auf UNIX-Systemem basierende Entwicklungstools können ohne großen Aufwand nach POSIX portiert werden wie z.B. Tools, die der Gnu Public Licence (GPL) unterliegen. Damit lassen sich Anwendungsprogramme erstellen, die auf UNIX-Systemen und auf POSIX ablaufen können.

2.1.3 Bestandteile von POSIX

POSIX setzt sich aus folgenden Bestandteilen zusammen:

- **Subsystem POSIX**
  Es beinhaltet einen ins BS2000/OSD portierten UNIX-Systemkern.
- **POSIX-Shell**
  Sie stellt mit den Shell-Kommandos entsprechend dem XPG4-Standard die Verbindung zwischen dem Systemkern und dem BS2000/OSD her.
- **Shared Libraries**
  Die Komponenten zur Unterstützung von Shared Objects werden mit POSIX ausgeliefert.
- **POSIX-Sockets und XTI**
  Die Programmschnittstellen für die Transportsystem- und Kommunikationsdienste werden mit POSIX-Sockets und XTI (X/Open Transport Interface) bereitgestellt und sind Bestandteil von BS2000/OSD-BC.

Zusätzlich bietet POSIX additive Programmschnittstellen, die über XPG4-Standard hinausgehen, Netzwerk-Programmschnittstellen (TLI, RPC und XDR) und Header-Dateien für die Programmentwicklung. Außerdem ist es möglich, POSIX-Events für die Erstellung von SecureAuditTrails (nutzbar nur mit dem BS2000-Produkt SECOS) zu protokollieren.

2.1.4 Hardware-Voraussetzungen für POSIX

POSIX kann auf allen Zentraleinheiten ablaufen, auf denen BS2000/OSD-BC ab V7.0 ablauffähig ist. POSIX wird als Liefereinheit POSIX-BC als Bestandteil von BS2000/OSD-BC ausgeliefert. Die Versionsabhängigkeiten zwischen POSIX-BC und BS2000/OSD-BC entnehmen Sie bitte der Freigabemitteilung zu POSIX-BC.

2.1.5 Unterstützung von Terminals

POSIX unterstützt neben den im BS2000 verwendeten Blockterminals auch die an UNIX-Systemen verwendeten Zeichenterminals. Diese Terminals sind an UNIX-Mehrplatzsystemen angeschlossen und werden von POSIX über Netze bedient. Beim Zugriff auf POSIX über eine Workstation wird ein Zeichenterminal emuliert. Bei UNIX-Workstations ist das z.B. ein Terminal vom Typ 97801.

Block- und Zeichenterminals unterscheiden sich durch ihre Funktionsweise:

- Von Blockterminals aus können BS2000- und POSIX-Kommandos eingegeben werden, wobei aber die Eingabe von POSIX-Kommandos geringfügigen Einschränkungen unterliegt (siehe POSIX-Handbuch „Kommandos“ [1]).

  Bei Blockterminals ist keine zeichenweise Verarbeitung möglich. Blockterminals übergeben den gesamten am Bildschirm eingegebenen Text als Datenblock an den BS2000-Rechner. Kontrollfunktionen werden im Gerät selbst durchgeführt.

- Bei Zeichenterminals wird jedes eingegebene Zeichen sofort an das UNIX-System übertragen und von dort als Antwort auf die Eingabe an den Bildschirm übergeben und abgebildet. Kontrollfunktionen wie Schreibmarken-Bewegung, Groß-/Kleinschreibung oder Pufferung der Übertragung werden von dem Rechner durchgeführt, an dem das Terminal angeschlossen ist.

  Zeichenterminals werden in POSIX wie Dateien behandelt. Sie besitzen einen eindeutigen Namen. Von ihnen kann gelesen und auf sie kann geschrieben werden. Dazu werden die gleichen Funktionen wie beim Dateizugriff benutzt.

  Bildschirm-orientierte Anwendungen - wie z.B. der vi-Editor in UNIX-Systemen - erfordern zeichenorientierte Operationen. Deshalb können sie nur ablaufen, wenn sie an einem Zeichenterminal gestartet werden.

Manche Eingaben sind terminalabhängig, d.h. sie unterscheiden sich bei Block- und Zeichenterminals:

Blockterminal	**Zeichenterminal**
@@d	END
@@c	DEL
@@/	CTRL \
EM DUE	↵
@@z	CTRL Z
-	CTRL S / CTRL Q ...

Eine Aufteilung des Bildschirms existiert im BS2000 nicht. Die Bildschirmdarstellung erfolgt immer von oben nach unten. Eingaben und Ausgaben erfolgen jeweils in der untersten aktiven Zeile. Wenn der Bildschirm voll ist, wird der Inhalt jeweils um eine Zeile nach oben geschoben. Die oberste Zeile geht dadurch verloren. Ein Zugriff auf vorhergehende Zeilen ist nicht mehr möglich.

2.1.6 An POSIX angepasste BS2000-Softwareprodukte

POSIX-BC ist ein Subsystem von BS2000/OSD. Verschiedene BS2000-Softwareprodukte wurden an die POSIX-Schnittstellen angepasst oder auf POSIX-Basis ins BS2000 portiert:

- BLS (Binder-Lader-System) (siehe Seite 85)
- C/C++-Compiler (siehe Seite 86)
- COBOL85 / COBOL2000 (siehe Seite 88)
- CRTE
- JENV (Java) (siehe Seite 90)
- AID (siehe Seite 97)
- EDT (siehe Seite 91)
- HSMS (siehe Seite 92)
- SBA-BS2
- SM2
- NFS (siehe Seite 93)
- SDF-A
- SECOS (siehe Seite 94)

- SPOOL (siehe Seite 96)
- Dprint
- SORT (siehe Seite 98)
- SOCKETS/XTI (POSIX-SOCKETS) (siehe Seite 95)
- TLI (POSIX-NSL) (siehe Seite 96)
- File-Transfer-Produkte (openFT, openFT-FTAM, openFT-AC) (siehe Seite 91)
- openUTM
- OMINS
- interNet Services (früher TCP-IP-AP, TCP-IP-SV und interNet Value Edition) (siehe Seite 99)
- openNet Server
- APACHE Webserver auf BS2000/OSD (siehe Seite 101)
- WebTransactions

Weitere Informationen zu den genannten BS2000-Softwareprodukten finden Sie im Internet unter *http://manuals.ts.fujitsu.com*.
Wählen Sie: *BS2000/OSD mainframes* > *Current manuals* und dann die jeweilige Produktgruppe bzw. das Produkt.

2.2 POSIX-Dateisystem

Ein POSIX-Dateisystem ist eine Behälterdatei (Container) im BS2000 mit der Struktur eines UNIX-Dateisystems (UFS). Es kann wie in UNIX-Systemen aus mehreren Dateisystemen bestehen. Es ist hierarchisch aufgebaut und besteht aus Dateiverzeichnissen und Dateien (POSIX-Dateien).

An der Spitze der Hierarchie steht das Dateiverzeichnis *root*, das durch einen Schrägstrich (/) gekennzeichnet ist. Von hier aus setzt sich die Verzeichnisstruktur weiter nach unten fort. Von Dateiverzeichnissen aus kann in weitere Dateiverzeichnisse oder in Dateien verzweigt werden. Eine Datei ist der tiefste Verzweigungspunkt. Von einer Datei aus ist keine Verzweigung mehr möglich.
Einschränkungen gibt es weder für die Anzahl der Verzeichnisebenen noch für die Anzahl der Dateiverzeichnisse und Dateien auf einer Ebene. Deshalb lässt sich ein POSIX-Dateisystem sehr gut strukturieren und organisieren.

Dateiverzeichnisse werden auch als Knotenpunkte eines POSIX-Dateisystems bezeichnet, in denen Namen von Dateien oder weiteren Dateiverzeichnissen stehen. Die Namen für die Dateiverzeichnisse und Dateien kann der Benutzer vergeben, wobei bestimmte Konventionen einzuhalten sind.

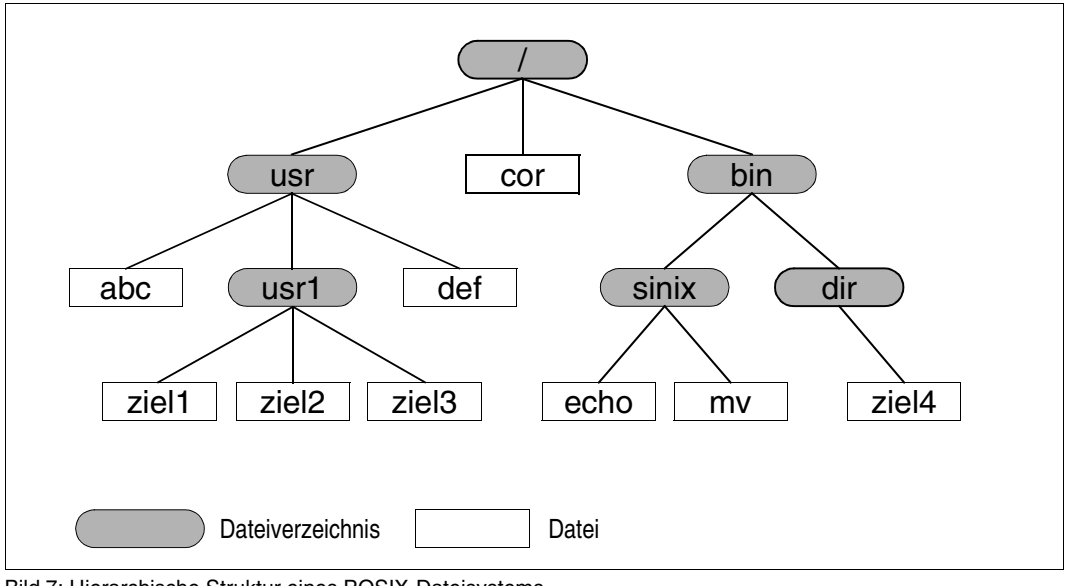

Bild 7: Hierarchische Struktur eines POSIX-Dateisystems

2.2.1 Vorteile eines hierarchischen Dateisystems

Ein hierarchisch strukturiertes Dateisystem wie das POSIX-Dateisystem bietet Ihnen mehrere Vorteile:

- Sie können ihren Datenbestand besser strukturieren.
- Sie können mit jeder Datei oder jedem Dateiverzeichnis des gesamten Dateisystems arbeiten, wenn Sie für die entsprechende Datei oder das entsprechende Dateiverzeichnis eine Zugriffsberechtigung besitzen (siehe Abschnitt „Zugriffsschutz für Dateien und Dateiverzeichnisse“ auf Seite 56).
- Eine Datei lässt sich leicht vom aktuellen Dateiverzeichnis in ein anderes Dateiverzeichnis übertragen. Dazu gibt es drei Möglichkeiten:
  - Sie können mit dem POSIX-Kommando *cp* (copy) eine Datei physikalisch in ein anderes Dateiverzeichnis kopieren; die Datei ist dann physikalisch mehrfach vorhanden.
  - Sie können aber auch nur den Namen einer Datei mit dem POSIX-Kommando *ln* (link) in ein anderes Dateiverzeichnis übertragen. Auf eine solche Datei bestehen dann mehrere Verweise; die Datei ist physikalisch aber nur einmal vorhanden.
  - Sie können mit dem POSIX-Kommando *mv* (move) eine Datei umbenennen oder an einen anderen Ort im Dateibaum versetzen. *mv* erzeugt innerhalb eines Dateisystems keine physische Kopie der versetzten oder umbenannten Datei, sondern ändert nur die Einträge im jeweils übergeordneten Dateiverzeichnis.

  Die POSIX-Kommandos *cp*, *ln* und *mv* sind im POSIX-Handbuch „Kommandos“ [1] ausführlich beschrieben.
- Sie können Ihre Dateien in ein oder mehrere Dateiverzeichnisse schreiben. Dadurch lassen sich Dateien übersichtlich und zusammenhängend organisieren.
- In einem Dateisystem dürfen mehrere Dateien mit demselben Namen vorhanden sein. Die Dateien müssen aber in unterschiedlichen Dateiverzeichnissen abgelegt sein.

2.2.2 Ablage von POSIX-Dateisystemen in Behälterdateien

POSIX-Dateisysteme werden im BS2000 in sog. Behälterdateien abgelegt; dies entspricht der in UNIX-Systemen üblichen Ablage von Dateisystemen in Partitions. Behälterdateien sind BS2000-PAM-Dateien, die sich auf einem Pubset befinden. Behälterdateien dürfen nicht auf Privatplatten oder auf Net-Storage abgelegt werden. Behälterdateien auf Shared Public Volume Sets (SPVS) können zu einem Zeitpunkt nur vom POSIX eines BS2000-Systems genutzt werden. Behälterdateien und andere BS2000-Dateien dürfen auf dem gleichen Pubset liegen.

Eine Behälterdatei wird ausschließlich durch das Subsystem POSIX bearbeitet, ihr Inhalt darf nicht durch andere Zugriffsmethoden verändert werden.

Aus der Sicht des Subsystems POSIX stellt eine Behälterdatei ein Dateisystem dar, das vom portierten UNIX-Systemkern verwaltet wird.

POSIX-Verwalter und BS2000-Systemverwalter mit Root-Berechtigung können Behälterdateien beim Einrichten neuer POSIX-Dateisysteme mit dem POSIX-Installationsprogramm anlegen (siehe Seite 131). Dabei wird auch die Größe der Behälterdatei und damit die des POSIX-Dateisystems festgelegt. Die Größe kann nachträglich mit *fsexpand* geändert werden.

Um für einen Benutzer den Speicherplatz zu begrenzen, kann für diesen Benutzer ein eigenes POSIX-Dateisystem mit der entsprechenden Größe eingerichtet werden. Dadurch wird der vorhandene Speicherplatz im BS2000 wirtschaftlicher genutzt.

Aus Performancegründen sollten die Behälterdateien von umfangreichen POSIX-Dateisystemen, die häufig benutzt werden, nicht auf demselben Pubset liegen, auf dem sich die Behälterdatei des Root-Dateisystems befindet.

2.2.3 Information über Dateisystem-Codierung (df)

Mit der Option *-c* gibt das Kommando *df* die Art der Codierung eines Dateisystems (EBCDIC oder ASCII) aus. Diese "Codierung" wird beim Einrichten eines ufs-Dateisystems über den Parameter POSIX filesystem *marker* = *yes* (EBCDIC) oder *no* (ASCII) vereinbart.

Die Auskunftsinformationen in den POSIX-Installationsmasken für Filesystem Administration werden im Hinblick auf die Art der Codierung ebenfalls verbessert (bisher war Auskunft nur über "modify"-Funktion möglich).

2.2.4 Vorteile durch das Anlegen mehrerer POSIX-Dateisysteme

Das Anlegen mehrerer POSIX-Dateisysteme bringt folgende Vorteile:

- Größere Datensicherheit
  Bei der Zerstörung eines POSIX-Dateisystems bleiben die übrigen POSIX-Dateisysteme erhalten.
  Einzelne POSIX-Dateisysteme können für eine Sicherung ausgewählt werden. Unveränderte POSIX-Dateisysteme können von einer Sicherung ausgenommen werden.
- Größerer Datenschutz
  Nur aktuell benötigte POSIX-Dateisysteme werden eingehängt und damit dem Benutzer verfügbar gemacht.
- Bessere Übersichtlichkeit und Strukturierung
  Ein POSIX-Dateisystem kann speziell für einen Benutzer oder für ein Projekt angelegt werden.

2.2.5 Konventionen für Namen von POSIX-Dateien und Dateiverzeichnissen

Jede Datei und jedes Dateiverzeichnis in einem POSIX-Dateisystem hat einen eindeutigen Pfadnamen. Der Pfadname gibt die Position einer Datei oder eines Dateiverzeichnisses innerhalb eines POSIX-Dateisystems an und zeigt, wie darauf zugegriffen werden kann. Der Pfadname besteht aus den Namen aller darüberliegenden Dateiverzeichnisse, ausgehend vom Dateiverzeichnis *root*, und dem eigentlichen Namen der Datei oder des Dateiverzeichnisses. Die Namen der Dateiverzeichnisse werden jeweils durch einen Schrägstrich (Slash) voneinander getrennt. Wenn man vom POSIX-Dateisystem in Bild 7 auf Seite 33 ausgeht, dann hat z.B. der Pfad vom Dateiverzeichnis *root* zur Datei *echo* folgenden Namen: */bin/sinix/echo*.

Wenn Sie eine Datei oder ein Dateiverzeichnis ohne Pfadangabe einrichten, wird der Name automatisch immer in dem Dateiverzeichnis eingetragen, in dem Sie sich gerade befinden.

Die Pfadnamen von POSIX-Dateien können maximal 1023 byte lang sein. Der Dateiname selbst kann maximal 255 Zeichen lang sein.

2.2.6 Kopieren und Konvertieren von Dateien

POSIX-Dateien enthalten keine Datensätze, sondern sie sind byte-orientiert.
BS2000-Dateien dagegen enthalten satzorientierte und/oder PAM-Block-orientierte Daten.

POSIX behandelt Dateien standardmäßig im EBCDIC-Format, UNIX-Systeme, MS-DOS und Windows im ASCII-Format. Im POSIX-Dateisystem abgelegte ASCII-Dateien können in der POSIX-Shell nur bearbeitet werden, wenn sie vorher konvertiert wurden.

Damit Dateien der beiden Formate wechselseitig benutzt werden können, stehen Kopier- und Konvertierungsroutinen zur Verfügung. Dabei wird vom Zeichensatz EBCDIC.DF.03 in den ASCII-ISO-7-Bit-Code konvertiert und umgekehrt. Diese Konvertierung ist nur für Textdateien sinnvoll.

Zusätzlich stehen Konvertierungsroutinen zur Verfügung, um vom Zeichensatz EBCDIC.DF.04 in den 8-Bit-Zeichensatz (ISO 8859) zu konvertieren und umgekehrt.

Automatische Konvertierung

Mit der Umgebungsvariable *IO_CONVERSION* wird gesteuert, ob Dateien beim Zugriff mit POSIX-Kommandos (z.B. *awk*, *cat*, *grep*...) auf montierte ASCII-Dateisysteme automatisch konvertiert werden. Standardmäßig ist die Umgebungsvariable *IO_CONVERSION* mit dem Wert „NO“ belegt, d.h. es erfolgt keine automatische Konvertierung. Die automatische Konvertierung wird mit folgendem Kommando eingeschaltet:

```
export IO_CONVERSION=YES
```

Als ASCII-Dateisysteme werden behandelt:

- Ferne mit NFS eingehängte UNIX-/Windows-Dateisysteme
- Lokale POSIX-Dateisysteme mit POSIX-Dateisystem-Marker=N
- Ferne mit NFS eingehängte POSIX-Dateisysteme mit POSIX-Dateisystem-Marker=N

Falls die automatische Konvertierung für einen POSIX-Benutzer bereits beim Starten der POSIX-Shell voreingestellt sein soll, muss dieses *export*-Kommando in die Datei *.profile* im HOME-Verzeichnis dieses Benutzers eingetragen werden.

Bei Verwendung der folgenden Tools darf die automatische Konvertierung nicht eingeschaltet sein, da diese Tools selbst konvertieren:
dd, *iconv*, *bs2cp* mit Schalter *-k*.

Behandlung von Archiven/Bibliotheken:
ar konvertiert nicht automatisch, da *ar*-Bibliotheken oft binäre Daten enthalten.

pax und *tar* konvertieren automatisch. Ein *pax*- oder *tar*-Archiv darf jedoch nicht mit *cp* kopiert werden, wenn die automatische Konvertierung eingeschaltet ist.

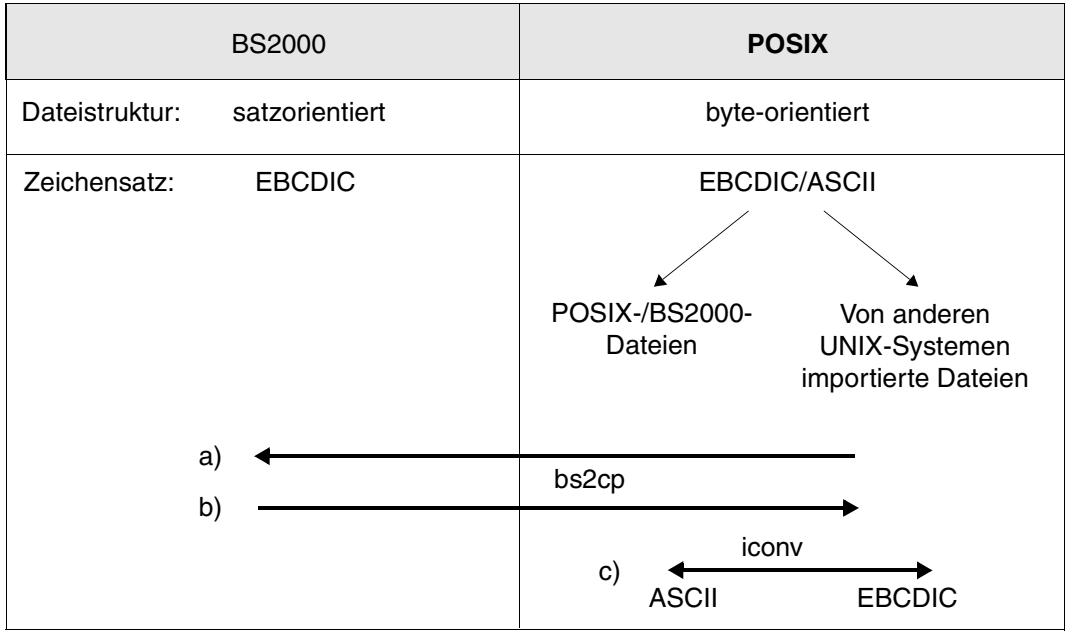

Bild 8: Austausch und Konvertierung von Dateien

a) Dateien von POSIX nach BS2000 übertragen (aus Sicht der POSIX-Shell):

   Mit dem POSIX-Kommando *bs2cp* übertragen Sie Dateien von POSIX nach BS2000. Sie müssen nicht die Option *-k* angeben, wenn in beiden Dateisystemen die Dateien im EBCDIC-Zeichensatz vorliegen.

   Zusätzlich können Sie für die BS2000-Datei noch Dateiattribute bestimmen. Dazu müssen Sie vor dem „Kopier"kommando *bs2cp* mit dem POSIX-Kommando *bs2file* die BS2000-Dateiattribute festlegen. *bs2file* wird auf das BS2000-Kommando SET-FILE-LINK abgebildet.

b) Dateien von BS2000 nach POSIX übertragen (aus Sicht der POSIX-Shell):

   Mit dem POSIX-Kommando *bs2cp* übertragen Sie Dateien von BS2000 nach POSIX. Sie müssen nicht die Option *-k* angeben, wenn in beiden Dateisystemen die Dateien im EBCDIC-Zeichensatz vorliegen.

   Abhängig von der Art der BS2000-Datei (SAM, ISAM) ist folgendes zu berücksichtigen:

   – Bei SAM-Dateien können Sie wählen, ob die Datei als Textdatei, als Binärdatei oder als binäre Textdatei im POSIX-Dateisystem hinterlegt wird. Dazu müssen Sie vor dem Kopierkommando *bs2cp* noch die Bearbeitungsart der Datei mit dem POSIX-Kommando *ftyp* festlegen.

   – ISAM-Dateien werden generell als Textdateien im POSIX-Dateisystem abgelegt.

c) Zur Konvertierung innerhalb eines POSIX-Dateisystems dient das POSIX-Kommando *iconv*. Es werden die Datei-Inhalte konvertiert.

2.2.7 Zugriff auf POSIX-Dateisysteme im BS2000

Beim Zugriff auf POSIX-Dateisysteme im BS2000 in **EBCDIC-Code** ist folgendes zu beachten:

- Beim Zugriff aus dem BS2000 sind keine Maßnahmen erforderlich.
- Beim Zugriff aus UNIX-Systemen müssen die Dateien erst konvertiert werden, z.B. mit dem POSIX-Kommando *iconv* (siehe POSIX-Handbuch „Kommandos“ [1]).

Beim Zugriff auf POSIX-Dateisysteme im BS2000 in **ASCII-Code** ist folgendes zu beachten:

- Beim Zugriff aus dem BS2000:

  CRTE bietet eine *automatische* Konvertierung in EBCDIC-Code an, wobei aber folgende Einschränkungen gelten:

  – Das zugehörige Dateisystem darf nicht als „von POSIX erzeugt“ markiert sein (siehe „Angaben zum POSIX-Dateisystem:“ auf Seite 129).

  – Die Datei muss mit dem Aufruf *fopen* eröffnet worden sein.

  – Die Datei darf nicht im Binärmodus eröffnet worden sein.

  – Die Umgebungsvariable *IO_CONVERSION* existiert nicht oder hat den Wert *YES*.

  Außerdem bietet CRTE eine *explizite* Konvertierung durch die Bibliotheksfunktionen *ascii_to_ebcdic* und *ebcdic_to_ascii* an.

- Beim Zugriff aus UNIX-Systemen sind keine Maßnahmen erforderlich.

2.2.8 Zugriff auf POSIX-Dateien

Auf POSIX-Dateien können Sie über POSIX-Programmschnittstellen zugreifen (siehe Abschnitt „POSIX-Programmschnittstellen“ auf Seite 71). Mehrere BS2000-Softwareprodukte unterstützen einen Zugriff auf POSIX-Dateien (siehe Kapitel „BS2000-Softwareprodukte im Umfeld von POSIX“ auf Seite 85).

2.2.9 Zugriff auf BS2000-Dateien und PLAM-Bibliothekselemente über das bs2fs-Dateisystem

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.

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.

Näheres dazu finden Sie im POSIX-Handbuch „BS2000-Dateisystem bs2fs“ [2].

2.2.10 Zugriff auf ferne Dateien

Mit POSIX können Sie nur auf POSIX-Dateisysteme zugreifen, die sich am lokalen Rechner befinden. Um auch mit den Dateisystemen eines fernen Rechners arbeiten zu können, muss das Softwareprodukt NFS (Network File System) auf dem fernen und lokalen Rechner installiert sein. Am fernen Rechner (NFS-Server) muss das einzuhängende Dateisystem mit dem NFS-Kommando *share* bereitgestellt und am lokalen Rechner (NFS-Client) mit dem NFS-Kommando *mount* eingehängt werden. Danach kann auf das ferne Dateisystem vom lokalen Rechner aus zugegriffen werden. NFS gibt es für BS2000/OSD, UNIX-Systemen, MS-DOS und Windows. NFS ist im Handbuch „NFS (BS2000/OSD)“ [8] beschrieben.

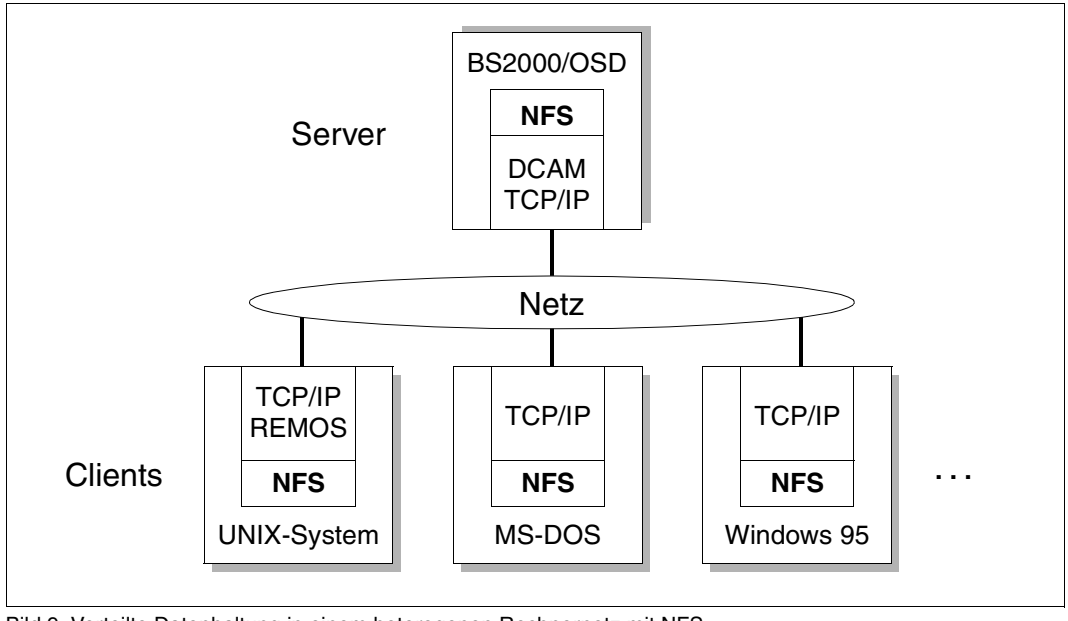

Bild 9: Verteilte Datenhaltung in einem heterogenen Rechnernetz mit NFS

r-Kommandos

Das Kommando *rcp* (remote copy) ermöglicht das Kopieren von Dateien oder ganzen Dateibäumen zwischen POSIX-BS2000 und UNIX-Systemen. Auch das Kopieren zwischen zwei POSIX-BS2000-Systemen ist möglich, wenn eine TCP/IP-Verbindung besteht.
Bei *rcp* wird automatisch eine ASCII/EBCDIC-Konvertierung durchgeführt. Soll keine Konvertierung stattfinden, so muss *rcp* mit dem Schalter *-b* (binary) aufgerufen werden.

Mit dem Kommando *rsh* (remote Shell) können Kommandos auf einem UNIX-System ausgeführt werden.

Eine ausführliche Beschreibung der Kommandos *rcp* und *rsh* finden Sie im POSIX-Handbuch „Kommandos“ [1].

2.3 Große Dateien im POSIX-Dateisystem

In der Vergangenheit wurden in POSIX nur solche Dateien unterstützt, die maximal 2 Gigabyte groß sein können. Dies lag daran, dass die Daten innerhalb einer Datei mit einer Variablen des Datentyps integer (vorzeichenbehaftet) adressiert wurden. Damit lassen sich maximal 2^{31}–1 Bytes adressieren, das sind 2 Gigabyte.

Diese Grenze verursachte bei verschiedenen Einsatzfällen Probleme, z.B. bei Druckdateien mit speicherintensiven Grafiken, weshalb mehr Anwender forderten, die maximale Dateigröße zu erhöhen. Dies wurde auch von den Standardisierungsgremien unterstützt und führte dazu, dass ein neuer Standard für große Dateien festgelegt wurde.

Standard für große Dateien

Kernpunkt dieses Standards ist, dass für die Adressierung innerhalb einer Datei eine Variable des Datentyps long long verwendet wird. Dieser Datentyp besteht aus einem integer-Paar, so dass man inklusive Vorzeichen eine Adressbreite von 2^{63}–1 Bytes erreicht.

Diese neue Klasse von Dateien sollte natürlich möglichst kompatibel zu den bestehenden sein, damit existierende Programme ohne großen Aufwand auch mit großen Dateien arbeiten können. D.h. große Dateien sollten syntaktisch und semantisch möglichst mit denselben Schnittstellen bearbeitet werden können wie bisherige Dateien.

2.3.1 Große POSIX-Dateisysteme

Die maximale Größe einer POSIX-Datei ist auch durch die Größe des Dateisystems beschränkt, in dem diese Datei liegt, d.h durch die Größe der Behälterdatei in BS2000, siehe Seite 35. Bisher konnte eine Behälterdatei maximal 2 Gigabyte groß sein, bedingt durch die interne Adressierung mit einer Variablen vom Typ integer.

Da zur internen Adressierung jetzt eine Variable vom Typ long long verwendet werden kann, lässt sich ein viel größerer Bereich adressieren, so dass Behälterdateien und damit auch die POSIX-Dateisysteme größer werden können als 2 Gigabyte.

Die folgende Tabelle zeigt die Grenzwerte für BS2000-Dateien und POSIX-Dateisysteme:

OSD-Version	Größe einer BS2000-Datei	Größe des POSIX-Dateisystems
ab V5.0	max. 4096 Gigabyte (= 4 Terabyte)	max. 1024 Gigabyte (= 1 Terabyte)

Die Differenz zwischen maximaler Größe von BS2000-Dateien und maximaler Behältergröße ist durch die vorgegebene Implementierung in POSIX bestimmt.
Die Größe einer Behälterdatei und damit eines POSIX-Dateisystems wird beim Einrichten mit dem Verwaltungs-Tool POSINST festgelegt, siehe Abschnitt „Administrate POSIX filesystems (POSIX-Dateisysteme verwalten)“ auf Seite 131.

2.3.2 Große POSIX-Dateien

Große POSIX-Dateien sind Dateien eines POSIX-Dateisystems, die größer als 2 Gigabyte sein können. Große POSIX-Dateien können nur in POSIX-Dateisystemen angelegt werden, die auf einem großen Behälter basieren und damit den Grenzwert von 2 Gigabyte überschreiten können, siehe vorheriger Abschnitt.

Die maximale Größe einer POSIX-Datei ist durch die Größe der Behälterdatei begrenzt, die sie enthält. Außerdem können Sie in POSIX eine maximale Dateigröße angeben, der für alle Dateien des POSIX-Dateisystems gilt (Kommando *ulimit* oder Parameter FILESIZE in der POSIX-Informationsdatei).

Programmschnittstellen für große POSIX-Dateien

Um mit POSIX-Dateien zu arbeiten, gibt es eine Anzahl von C-Bibliotheksfunktionen, wie *open()*, *close()*, die von CRTE zur Verfügung gestellt werden. Eine Teilmenge dieser Funktionen liegt in 64-bit-Ausprägung vor, um große POSIX-Dateien zu bearbeiten. Diese Funktionen haben den gleichen Namen, ergänzt um den Suffix „64“, z.B. *open64()*. Außerdem wurden einige Datenstrukturen und -typen auf 64-bit umgestellt. Näheres finden Sie in Abschnitt „Programmschnittstelle für große POSIX-Dateien“ auf Seite 77.

Shell-Kommandos für große POSIX-Dateien

Die meisten Dateiverarbeitungs-Kommandos der POSIX-Shell können große POSIX-Dateien erkennen und z.T. auch verarbeiten. Dabei unterschiedet man 2 Kategorien:

large file aware Das Kommando kann große POSIX-Dateien korrekt bearbeiten. Einige der Kommandos dieser Kategorie können große Dateien nur bis zu einer bestimmten Dateigröße bearbeiten, zum Beispiel *cpio* bis max. 8 Gbyte.

large file safe Das Kommando erkennt große POSIX-Dateien, weist die Verarbeitung jedoch zurück, z.B. mit einer entsprechenden Meldung.

Zu welcher Kategorie ein Kommando gehört, finden Sie im Abschnitt „Kommandoumfang der POSIX-Shell“ auf Seite 270 in der Spalte LFS.

Daneben werden BS2000-Programme, die mit POSIX-Dateien arbeiten und bei denen eine Verarbeitung großer Dateien notwendig oder sinnvoll erscheint, ebenso umgestellt (z.B. HSMS, SORT, SPOOL).

2.4 Journaling für Dateisysteme

Für den schnellen Wiederanlauf nach einem Systemabsturz bietet POSIX die Möglichkeit, ein Journal mit modifizierten Metadaten zu führen (Dateisystem-Journaling). Das Führen eines Journals muss explizit vereinbart werden, siehe unten, Abschnitt „Journaling vereinbaren“.

Wird ein solches Journal geführt, dann werden die modifizierten Metadaten beim Wiederanlauf entweder an ihre endgültige Position auf Platte geschrieben oder verworfen, je nach Status der zuhörigen Datenblöcke. Dadurch beschleunigt sich die Wiederherstellung eines konsistenten Zustandes des Dateisystems, da nur noch die offenen Aktionen laut Journal bearbeitet werden müssen. Ein Dateisystem-Check (z.B. mit *fsck*) hingegen muss das komplette Dateisystem nach Inkonsistenzen durchsuchen.

Journaling vereinbaren

Journaling kann auf mehrere Arten vereinbart werden:

- Mit dem POSIX-Installationsprogramm bei der Erstinstallation

  Bei der Erstinstallation im Dialog und im Batch kann Journaling für das *root*- und/oder *var*-Dateisystem vereinbart werden. Die Erstinstallation erfolgt immer Offline, d.h. das POSIX-Subsystem ist nicht gestartet.

- Mit dem POSIX-Installationsprogramm bei der Filesystem Administration

  Bei der Filesystem Administration im Dialog und im Batch kann Journaling für beliebige Dateisysteme inkl. *root* und *var* vereinbart werden.

  Für *root* und *var* ist Journaling nur über *modify* vereinbar und wirkt sich erst beim nächsten POSIX-Start aus. Für andere Dateisysteme ist Journaling über *append* und *modify* vereinbar.

  Das Neuanlegen (*append*) oder Modifizieren (*modify*) von Dateisystemen erfolgt immer Online, d.h. das POSIX-Subsystem ist gestartet.

- Mit dem Kommando *mount*

  Durch die Option *-o journal* wird für das angegebene Dateisystem ein Journal angelegt, siehe POSIX-Handbuch „Kommandos“ [1].

Das Journal wird im ufs-Dateisystem während des Mountvorganges angelegt (siehe Seite 173), sofern es nicht bereits existiert und nicht *readonly* montiert wird.

Ist nicht genügend Platz im Dateisystem für das Journal vorhanden, so wird es ohne Journaling eingehängt und auf der BS2000-Konsole wird auf den Engpass hingewiesen.

Die Größe des Journals hängt wie folgt von der Größe des Dateisystems ab:

Größe des Dateisystems	**Größe des Journals**
< 100 MB	1 MB
100 MB - 1600 MB	1 % der Grösse des Dateisystems
> 1600 MB	16 MB

2.5 POSIX als Subsystem im BS2000

POSIX ist ein privilegiertes BS2000-Subsystem, das die Aufträge privilegierter und nicht-privilegierter Benutzer bearbeitet. Das Subsystem POSIX besteht im wesentlichen aus drei Teilen:

- einem UNIX-Systemkern, der ins BS2000 portiert wurde;
- BS2000-Anschlüsse und Diensten, die eine Verbindung zwischen dem portierten UNIX-Systemkern und dem BS2000 herstellen.
- Routinen für die Initialisierung und Beendigung des Subsystems POSIX.

Das Subsystem POSIX unterstützt das POSIX-Dateisystem.

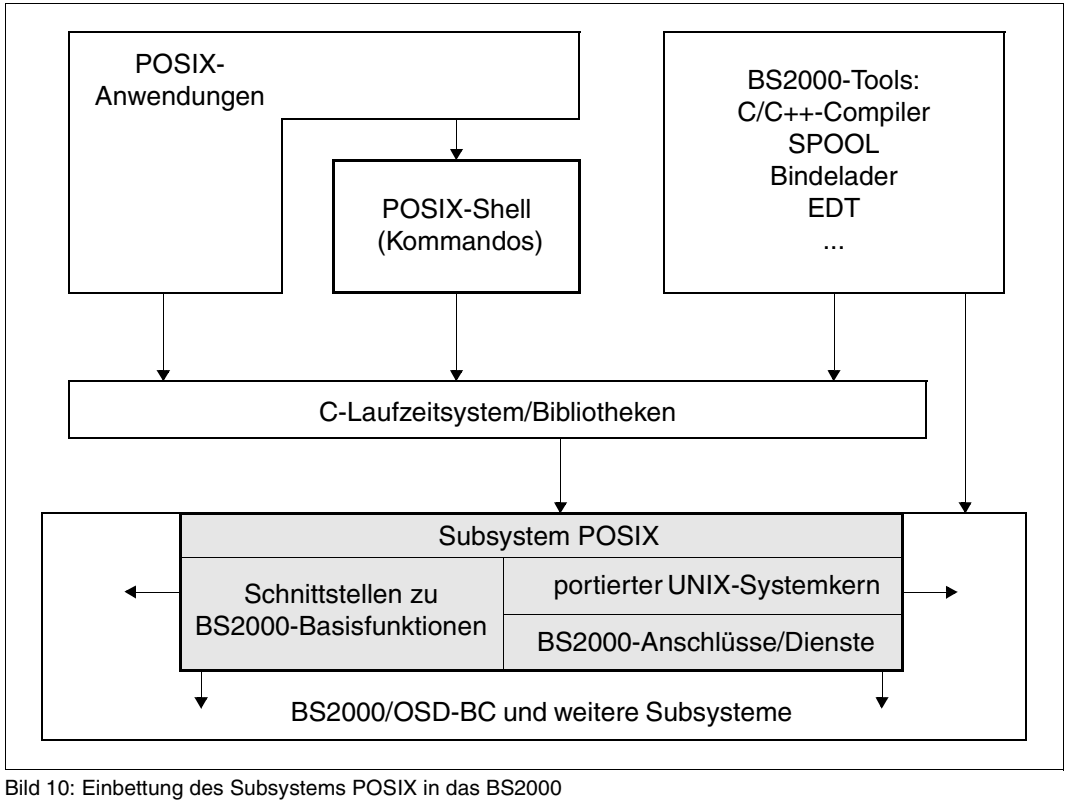

Bild 10: Einbettung des Subsystems POSIX in das BS2000

Allgemeine Informationen zu Subsystemen im BS2000 finden Sie im Handbuch „DSSM /SSCM“ [29].

2.5.1 Verwaltung des Subsystems POSIX durch DSSM

Die Dynamische Subsystem-Verwaltung (DSSM) des BS2000/OSD bindet das Subsystem POSIX aus Bindelademodulen, die in folgenden plattformspezifischen Programmbibliotheken bereitgestellt werden:

– SYSLNK.POSIX-BC.<version> (/390)

– SPMLNK.POSIX-BC.<version> (SPARC)

– SKMLNK.POSIX-BC.<version> (X86, ab BS200/OSD V8.0)

Der Systemkern-Code von POSIX enthält - ebenso wie der Systemkern-Code des Original-UNIX - einige Steuerparameter (Tuningparameter), die ein Systemverwalter in der POSIX-Informationsdatei SYSSSI.POSIX-BC.<version> entsprechend dem speziellen Einsatzfall setzen kann. Diese Steuerparameter dienen zum Konfigurieren des Systemkerns und zur Verbesserung der Performance. Außerdem wird in der POSIX-Informationsdatei der Name des Root-Dateisystems festgelegt.

In POSIX wird der UNIX-Tuningmechanismus auf den Parameterservice von DSSM abgebildet. Die POSIX-Informationsdatei enthält neben den Steuerparametern des Systemkern-Codes den Namen des Root-Dateisystems und andere POSIX-spezifische Steuerparameter. Der Inhalt der POSIX-Informationsdatei ist auf Seite 145 beschrieben.

Die POSIX-Informationsdatei wird zusammen mit anderen Komponenten an den Kunden ausgeliefert. Sie ist als SAM-Datei eingerichtet und enthält bereits Standardwerte. Die Standardwerte sind so gewählt, dass das Subsystem POSIX in beliebiger Umgebung ablaufen kann, ohne das Gesamtsystem durch übermäßigen Ressourcenverbrauch zu belasten. In vielen Fällen wird es aber sinnvoll sein, einige Steuerparameter, wie z.B. die maximale Anzahl von POSIX-Prozessen, an die spezielle POSIX-Anwendung und den Ressourcenvorrat des Gesamtsystems anzupassen.

Wenn der Systemverwalter einen ungültigen Parameterwert in die POSIX-Informationsdatei einträgt, wird die Meldung POS1020 ausgegeben. Statt des ungültigen Parameterwertes wird der Standardwert in die subsysteminterne Parametertabelle eingetragen.

Wenn beim Start des Subsystems POSIX keine POSIX-Informationsdatei vorhanden ist oder wenn sie nicht eröffnet werden kann, wird eine entsprechende Meldung ausgegeben und das Subsystem nicht gestartet.

Einige ausgewählte Steuerparameter lassen sich mit dem privilegierten POSIX-Kommando *usp* auch während einer laufenden POSIX-Session ändern. Die gewünschten Ressourcen stehen dann ohne Neustart des Subsystems zur Verfügung.

2.5.2 POSIX-Prozessverwaltung

In POSIX findet der Programmablauf in einem Prozess statt, im BS2000 in einer Task. POSIX-Prozesse werden auf BS2000-Tasks abgebildet.

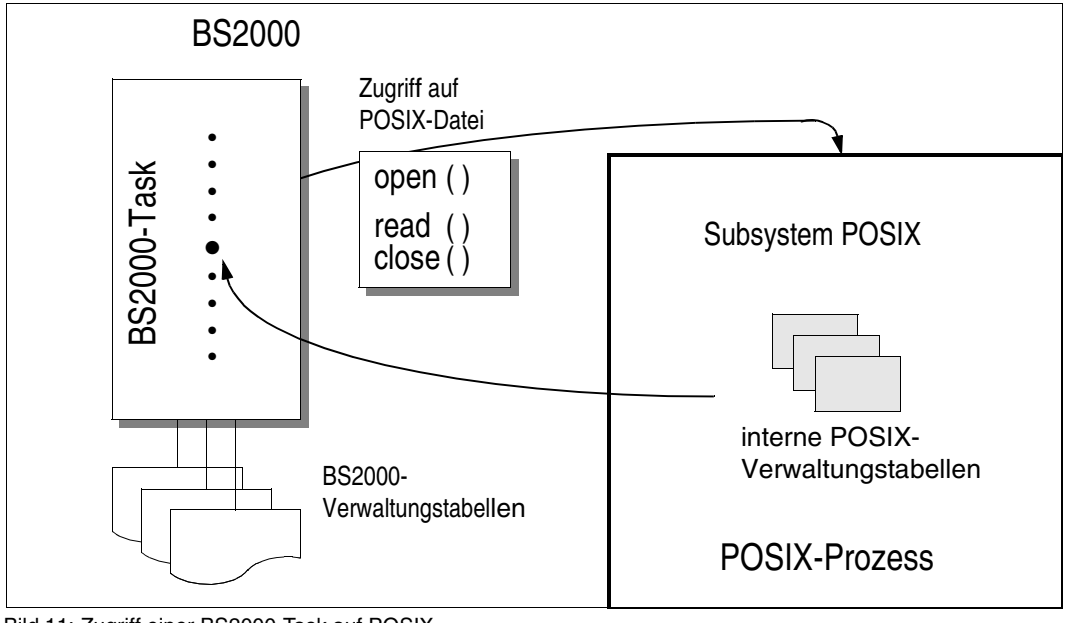

Bild 11: Zugriff einer BS2000-Task auf POSIX

In POSIX sind sämtliche Prozesse hierarchisch strukturiert. Die Prozess-Hierarchie entsteht durch einen initiierenden Prozess (*init*) und weitere Prozesse, die diesem initiierenden Prozess untergeordnet sind. Man spricht von einer Vater-Sohn-Beziehung. Der initiierende Prozess ist der Vater aller Prozesse. Die direkt untergeordneten Prozesse sind die Söhne, die wiederum Söhne haben können. Diese Rangfolge kann bis zu einer konfigurierbaren maximalen Prozessanzahl (Steuerparameter NPROC in der POSIX-Informationsdatei) fortgeführt werden.

Im Folgenden sind die einzelnen POSIX-Mechanismen und Prozesse näher beschrieben.

fork

Durch den Aufruf der Funktion *fork* wird von einem Vaterprozess ein neuer Sohnprozess erzeugt. Die Funktion *fork* erzeugt eine neue Prozessumgebung und kopiert ausgewählte Informationen des Vaterprozesses für den Sohnprozess. Dem Sohnprozess steht ein eigener, vom Vaterprozess abgesonderter Adressraum zur Verfügung. Der Sohnprozess kann auf alle vom Vaterprozess geöffneten POSIX-Ressourcen zugreifen.

Die beiden Prozesse laufen direkt nach dem Funktionsaufruf unabhängig voneinander weiter. Sie können durch den jeweiligen Returncode unterschieden werden: Der Sohnprozess erhält den Wert 0 zurück, der Vater die Prozessidentifikation (PID) des Sohnprozesses.

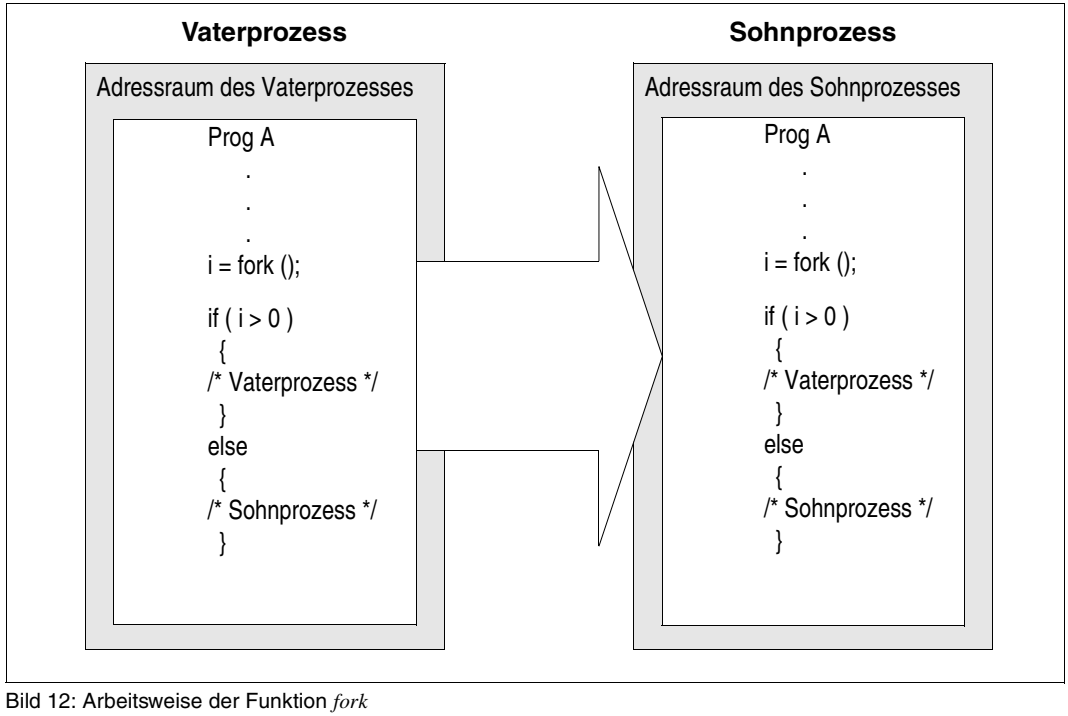

Bild 12: Arbeitsweise der Funktion *fork*

Bei einem Aufruf der Funktion *fork* werden DVS-Dateien und weitere BS2000-Betriebsmittel nicht mitvererbt.

exec

Wenn in einem Programm die Funktion *exec* aufgerufen wird, wird die aktuelle Prozessumgebung von einer neuen vollkommen überlagert. Dadurch kann z.B. in einem Sohnprozess ein anderes Programm ablaufen als im Vaterprozess. Die Prozesse bleiben aber durch die Vater-Sohn-Beziehung weiterhin verbunden.

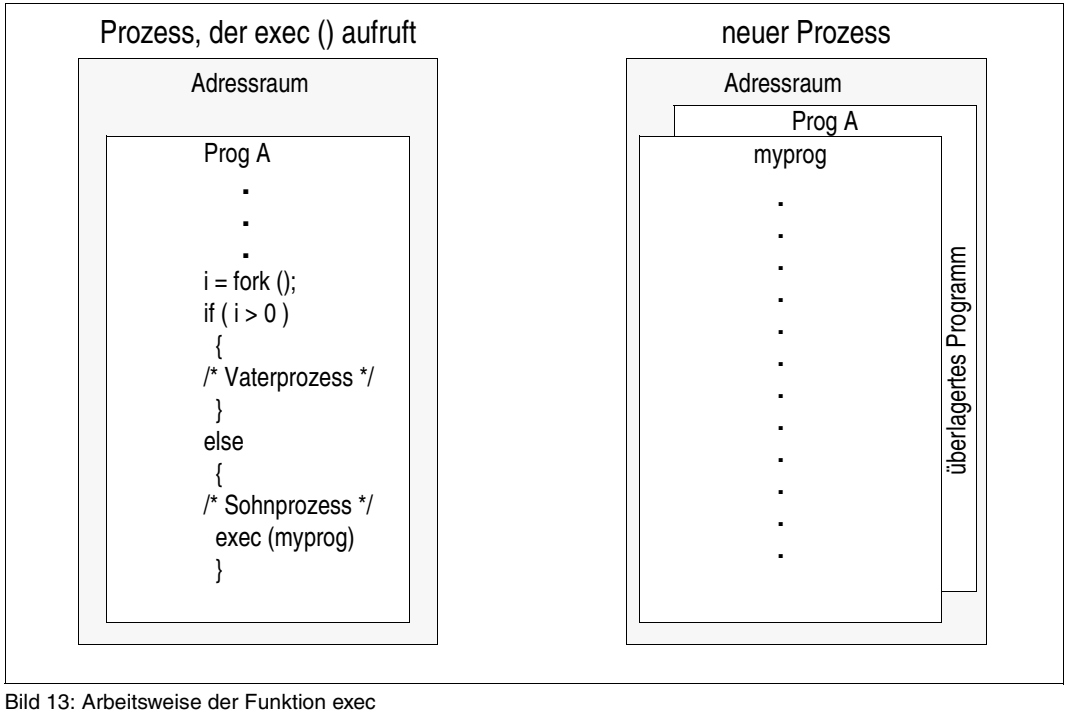

Bild 13: Arbeitsweise der Funktion exec

Kombination von fork und exec

Die Funktionen *fork* und *exec* können auch kombiniert werden. Der Vorteil einer solchen Kombination besteht darin, dass Teilaufgaben auf einen anderen Prozess ausgelagert werden können. Nachdem alle Teilaufgaben beendet sind, kann der Prozess beendet werden.

Ein Beispiel dafür ist die POSIX-Shell. Die POSIX-Shell startet für einige Kommandos einen neuen Prozess, der sich selbst überlagert und das kommandoausführende Programm startet. Nachdem dieses Programm beendet ist, wird der dafür erzeugte Prozess beendet und die POSIX-Shell fortgesetzt.

pipe

Anwendungsprogrammierern steht mit POSIX die Funktion *pipe* zur Interprozess-Kommunikation zur Verfügung. Mit der Funktion *pipe* wird eine Pipe erzeugt. Eine Pipe ist ein Datenbehälter vom Typ „first in - first out“. Ein Prozess kann eine Pipe verwenden, um Informationen an einen anderen Prozess zu senden.

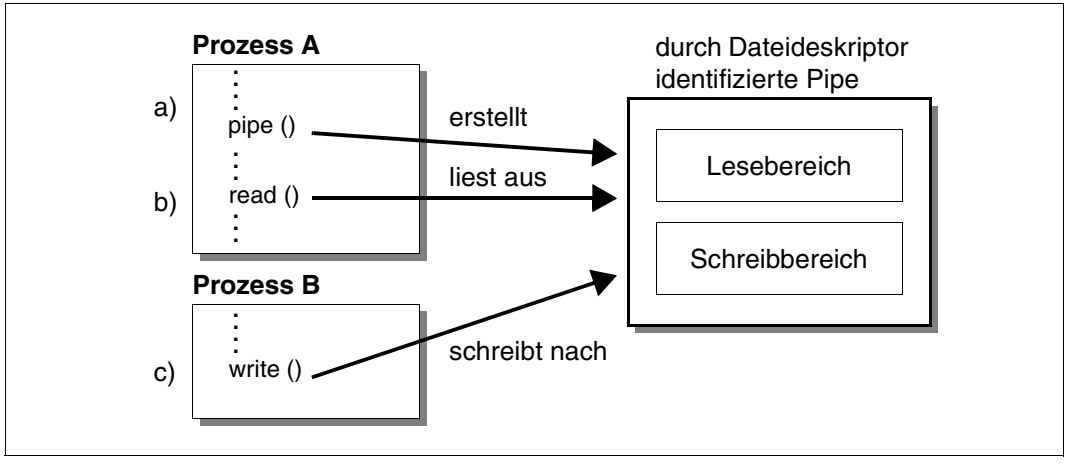

Bild 14: Arbeitsweise der unbenannten Pipe

a) Durch den Aufruf der Funktion *pipe* durch Prozess A wird eine Pipe zum Lesen und Schreiben von Daten erstellt. Die Pipe wird durch einen Dateideskriptor identifiziert. Der Dateideskriptor kennzeichnet eine Datei als offen.

b) Prozess B kann später in die Pipe schreiben und eine Nachricht hinterlassen, die Prozess A danach zu einem beliebigen Zeitpunkt lesen kann.

c) Prozess A ruft die Funktion *read* auf, spezifiziert den auf die Pipe bezogenen Dateideskriptor und liest die vom Prozess B hinterlegte Nachricht.

„copy-on-write“-Mechanismus

Beim Aufruf der Funktion *fork* wird die komplette Prozessumgebung vom Vater auf den Sohn vererbt. Dabei kann das Kopieren des kompletten Adressraums relativ lange dauern. Der *copy-on-rwrite*-Mechanismus verbessert die Performance erheblich: Bei einem *fork*-Aufruf wird der Adressraum nicht kopiert, sondern lediglich markiert. Somit steht der Speicher dem Vater- und dem Sohnprozess gemeinsam zur Verfügung. Wenn einer der beiden Prozesse zum ersten Mal auf eine Seite zugreift, wird anhand der Markierung festgestellt, dass eine Aktion auszuführen ist. In diesem Fall wird die betreffende Speicherseite für den Sohnprozess kopiert, so dass beide Prozesse ihre eigene Version besitzen. Dadurch wird erreicht, dass nur die Speicherseiten kopiert werden, die tatsächlich benötigt werden. Der *fork*-Aufruf kann somit wesentlich performanter ablaufen.

Ein weiterer Vorteil des *copy-on-write*-Mechanismus besteht darin, das häufig nach einem *fork*-Aufruf auch ein *exec*-Aufruf folgt. In diesem Fall ist das Kopieren des Speicherbereichs überflüssig, da er beim *exec*-Aufruf überlagert wird. Ohne *copy-on-write* würde der gesamte Speicherbereich kopiert und sofort wieder überschrieben.

Holdertask

POSIX benutzt eine Holdertask. Die Initialisierung und Terminierung laufen unter Kontrolle dieser Holdertask ab. Während der Subsystem-Sitzung, d.h. zwischen Initialisierung und Terminierung, steht die Holdertask dem Subsystem POSIX nicht zur Verfügung.

Aufrufertask

Die Aufrufertask wird beim ersten POSIX-Systemaufruf mit dem Subsystem POSIX verbunden, bei Programmende wird sie getrennt. Beim ersten POSIX-Systemaufruf erhält die Aufrufertask eine POSIX-Prozessumgebung.

Subsystem-private Server

Im Subsystem POSIX gibt es zwei Arten von subsystem-privaten Servern: Systemprozesse und Dämonen. Systemprozesse sind als BS2000-Systemtasks realisiert, Dämonen als nichtprivilegierte BS2000-Tasks.

Während das Kommando /START-SUBSYSTEM SUBSYSTEM-NAME=POSIX verarbeitet wird, werden nur Systemprozesse als Server initialisiert. Erst anschließend werden Dämonen erzeugt.

2.6 Sicherheitskonzept

In diesem Abschnitt ist beschrieben, wie POSIX in das BS2000 eingebettet wurde, um die Sicherheit des Gesamtsystems zu gewährleisten. Die nötigen Funktionen wurden zum Teil mit dem UNIX-Systemkern portiert, zum Teil sind sie Bestandteil des BS2000-Bausteins SRPM (System Resources and Privileges Management). Näheres zu SRPM finden Sie im Kapitel „POSIX-Benutzer verwalten“ auf Seite 181.

Das Sicherheitskonzept umfasst:

- die Benutzerdatenverwaltung
- die Gruppenverwaltung
- den Zugriffsschutz für Behälterdateien
- den Zugriffsschutz für Dateien und Dateiverzeichnisse
- den Zugangsschutz bei Zugang über einen fernen Rechner

2.6.1 Benutzerdatenverwaltung

Im POSIX-Standard sind Schnittstellen für die Sicherheitskontrolle eines Benutzers definiert. Mit diesen Schnittstellen werden feste Informationen über einen Benutzer erfragt, bevor er ein Betriebssystem benutzen darf. Folgende Benutzerdaten stehen für die Authentisierung zur Verfügung:

- Benutzerkennung/Login-Name des Benutzers
- Kennwort
- Benutzernummer
- Gruppennummer
- Initialwert für das Arbeitsverzeichnis
- zu startendes Programm

Diesen Informationen können nach Bedarf noch weitere hinzugefügt werden.

Beim Anmelden eines Benutzers im System wird die eingegebene Benutzerkennung und das zugehörige Kennwort gegen diese Informationen geprüft. Wenn die eingegebenen Werte stimmen, erhält der Benutzer Zutritt zum Betriebssystem. Näheres dazu finden Sie im Abschnitt „POSIX-Benutzerattribute vergeben“ auf Seite 185.

Zwischen den POSIX-Benutzerdaten und den BS2000-Benutzerdaten bestehen folgende Beziehungen:

- Der Login-Name des Benutzers und die BS2000-Benutzerkennung sind gleich.
- Das POSIX-Kennwort und das BS2000-LOGON-Kennwort sind gleich.

Zu den übrigen POSIX-Benutzerdaten gibt es auf der BS2000-Seite kein Äquivalent.

Für den Login-Namen gibt es in POSIX die Einschränkung, dass nur Großbuchstaben in Benutzernamen unterstützt werden.

Die POSIX-Benutzerdaten werden von der BS2000-Benutzerverwaltung gespeichert und verwaltet (näheres dazu siehe Kapitel „POSIX-Benutzer verwalten“ auf Seite 181). Sie sind als POSIX-Benutzerattribute in die BS2000-Benutzerdaten integriert. Der Zugriff auf POSIX-Benutzerdaten wird über die Benutzer- und Systemverwalterkommandos des BS2000 abgewickelt.

2.6.2 Gruppenverwaltung

Die Gruppenverwaltung in POSIX entspricht der im UNIX. Sie unterscheidet sich gegenüber der Gruppenverwaltung im BS2000 in folgenden Punkten:

- In POSIX dienen Gruppen ausschließlich der Zuteilung von Zugriffsrechten auf Dateien. Im BS2000 haben Gruppen zusätzlich den Zweck, den Verbrauch von Ressourcen wie Plattenspeicher, Rechnerleistung usw. zu steuern.
- In POSIX kann ein Benutzer gleichzeitig maximal 16 Gruppen angehören, im BS2000 nur einer einzigen Gruppe.
- BS2000-Gruppen sind hierarchisch angeordnet; in POSIX gibt es dieses Merkmal nicht.
- In POSIX kann ein Benutzer die aktuelle Gruppe wechseln; im BS2000 ist dies nicht möglich.

Wegen dieser großen Unterschiede existieren POSIX- und BS2000-Gruppen nebeneinander. Die POSIX- und BS2000-Gruppen werden getrennt verwaltet: Die POSIX-Gruppen auf der Shell-Ebene, die BS2000-Gruppen auf der BS2000-Ebene. Dies entspricht den unterschiedlichen Schutzmechanismen von POSIX- und BS2000-Dateien.

POSIX- und BS2000-Gruppen können unter Verzicht auf die Hierarchie identisch definiert werden, d.h. sie enthalten dann dieselben Benutzer.

Nähere Informationen zur Gruppenverwaltung finden Sie im Abschnitt „BS2000- und POSIX-Gruppen verwalten“ auf Seite 187.

2.6.3 Zugriffsschutz für Behälterdateien

POSIX-Dateisysteme werden in Behälterdateien (Container) abgelegt. Behälterdateien sind BS2000-PAM-Dateien im Non-Key-Format. Sie sind vor unberechtigtem Zugriff über die Standard-Zugriffskontrolle des BS2000 (ACCESS-/USER-ACCESS-Attribute) geschützt.

Das POSIX-Installationsprogramm legt Behälterdateien als nicht mehrfach benutzbar und mit ACCESS=*WRITE an. Diese Schutzattribute dürfen nicht verändert werden. Außerdem darf kein Dateikennwort vergeben werden.

Der Benutzer einer POSIX-Datei benötigt kein Zugriffsrecht für die Behälterdatei.

2.6.4 Zugriffsschutz für Dateien und Dateiverzeichnisse

Der Zugriffsschutz für Dateien und Dateiverzeichnisse ist in POSIX durch folgende Schutzmechanismen realisiert:

- Benutzerkennungen
- Kennworte für Benutzerkennungen
- Zusammenfassen von Benutzerkennungen zu Gruppen
- Schutzbits für Dateien und Dateiverzeichnisse

Diese Schutzmechanismen verhindern, dass ein Benutzer die Dateien und Dateiverzeichnisse eines anderen Benutzers unberechtigt lesen und verändern kann.

Zugriffsschutz durch Benutzerkennung, Kennwort und Gruppennummer

Jeder, der POSIX benutzen will, benötigt am entsprechenden BS2000-Rechner eine Benutzerkennung, die der BS2000-Systemverwalter einrichten muss. Der Benutzer selbst kann ein Kennwort festlegen oder verändern, um seine Benutzerkennung vor unberechtigtem Zugriff zu schützen.

Siehe auch Abschnitt „Zugangsschutz bei Zugang über einen fernen Rechner“ auf Seite 59.

Benutzer können zu Gruppen zusammengefasst werden. Dadurch können Dateien und Dateiverzeichnisse allen Mitgliedern dieser Gruppe zugänglich gemacht werden. Der Systemverwalter muss dazu jedem Benutzer eine Gruppennummer zuordnen. Benutzer mit der gleichen Gruppennummer gehören der gleichen Gruppe an (siehe Abschnitt „BS2000- und POSIX-Gruppen verwalten“ auf Seite 187).

Zugriffsschutz durch Schutzbits

Jeder Datei und jedem Dateiverzeichnis werden beim Erstellen automatisch die Benutzer- und Gruppennummer des erstellenden Prozesses und Schutzbits zugewiesen. Diese Schutzbits sind für bestimmte Zugriffe standardmäßig vorbelegt.
Es gibt Schutzbits für die folgenden drei Benutzerklassen:

- Eigentümer der Datei
- Gruppe, der der Eigentümer angehört
- Andere

Jede dieser Benutzerklassen besitzt je ein Schutzbit für Leseberechtigung (**r**ead), Schreibberechtigung (**w**rite) und Ausführberechtigung (e**x**ecute).

Beispiel

Eigentümer: `r w x`

Gruppe: `r w -`

Andere: `r - -`

Die Schutzbits gelten ausschließlich für ihre Benutzerklasse. Wenn z.B. nur der Eigentümer eine Zugriffsberechtigung für eine Datei besitzt, darf weder die Benutzerklasse *Gruppe* noch die Benutzerklasse *Andere* mit dieser Datei arbeiten.

Die Zugriffsberechtigungen haben für Dateien und Dateiverzeichnisse unterschiedliche Bedeutung:

Zugriffsberechtigung	**Datei**	**Dateiverzeichnis**
read	lesen	Einträge lesen
write	schreiben	Einträge (Dateien) löschen/anlegen
execute	ausführen	durchlaufen/durchsuchen

Vor dem ersten Schutzbit für den Eigentümer steht noch ein Identifikationszeichen, das automatisch vergeben wird. Es hat folgende Bedeutung:

- Datei

b blockorientiertes Gerät

c zeichenorientiertes Gerät

d Dateiverzeichnis

l Symbolischer Verweis

Die Schutzbits können mit dem POSIX-Kommando *chmod* geändert werden. Ein Benutzer mit der Benutzernummer 0 kann die Schutzbits von allen Dateien und Dateiverzeichnissen ändern, der Eigentümer nur von seinen eigenen Dateien und Dateiverzeichnissen. Auch wenn jemand aus der Benutzerklasse *Gruppe* oder *Andere* volle Zugriffsberechtigung auf eine Datei oder ein Dateiverzeichnis besitzt, kann er die Schutzbits nicht ändern.

Die Schutzbits für die Benutzerklasse *Gruppe* werden entsprechend der Gruppenzugehörigkeit des Eigentümers vergeben. Beim Anlegen einer neuen Datei wird die Gruppennummer und damit die Gruppenzugehörigkeit vom aktuellen Dateiverzeichnis übernommen.

Die aktuell gültige Schutzbit-Maske kann mit dem POSIX-Kommando *umask* ausgegeben oder geändert werden. Diese Schutzbit-Maske legt fest, welche Zugriffsrechte die Dateien und Dateiverzeichnisse erhalten, die Sie ab jetzt in der aktuellen Shell oder in einer ihrer Subshells neu anlegen.

Wenn Sie mit *umask* die Schutzbit-Maske ändern, gilt diese Änderung so lange, bis Sie entweder mit *umask* einen neuen Wert vereinbaren oder die Shell beenden, in der Sie *umask* aufgerufen haben.

POSIX-Verwalter können mit *umask* den Wert der Schutzbit-Maske in der Datei */etc/profile* festlegen. Da die Datei */etc/profile* von jeder Login-Shell ausgeführt wird, gelten die so bestimmten Zugriffsrechte für jeden am System angemeldeten Benutzer.

Nähere Informationen zu den POSIX-Kommandos *chmod* und *umask* finden Sie im POSIX-Handbuch „Kommandos“ [1].

2.6.5 Zugangsschutz bei Zugang über einen fernen Rechner

POSIX kann auch von fernen Rechnern aus genutzt werden (siehe Abschnitt „Zugang zur POSIX-Shell“ auf Seite 63). Benutzer, die sich mit dem Kommando *rlogin* an POSIX anschließen, sind wie lokale Benutzer in der BS2000-Benutzerverwaltung des Zentralrechners eingetragen. Der BS2000-Baustein SRPM überprüft die Zugangsberechtigung während der *rlogin*-Verarbeitung.

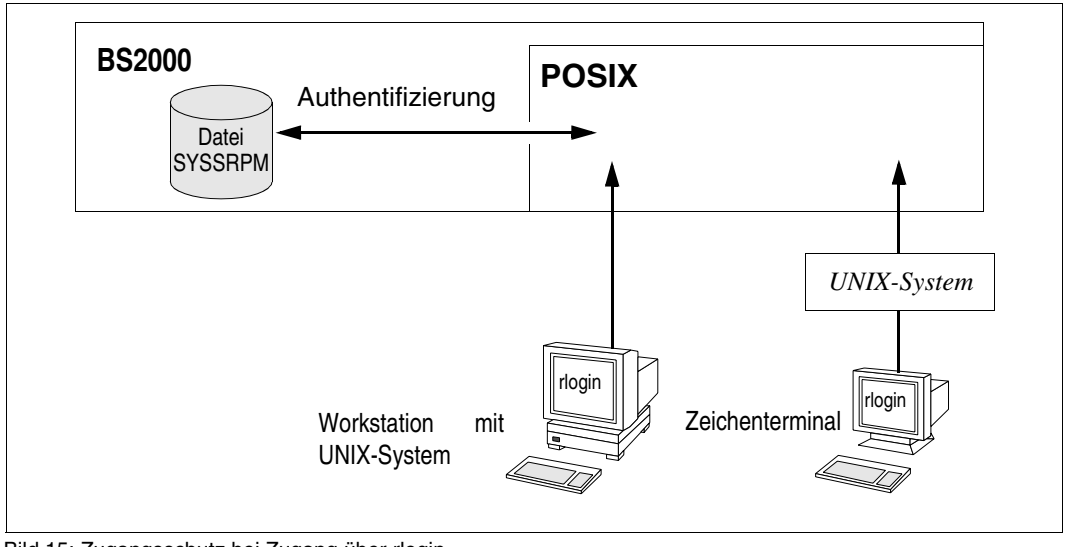

Bild 15: Zugangsschutz bei Zugang über rlogin

Für die Kommandos *rcp* und *rsh* gilt:

Die Zugangsberechtigungen werden wie in UNIX geregelt, d.h. zugelassene Rechner und Benutzer werden der Datei *$HOME/.rhosts* entnommen. Für SECOS lässt sich dies auch mit BS2000-Mitteln einstellen, siehe Abschnitt „Zugangsberechtigung für Benutzer eines fernen Rechners erteilen“ auf Seite 191.

ACHTUNG!
Einträge in der Datei */.rhosts* (im Root-Verzeichnis) erlauben die Ausführung von Kommandos unter TSOS und sind daher sicherheitskritisch!

3 Arbeiten mit POSIX

Dieses Kapitel wendet sich an alle POSIX-Benutzer. Es informiert Sie über die POSIX-Shell und die POSIX-Programmschnittstellen. Außerdem enthält es eine Beispielsitzung.

3.1 POSIX-Shell

Die POSIX-Shell ist die Schnittstelle, die Sie - über das C-Laufzeitsystem/Bibliotheken - mit dem Subsystem POSIX verbindet. Das folgende Bild zeigt die Struktur von POSIX im BS2000 und die Einbettung der POSIX-Shell.

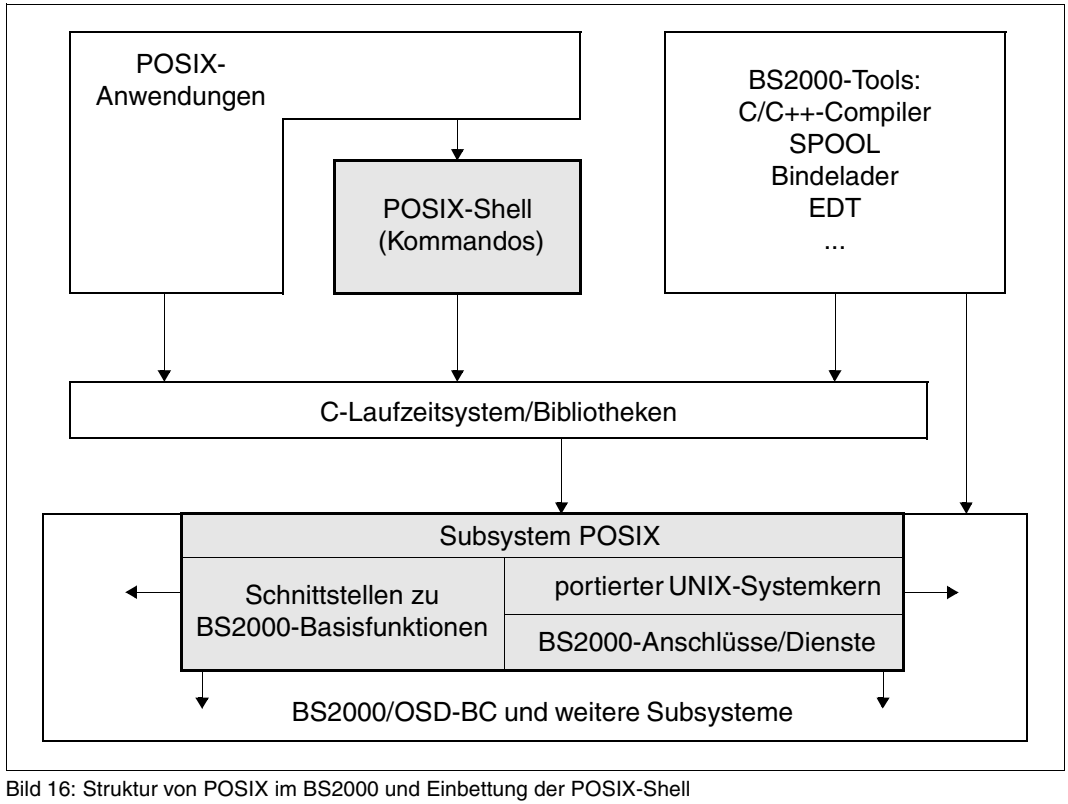

Bild 16: Struktur von POSIX im BS2000 und Einbettung der POSIX-Shell

Die POSIX-Shell ist eine Kommandoschnittstelle, die Sie zusätzlich zur Kommandoschnittstelle des BS2000/OSD verwenden können (siehe Bild 17).

Nach erfolgreichem Zugang zur POSIX-Shell (siehe Seite 63) stehen Ihnen alle Kommandos der POSIX-Shell zur Verfügung. Nach dem Verlassen der POSIX-Shell können Sie wieder BS2000-Kommandos eingeben.

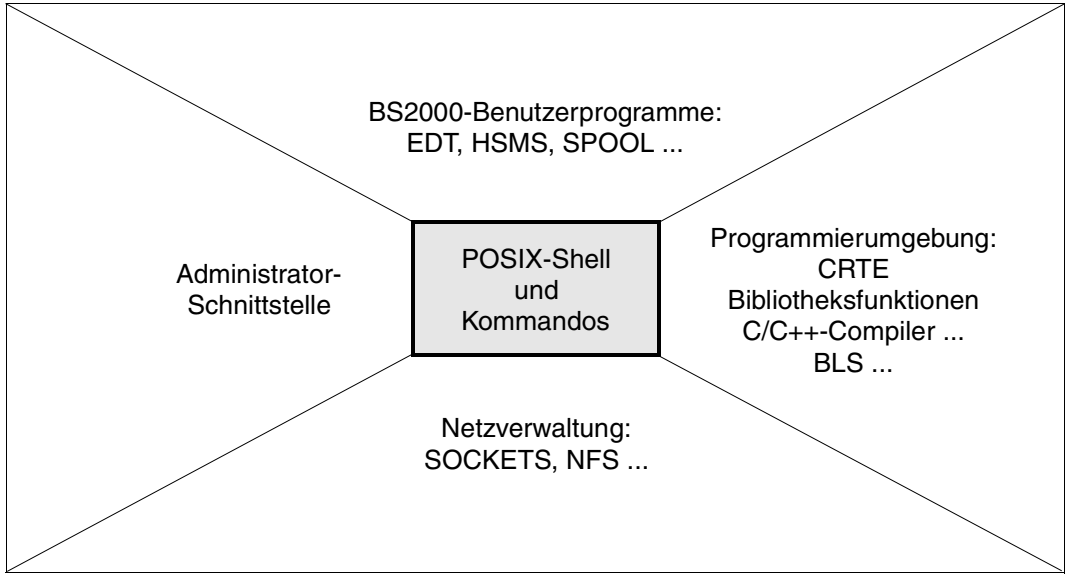

Bild 17: Die Kommandoebene POSIX-Shell im Subsystem POSIX

Die POSIX-Shell liest Kommandos von einer Datensichtstation oder aus einer POSIX-Datei, interpretiert sie nach bestimmten Regeln und sorgt für die Ausführung. Eine Datei, die Kommandos für die POSIX-Shell enthält, heißt Shell-Prozedur (Shell-Skript).

Die Bedienung und Leistung der POSIX-Shell hängen davon ab, ob die Datensichtstation, an der der Benutzer arbeitet, ein Block- oder Zeichenterminal ist.

Die POSIX-Shell bietet Ihnen eine umfangreiche Kommandosprache, die sich wie eine Programmiersprache anwenden lässt. Sie können mit den vorhandenen Kommandos eigene Programme erstellen und ohne vorheriges Übersetzen ausführen.

3.1.1 Zugang zur POSIX-Shell

Zur POSIX-Shell gibt es folgende Zugangsmöglichkeiten:

- über ein BS2000-Terminal (Blockterminal)
- von einem Terminal eines UNIX-Systems (Zeichenterminal)
- über eine Terminal-Emulation

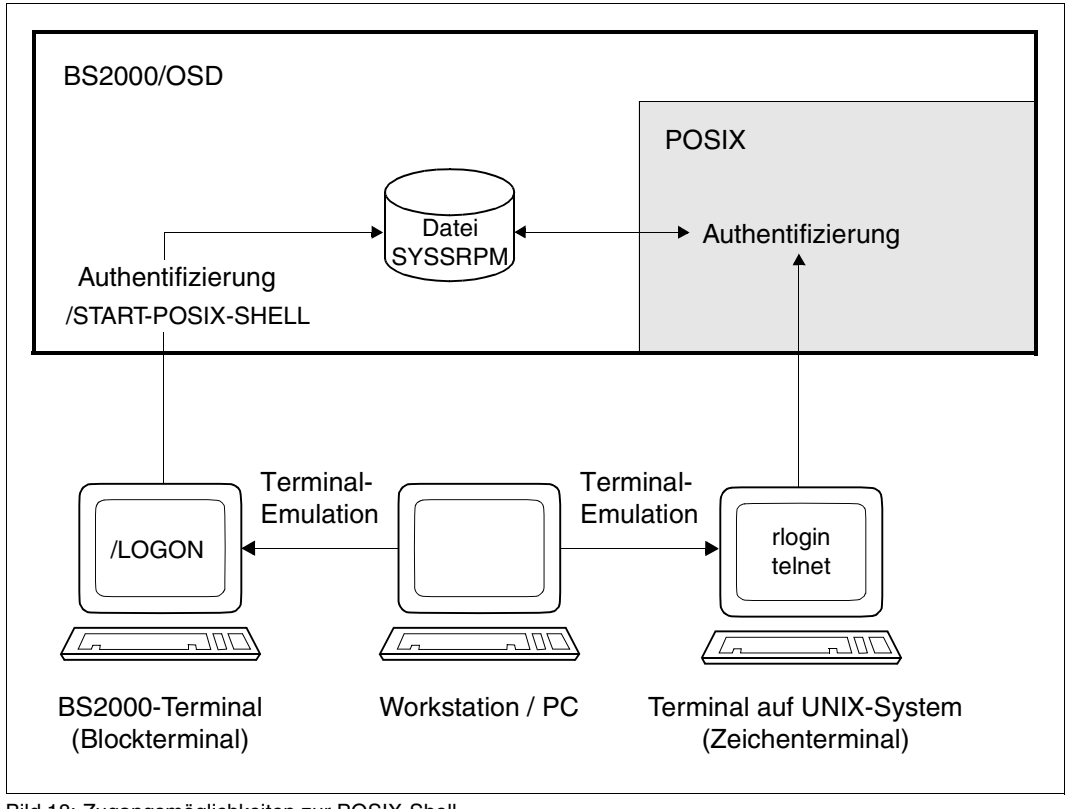

Bild 18: Zugangsmöglichkeiten zur POSIX-Shell

Zugang von einem Blockterminal

Jeder BS2000-Benutzer kann nach dem erfolgreichen BS2000-LOGON mit dem BS2000-Kommando /START-POSIX-SHELL (siehe Seite 263) die POSIX-Shell starten. Dieses Kommando besitzt keine POSIX-relevanten Operanden, da lediglich die POSIX-Umgebung aufgebaut und das Programm aufgerufen wird, das in der Datei SYSSRPM für den betreffenden Benutzer eingetragen ist (siehe Benutzerattribut „Programm“ im Abschnitt „POSIX-Benutzerattribute vergeben“ auf Seite 185).

Wenn ein Benutzer die POSIX-Shell als Standardprogramm in seinen Benutzerdaten eingetragen hat, kann er interaktiv mit der POSIX-Shell arbeiten, nachdem er das BS2000-Kommando /START-POSIX-SHELL eingegeben hat. Ihm stehen dann alle Kommandos und Funktionen der POSIX-Shell zur Verfügung. Für Interaktionen zwischen dem BS2000 und dem Subsystem POSIX gibt es in der POSIX-Shell eigene Kommandos (siehe Abschnitt „Kommandoumfang der POSIX-Shell“ auf Seite 270).

Eine Rückkehr ins BS2000 ist nur möglich, indem der Benutzer mit dem POSIX-Kommando *exit* die POSIX-Shell beendet.

Zugang von einem Zeichenterminal

Zugang über rlogin

Der Benutzer kann sich von einem Terminal auf einem UNIX-System mit dem Kommando *rlogin* an einem BS2000-Rechner anmelden, wenn der Benutzer für den BS2000-Rechner eine Zugangsberechtigung besitzt. Dazu benötigt er am BS2000-Rechner eine Benutzerkennung mit dem zugehörigen Kennwort. Nach dem Anmelden kann er POSIX wie im lokalen Modus benutzen.

Um sich an einem BS2000-Rechner anzuschließen, muss der Benutzer folgendes Kommando in der Shell eingeben:

```
rlogin <host> [-l <benutzerkennung>]
```

Wenn der Benutzer keine Benutzerkennung eingibt, wird die Kennung verwendet, unter der er am lokalen Rechner angemeldet ist. Beim *rlogin* wird für die gewünschte Benutzerkennung das Kennwort erfragt. Das Kennwort wird über die BS2000-Komponente SRPM (**S**ystem **R**esources and **P**rivileges **M**anagement) verifiziert: Die Angaben für die BS2000-Benutzerkennung und das Kennwort werden gegen die im Home-Pubset gesicherten Zugangskontrollattribute geprüft. Wenn Übereinstimmung besteht, erhält der Benutzer Zugang zum Subsystem POSIX. Wenn das Produkt SECOS im Einsatz ist, kann die Zugangskontrolle über die LOGON-Protection noch weiter verfeinert werden.

Für die Abrechnung eines Remote-Login-Systemlaufs muss eine Abrechnungsnummer vorhanden sein. Diese kann mit dem Operanden POSIX-RLOGIN-DEFAULT beim Kommando ADD-USER (siehe Seite 198) bzw. dem Kommando MODIFY-USER-ATTRIBUTES (Seite 237) festgelegt werden.

Beim Zugang über *rlogin* kann der Benutzer nur eingeschränkt auf BS2000-Kommandos zugreifen (z.B. wegen der fehlenden SYSFILE-Umgebung).

Zugang über telnet

Mit dem *telnet*-Dämon *telnetd* wird ein direkter Zugang zum BS2000 über das *telnet*-Protokoll sowohl vom UNIX-System aus als auch vom PC direkt über die *telnet*-Anwendung realisiert, die sich gegenüber POSIX wie ein Zeichenterminal verhält. Die Zugangskontrolle erfolgt in derselben Weise wie bei *rlogin*, d.h. über BS2000-Zugangsmechanismen. Der Zugang ohne Angabe eines Kennwortes, wie es zwischen UNIX-Systemen möglich ist (durch einen Eintrag in der *.rhosts*-Datei), wird nicht unterstützt.

Ein Parallelbetrieb von TELNET aus der Liefereinheit interNet Services (BS2000-TELNET) und TELNET aus POSIX ist ohne Konfigurationsänderungen nicht möglich, da in beiden Fällen die Standard-Portnummer 23 verwendet wird. Daher wird der Dämon *in.telnetd* in POSIX grundsätzlich nur gestartet, wenn in der Konfigurationsdatei des *inetd*-Dämonen (*/etc/inetd.conf*) das Kommentarzeichen '#' vor dem Eintrag *telnet* entfernt worden ist.

Wenn der BS2000-TELNET im System aktiv ist, sind für die Nutzung des TELNET aus POSIX zusätzlich folgende Schritte notwendig:

– Deaktivieren des BS2000-Batch-Jobs TELSR mit /STOP-TELNET-DEMON

– einige Minuten warten und den Dämon *in.telnetd* mit folgendem Kommando aktivieren:

```
kill -s HUP <pid von inetd-Daemon>
```

(die *pid* lässt sich z.B. ermitteln mit `ps -ef|grep inetd`)

Ein Parallelbetrieb beider TELNET-Server kann z.B. ermöglicht werden, indem der POSIX-TELNET-Server auf einem alternativen Port betrieben wird. Dazu ist ein neuer Dienst in die Datei */etc/services* einzutragen (z.B. "telnetx 1023/tcp"), in der Datei */etc/inet/inetd.conf* der Dienstname "telnet" nach "telnetx" zu ändern und das Kommentarzeichen am Anfang dieser Zeile zu entfernen. Danach muss mit "kill -s HUP <pid>" die Rekonfiguration des *inetd* gestartet werden. Die TELNET-Client-Programme können dann unter Verwendung der alternativen Portnummer (im Beispiel 1023) auf den POSIX-TELNET-Server und mit der Standardportnummer (23) auf den BS2000-TELNET-Server zugreifen.

Zugang über eine Terminal-Emulation

Die dritte Zugangsmöglichkeit besteht über eine Terminal-Emulation. Dazu meldet sich der Benutzer an einer Workstation oder an einem PC an. Anschließend startet er eine Terminal-Emulation, wobei entweder ein Terminal eines UNIX-Systems oder ein BS2000-Blockterminal emuliert werden muss:

BS2000-Terminal-Emulation

Beim Zugang über eine BS2000-Terminal-Emulation wie z.B. EM9750 oder MT9750 steht dem Benutzer ein Terminal in der Art eines Blockterminals zur Verfügung. Der Benutzer muss sich wie im BS2000 üblich authentisieren und kann danach BS2000-Kommandos und /START-POSIX-SHELL wie über ein BS2000-Terminal eingeben (siehe Seite 64).

Terminal-Emulation für UNIX-Systeme

Terminal-Emulationen für UNIX-Systeme stehen für Workstations mit UNIX-System und grafischen, OSF/Motif-basierten Oberflächen und für Windows-PCs zur Verfügung (z.B. EM97801, SINIX-TE). Dabei wird ein Zeichentterminal eines UNIX-Systems emuliert und der Benutzer kann Kommandos wie z.B. *rlogin* eingeben (siehe Seite 64).

3.1.2 Besonderheiten für das Arbeiten mit der POSIX-Shell

Voreinstellungen in der Benutzerumgebung

Nach erfolgreichem Zugang zum Subsystem POSIX wird die POSIX-Shell gestartet. Bevor die POSIX-Shell ihr Bereitzeichen ausgibt, werden folgende Voreinstellungen in der Benutzerumgebung getroffen:

- Die POSIX-Shell initialisiert die Standard-Shell-Variablen. Sie weist den folgenden Variablen ihre Standardwerte zu:
  HOME, LANG, LOGNAME, MAIL, PATH, PS1, PS2, PS3, PS4, SHELL, TTY, TERM, TZ und USER.
  Falls eine Variable <x> durch die BS2000-S-Variable SYSPOSIX.<x> bereits definiert ist, wird dieser Wert genommen. Es dürfen aber nicht die Shell-Variablen USER, TERM, TYP, LOGNAME, HZ, HOME und MAIL vom Anwender gesetzt werden.
- Die Datei */etc/profile* wird ausgeführt.
- Die Datei *$HOME/.profile* wird ausgeführt, falls Sie diese Datei angelegt haben.

Sonderfunktionen (P-Tasten, Ctrl-Taste)

Sie können die P-Tasten P3, P4 und P5 durch den Aufruf des POSIX-Kommandos *bs2pkey* folgendermaßen belegen:

P3 mit @@c ([CTRL] C)
P4 mit @@d ([CTRL] D)
P5 mit @@z ([CTRL] Z)

Das Programm wird entweder in der POSIX-Shell (ohne Optionen) aufgerufen oder kann in die Datei */etc/profile* aufgenommen werden. Dann wird das Programm bei jedem Aufruf der Shell aktiviert.

3.1.3 POSIX-Lader

Der POSIX-Lader ist Bestandteil des Subsystems POSIX. Er verwaltet systemglobale, benutzer- oder sitzungsspezifische Programm-Caches variabler Größe, in denen ablaufbereite Core-Images von POSIX-Programmen bereitgestellt und zur Ausführung in den Speicher kopiert werden. Mit Hilfe des POSIX-Laders können Ladevorgänge in POSIX wesentlich beschleunigt werden. Eine ausführliche Beschreibung des POSIX-Laders finden Sie im Abschnitt „POSIX-Lader“ auf Seite 158.

Programm-Caches einrichten

Beim Start von POSIX sind noch keine Programm-Caches eingerichtet. Der globale Programm-Cache wird entsprechend den Angaben in der POSIX-Informationsdatei implizit angelegt. Der globale Programm-Cache kann auch nachträglich vom Superuser mit dem *posdbl*-Kommando eingerichtet werden. Die benutzer- und sitzungsspezifischen Programm-Caches kann sich der jeweilige Benutzer mit dem Kommando *pdbl* einrichten.

Aufnahme in einen Programm-Cache

Nachdem die Programm-Caches eingerichtet worden sind, können POSIX-Programme auf folgende Weise in einen Programm-Cache gelangen:

– Implizit: Beim ersten Aufruf eines (nicht builtin-) POSIX-Kommandos (Basis-Shell und erweiterte Shell) wird das Programm über BLS geladen. Sein Core-Image wird im globalen Programm-Cache abgelegt, in den Speicher geladen und gestartet.

– Explizit: POSIX-Programme können im globalen Programm-Cache (durch den Superuser mit dem Kommando *posdbl*) oder im benutzer-/sitzungsspezifischen Programm-Cache (durch den Benutzer mit dem Kommando *pdbl*) abgelegt werden.

Ladevorgang beim Programmaufruf

Bei jedem Aufruf eines (nicht builtin-) POSIX-Kommandos über den Systemaufruf *exec* werden vorrangig folgende Bedingungen in der angegebenen Reihenfolge geprüft:

– Ist die Task eine durch *fork* erzeugte Task?
– Wird das Programm nicht im Debug-Modus gestartet?
– Ist das entsprechende Core-Image in einem Programm-Cache gespeichert?

Ist eine der Bedingungen nicht erfüllt, wird das Programm „klassisch“ über BLS geladen und gestartet. Ansonsten wird das in einem Programm-Cache gespeicherte Programm direkt in den Speicher kopiert und gestartet. Die Umgehung von BLS bewirkt, dass das Programm nur unvollständig in die BS2000-Umgebung eingebettet ist. Die damit verbundenen Einschränkungen sind im Abschnitt „Ladevorgang“ auf Seite 163 aufgeführt.

3.1.4 Kommandos von der POSIX-Shell aus eingeben

Nach erfolgreichem Zugang zum Subsystem POSIX wird die POSIX-Shell gestartet.

Wenn Sie die POSIX-Shell interaktiv benutzen, gibt die POSIX-Shell den Wert der Umgebungsvariablen PS1 als Bereitzeichen aus, bevor sie ein Kommando einliest. Im Standardfall ist dies das Dollarzeichen ($) bzw. (#) für einen privilegierten Benutzer und ein anschließendes Leerzeichen (␣).

Die Kommandoeingabe hat folgendes Format:

`kommando[␣optionen][␣parameter]␣ ...`↵

Bei *kommando* müssen Sie den Namen eines POSIX-Kommandos oder einer Shell-Prozedur angeben, die ausgeführt werden soll. Mit *optionen* geben Sie Steueranweisungen zur Kommandoausführung. Bei *parameter* müssen Sie ein Aufrufargument eingeben, das die POSIX-Shell an *kommando* übergibt. Abhängig vom Kommando können Sie auch mehrere Aufrufargumente angeben.

Den Kommandonamen und die Aufrufargumente müssen Sie bei Zeichenterminals durch Tabulator- oder Leerzeichen voneinander trennen. Das letzte Aufrufargument und damit die Eingabe des Kommandos schließen Sie bei Zeichenterminals durch ↵ und bei Blockterminals durch EM DUE ab.

Das Starten von reinen BS2000-Programmen aus der POSIX-Shell wird nicht unterstützt.

Wenn die Bildschirmzeile für eine Eingabe zu kurz ist, haben Sie zwei Möglichkeiten:

- Sie schreiben am Zeilenende einfach weiter, ohne die Taste ↵ zu drücken. Nachdem Sie das Kommando vollständig eingegeben haben, schließen Sie es mit der Taste ↵ ab.
- Sie setzen die Zeile mit \ ↵ fort. Das Zeichen Gegenschrägstrich (\) entwertet die Kommandoabschlussfunktion der Taste ↵. Anschließend können Sie die Kommandoeingabe fortsetzen. Nach dem Drücken von ↵ (ohne \) wird das Kommando ausgeführt.

Jedes POSIX-Kommando gibt an die POSIX-Shell, in der es aufgerufen wurde, einen Wert zurück, nämlich seinen Endestatus. Dieser Wert ist bei einem fehlerfreien Ablauf 0, bei einem fehlerhaften Ablauf ungleich 0.

Wenn ein Kommando Informationen auf den Bildschirm ausgibt und die Ausgabe größer als eine Bildschirmseite ist, können Sie bei Zeichenterminals die Ausgabe durch Drücken der Tasten CTRL s anhalten und anschließend mit CTRL q fortsetzen. Bei Blockterminals wird diese Funktion nicht unterstützt.

Ausführliche Informationen zur Eingabe von Kommandos finden Sie im POSIX-Handbuch „Kommandos“ [1].

3.1.5 Kommandos für große POSIX-Dateien

Sie können große POSIX-Dateien (> 2Gbyte) nur mit solchen Kommandos bearbeiten, die auch dafür geeignet sind (= large file aware). Eine Reihe von Kommandos erkennen zwar große Dateien, weisen eine Bearbeitung aber zurück (= large file safe). Dateibearbeitungskommandos, die weder *large file safe* noch *large file aware* sind, sollten Sie nicht zur Bearbeitung von großen Dateien verwenden.

Die folgende Tabelle zeigt, welche Kommandos *large file aware* und *large file safe* sind (in alphabetische Reihenfolge):

Eigenschaft	**Kommandos**
large file aware	awk, bs2cp, cat, cd, chgrp, chmod, cmp, chown, cksum, compress, cp, cpio, df, diff, du, file, find, getconf, grep, hd, head, iconv, join, ln, ls, mkfifo, mknod, more, mv, pax, rcp, rm, rmdir, sh, split, sum, tail, tar, touch, tr, ulimit, uncompress, wc, zcat
large file safe	ar, comm, csplit, cut, dd, edt, edtu, egrep, expand, fgrep, fold, nl, od, paste, sort, strings, unexpand, vi

3.2 POSIX-Programmschnittstellen

Die POSIX-Programmschnittstellen stehen zusammen mit den BS2000-Programmschnittstellen zur Verfügung. Deshalb sind reine BS2000-Programme, reine POSIX-Programme und gemischte Programme ablauffähig. Gemischte Programme enthalten sowohl BS2000- als auch POSIX-Programmschnittstellen. Für gemischte Programme bestehen einige Einschränkungen (siehe Seite 72).

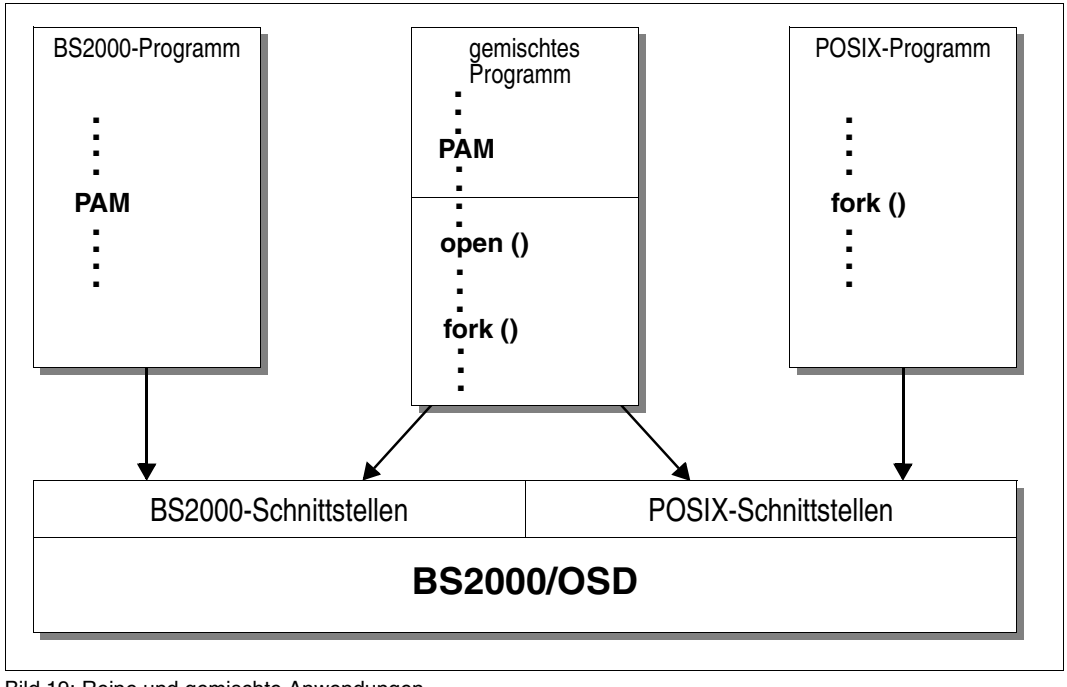

Bild 19: Reine und gemischte Anwendungen

Die POSIX-Programmschnittstellen enthalten C-Bibliotheksfunktionen mit BS2000- und POSIX-Funktionalität. Programme, die auf anderen Plattformen (UNIX, Windows, ...) gemäß dem XPG4-Standard geschrieben wurden, müssen lediglich neu übersetzt werden, um im BS2000 ablaufen zu können.

Auf POSIX-Dateien kann über reine oder gemischte Programme zugegriffen werden. Darüber hinaus können POSIX-Dateien von einigen BS2000-Softwareprodukten wie EDT und HSMS verarbeitet werden (siehe Kapitel „BS2000-Softwareprodukte im Umfeld von POSIX“ auf Seite 85).

3.2.1 Einschränkungen für Programme mit gemischter Funktionalität

Programme, die POSIX-Schnittstellen nutzen, werden analog zu Programmen mit reinen BS2000-Schnittstellen behandelt. Einschränkungen ergeben sich dort, wo ein Prozess mit einem *fork*-Aufruf erzeugt wird und ihm seine BS2000-Umgebung nicht mitvererbt wird.

Zwischen folgenden Aufrufen eines gemischten Programms muss unterschieden werden:

1. Logon-Prozess (Dialog-Task), der nicht durch einen *fork*-Aufruf erzeugt wurde

   BS2000- und POSIX-Programmschnittstellen können beliebig gemischt werden.

2. Prozess, der durch einen *fork*-Aufruf aus einem Logon-Prozess erzeugt wurde

   Die SYSFILE-Umgebung wird nicht weitervererbt, weil generell keine vom Vaterprozess geöffneten BS2000-Dateien an den Sohnprozess vererbt werden.
   Die SYSFILE-Umgebung ist je eine anwenderspezifische Systemdatei zur Kommandoeingabe (SYSCMD), zur Dateneingabe (SYSDTA), zum Logging (SYSLST) und zur Meldungs- bzw. Datenausgabe (SYSOUT).

   Eine Mischung von BS2000- und POSIX-Programmschnittstellen ist in folgendem Umfang erlaubt:

   - Parallele Ausgabe über POSIX-Mechanismen und WROUT erlaubt
   - Keine Eingabe über RDATA möglich
   - Kein Checkpoint/Restart möglich
   - Kein *BKPT* möglich
   - Kein *fork* möglich, wenn DIV- oder FASTPAM-Bereiche existieren

   Ansonsten können die BS2000- und POSIX-Programmschnittstellen beliebig gemischt werden.

3. Gemischtes Programm, das aus der POSIX-Shell gestartet wird

   Ein gemischtes Programm, das aus der POSIX-Shell gestartet wird, hat eine andere SYSFILE-Umgebung als die POSIX-Shell, da es durch einen *fork*-Aufruf erzeugt wurde. Es gilt dasselbe wie bei 2.

3.2.2 Einschränkungen für Makroaufrufe

Jeder Prozess, der durch einen Aufruf der Funktion *fork* erzeugt wurde, besitzt eine SYSFILE-Umgebung, aber keine Systemdatei SYSCMD (ausgenommen *rlogin*-Sessions). Deshalb werden Zugriffe auf die Systemdatei SYSCMD mit einem Returncode zurückgewiesen. Ansonsten können BS2000-Kommandos per CMD-Makro eingegeben werden, wobei folgendes zu beachten ist:

- Ein CMD-Makroaufruf erfordert eine nachträgliche SDF-Initialisierung der durch *fork* erzeugten Task. Dadurch verschlechtert sich die Performance.
- Folgende Kommandos sind in einem CMD-Makroaufruf nicht zugelassen:
  - Alle Kommandos, die bereits in BS2000/OSD-BC nicht zugelassen sind (siehe Handbuch „Makroaufrufe an den Ablaufteil“ [30]); u.a. ist das Verbot von /HOLD-PROGRAM zu beachten.
  - Kommandos, die auf die Systemdatei SYSCMD zugreifen, z.B. das %TRACE-Kommando von AID

Die BS2000-Kommandos /EXIT-JOB und /LOGOFF führen wie in BS2000/OSD-BC zur Beendigung des Tasks und zur Rückkehr zum Vaterprozess.

3.2.3 Vererbung

Bei einem *fork*-Aufruf werden nur POSIX-Ressourcen vererbt. Deshalb sind POSIX-Dateien, die der Vaterprozess eröffnet hat, auch in Sohnprozessen offen. Dagegen sind BS2000-Dateien, die im Vaterprozess geöffnet wurden, nicht offen.

Der Klasse-6-Speicher eines Programms wird vollständig vererbt. Beim Klasse-5-Speicher werden nur die vorher als vererbbar markierten Seiten vererbt.

Eine Quasi-Vererbung von BS2000-Ressourcen ist programmgesteuert möglich, indem man im Vaterprozess diese Ressourcen (z.B. BS2000-Dateien) mehrfach benutzbar öffnet. Die Information über diese Ressourcen kann dann über eine privat zu definierende Datenstruktur dem Sohnprozess, in dem ja dasselbe Programm wie im Vaterprozess abläuft, mitgegeben werden. Der Sohnprozess kann sich dann an diese Ressourcen wieder explizit anschließen.

3.3 Beispielsitzung

In diesem Abschnitt finden Sie ein Beispiel für das Arbeiten mit der POSIX-Shell. Sie melden sich an das BS2000 an, lassen sich das Inhaltsverzeichnis Ihrer Benutzerkennung ausgeben und starten dann die POSIX-Shell.
In der POSIX-Shell erstellen Sie zuerst eine *.profile*-Datei, in der Sie zur Arbeitsvereinfachung Aliasvariablen und zur besseren Orientierung ein neues Bereitzeichen definieren, das den jeweils aktuellen Pfad ausgibt. Nach der Ausführung der *.profile*-Datei sind die dort getroffenen Definitionen wirksam.
Anschließend übertragen Sie eine Datei des BS2000-Dateisystems in das POSIX-Dateisystem und bearbeiten sie dort.

```
/set-logon-parameters user-id=user1,account=...  ————————————————————  (1)
/show-file-attributes  ——————————————————————————————————————————————  (2)
%     114 :1OSN:$USER1.ANHANG.V2
%       3 :1OSN:$USER1.AVASQUER
%      78 :1OSN:$USER1.BIB.EXAMPLES.SDF
%       6 :1OSN:$USER1.DO.MSGCHECK
%    5007 :1OSN:$USER1.FS.USER1
%       3 :1OSN:$USER1.MSG.PROT
%       3 :1OSN:$USER1.OUTPUT
%       3 :1OSN:$USER1.PROG.C
%       3 :1OSN:$USER1.SYS.SDF.LOGON.USERPROC
/start-posix-shell  ——————————————————————————————————————————————————  (3)
POSIX Basisshell 09.0A43 created Feb 20 2012
POSIX Shell 08.0A43 created Aug 02 2011
Copyright (C) Fujitsu Technology Solutions 2009
              All Rights reserved
Last login: Sun Jul 22 19:42:55 2012 on term/001  ————————————————————  (4)
$ edtu .profile  ———————————————————————————————————————————————————  (5)
```

(1) Melden Sie sich in gewohnter Weise an BS2000 an.

(2) Lassen Sie sich mit dem BS2000-Kommando /SHOW-FILE-ATTRIBUTES das Inhaltsverzeichnis Ihrer Benutzerkennung ausgeben.

(3) Rufen Sie die POSIX-Shell mit dem BS2000-Kommando /START-POSIX-SHELL auf.

(4) Sie sind als POSIX-Shell-Benutzer akzeptiert.

(5) Erzeugen Sie die Datei *.profile* mit dem POSIX-Editor *edtu*.
Da die Datei noch nicht vorhanden ist, legt *edtu* eine neue Datei an (siehe Seite 75).

```
   1.00 alias ll='ls -l'
   2.00 alias la='ls -al'
   3.00 PS1='$PWD> '
   4.00
   5.00
   6.00
   7.00
   8.00
   9.00
  10.00
  11.00
  12.00
  13.00
  14.00
  15.00
  16.00
  17.00
  18.00
  19.00
  20.00
  21.00
  22.00
            POSIX editor ready for file .profile: new file
return                                                    0000.00:001(0)
LTG     EM:1                                             TAST
```

```
$ . .profile ──────────────────────────────────────────────────────────  (6)
/home/user1> la ───────────────────────────────────────────────────────  (7)
total 84
drwxr-xr-x   5 USER1    USROTHER    2048 Dec 22 14:03 .
drwxr-xr-x  63 SYSROOT  POSSYS      2048 Dec 22 06:35 ..
-rw-r--r--   1 USER1    USROTHER      48 Dec 22 14:02 .profile
-rw-------   1 USER1    USROTHER    2576 Dec 22 14:06 .sh_history
drwxr-xr-x   2 USER1    USROTHER    2048 Dec 15 17:18 c-source
drwxr-xr-x   2 USER1    USROTHER    8192 Dec  5 13:47 lost+found
-rw-r--r--   1 USER1    USROTHER      94 Dec 21 14:02 letter1
drwxr-xr-x   2 USER1    USROTHER    2048 Dec 19 15:05 test
  ...
/home/user1> cd c-source ──────────────────────────────────────────────  (8)
```

(6) Nachdem Sie die *.profile*-Datei mit *edtu* erstellt und den Editor mit dem Kommando *return* wieder beendet haben, soll die *.profile*-Datei in der aktuellen Shell ausgewertet werden. Dazu geben Sie das Punkt-Kommando `. .profile` ein.

(7) Die POSIX-Shell meldet sich mit dem neu definierten Bereitzeichen, das den aktuellen Pfad */home/user1* ausgibt. Sie lassen sich das Inhaltsverzeichnis mit allen Dateien über das mit dem Aliasnamen *la* definierte Kommando anzeigen.

(8) Wechseln Sie in das Unterverzeichnis *c-source*, in dem Sie beispielsweise Ihre C-Programme speichern.

```
/home/user1/c-source> bs2cp bs2:prog.c prog.c ———————————————————— (9)
/home/user1/c-source> la
total 60
drwxr-xr-x  2  USER1      USER1GRP     2048   Jul 6 .
drwxr-xr-x  2  USER1      USER1GRP     2048   Jul 6 ..
-rw-r--r--  1  USER1      USER1GRP     2048   Jul 6 prog.c
/home/user1/c-source> cat prog.c ———————————————————————————— (10)
#include <stdio.h>
main()
{
  printf("hello world\n");
  return(0);
}
/home/user1/c-source> cc -o prog prog.c ————————————————————— (11)
/home/user1/c-source> prog ——————————————————————————————— (12)
hello world
/home/user1/c-source> exit ——————————————————————————————— (13)
.... ————————————————————————————————————————————————— (14)
/exit-job ————————————————————————————————————————————— (15)
```

(9) Kopieren Sie die im BS2000-Dateisystem liegende Datei *prog.c* in das POSIX-Dateisystem. Die Datei wird in das aktuelle Verzeichnis *c-source* geschrieben.

(10) Lassen Sie sich den Inhalt der Datei *prog.c* mit *cat* ausgeben.

(11) Übersetzen Sie die Datei *prog.c* mit dem C-Compiler. Das Ergebnis des Übersetzungslaufs soll in die Datei *prog* geschrieben werden.

(12) Lassen Sie das Programm *prog* ablaufen. Es gibt die Zeichenfolge „hello world“ auf dem Bildschirm aus.

(13) Beenden Sie mit dem Kommando *exit* die POSIX-Shell.

(14) Eingabe von weiteren BS2000-Kommandos, falls gewünscht.

(15) Melden Sie sich am BS2000 ab.

3.4 Programmschnittstelle für große POSIX-Dateien

Dieses Unterkapitel beschreibt, wie Sie neue Programme zur Behandlung großer Dateien erstellen und wie Sie bestehende Programme ändern müssen, um auf große Dateien zugreifen zu können.

3.4.1 Neue Programme erstellen

Wenn Sie neue Programme erstellen, die auf große POSIX-Dateien zugreifen, dann müssen Sie Folgendes beachten:

- Setzen Sie vor der ersten Include-Anweisung folgende Define-Anweisung:

  ```
  #define _LARGEFILE64_SOURCE   1
  ```

- Geben Sie den Header *unistd.h* als erste Include-Anweisung an.
  Damit stehen dem Programm die notwendigen Schnittstellen und Datentypen zur Verfügung.

- Benutzen Sie die 64-bit-Funktionen, um auf große POSIX-Dateien zuzugreifen. D.h. Sie verwenden einfach *open64()*, *lseek64()*, ... an Stelle von der gewohnten Funktionen *open()*, *lseek()*,... .

  Die folgende Liste gibt eine Überblick über alle 64-bit-Funktionen:

creat64()	fstat64()	lseek64()	stat64()
fgetpos64()	fstatvfs64()	lstat64()	statvf64()s
fopen64()	ftell64()	mmap64()	statvfs64()
freopen64()	ftruncate64()	open64()	truncate64()
fseek64()	getdents64()	readdir64()	
fsetpos64()	getrlimit64()	setrlimit64()	

- Verwenden Sie im Programm die 64-bit-Datentypen an Stelle der 32-bit-Datentypen, z.B. *off64_t* (64-Bit) statt *off_t* (32-Bit). Diese Datentypen sind in der Include-Datei *sys/types.h* definiert. Dies ermöglicht es, in einem Programm die 64-bit- und die 32-bit-Schnittstellen parallel zu verwenden, z.B. *lseek()* und *lseek64()*. Damit wird die Migration von Programmen erleichtert.

3.4.2 Existierende Programme an große Dateien anpassen

Wenn Sie existierende Programme für den Zugriff auf große Dateien anpassen möchten, dann müssen Sie dasselbe tun wie in Abschnitt „Neue Programme erstellen“ auf Seite 77 beschrieben, d.h. die zugehörigen Define-Anweisungen sowie die Include-Datei *unistd.h* hinzufügen, 32-bit-Funktionen (*open()*, *lseek()*, ...) durch die entsprechenden 64-Bit-Funktionen *open64()*, *lseek64()*, ... ersetzen und ggf. die 64-bit-Datentypen verwenden. Dies hat unter Umständen zur Folge, dass interne Datenstrukturen inkompatibel geändert werden müssen.

Um die Problematik des Zugriffs auf große Dateien zu verdeutlichen, wird im Folgenden das kleines Beispielprogramm *prog32* vorgestellt. Dieses Programm wird anschließend so umgestellt, dass es auf große Dateien zugreifen kann (Beispielprogramm *prog64* auf Seite 80).

Beispielprogramm prog32

prog32 öffnet eine vorgegebene Datei und gibt von dieser Datei maximal 32 Bytes ab einer vorgegebenen Stelle aus. Zur leichteren Lesbarkeit wurde die Prüfung der Eingabeparameter weggelassen.

```
/*
** prog.c
**
** 1. Parameter: Name der zu lesenden Datei
** 2. Parameter: Offset der zu lesenden Daten innerhalb der Datei
**
*/

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/fcntl.h>

#define BUFFER_LENGTH   8192
#define READ_LENGTH     32

char buffer[BUFFER_LENGTH];

int
main(int argc, char *argv[]) {
int fd;
int len;
int i;
```

```
off_t filelen;
off_t offset_in_file;

   offset_in_file = atol(argv[2]);
   printf ("reading from file <%s> with offset %d and length %d\n", argv[1],
         offset_in_file, READ_LENGTH);

   /* open file */
   if ((fd = open (argv[1], O_RDONLY)) < 0) {

      printf ("open not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   /* now get the length of the file */
   if ((filelen = lseek(fd, (off_t)0, SEEK_END)) == (off_t)(-1)) {

      printf ("lseek to end of file not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   if (offset_in_file > filelen) {

      printf ("offset %d is greater than filelength %d, termination\n",
            offset_in_file, filelen);
      exit(EXIT_FAILURE);
   }

   /* now seek to the offset to be read */
   if (lseek(fd, offset_in_file - filelen, SEEK_CUR) == (off_t)(-1)) {

      printf ("lseek not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   /* read the data */
   if ((len = read (fd, &buffer[0], READ_LENGTH)) <= 0 ) {

      printf ("read not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }
   else {
      /* now print the data that were read in hexadecimal form */
      printf ("data of size %d (expected %d) were read\n", len, READ_LENGTH);
```

```
      for (i = 0; i < len; i++) {

         printf ("%02X ", buffer[i]);
      }
      printf ("\n");
   }

   if (close(fd) != 0) {

      printf ("close not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }
}
```

Beispielprogramm prog64

prog64 wird aus *prog32* abgeleitet und so umgestellt, dass es auf große Dateien zugreifen kann. Die Zeilen, in denen Änderungen gegenüber *prog32* vorgenommen wurden,sind **fett** hervorgehoben:

```
/*
** prog.c
**
** 1. Parameter: Name der zu lesenden Datei
** 2. Parameter: Offset der zu lesenden Daten innerhalb der Datei
**
*/

#define _LARGEFILE64_SOURCE   1

#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/fcntl.h>

#define BUFFER_LENGTH        8192
#define READ_LENGTH          32

char buffer[BUFFER_LENGTH];

int
main(int argc, char *argv[]) {
int fd;
int len;
int i;
```

```
off64_t filelen;
off64_t offset_in_file;

   offset_in_file = atoll(argv[2]);
   printf ("reading from file <%s> with offset %lld and length %ld\n",
             argv[1], offset_in_file, READ_LENGTH);

   /* open file */
   if ((fd = open64 (argv[1], O_RDONLY)) < 0) {

      printf ("open not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   /* now get the length of the file */
   if ((filelen = lseek64(fd, (off64_t)0, SEEK_END)) == (off64_t)(-1)) {

      printf ("lseek to end of file not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   if (offset_in_file > filelen) {

      printf ("offset %lld is greater than filelength %lld, termination\n",
            offset_in_file, filelen);
      exit(EXIT_FAILURE);
   }

   /* now seek to the offset to be read */
   if (lseek64(fd, offset_in_file - filelen, SEEK_CUR) == (off64_t)(-1)) {

      printf ("lseek not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }

   /* read the data */
   if ((len = read (fd, &buffer[0], READ_LENGTH)) <= 0 ) {

      printf ("read not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }
   else {
      /* now print the data that were read in hexadecimal form */
```

```
      printf ("data of size %d (expected %d) were read\n", len, READ_LENGTH);

      for (i = 0; i < len; i++) {

         printf ("%02X ", buffer[i]);
      }
      printf ("\n");
   }

   if (close(fd) != 0) {

      printf ("close not successful, termination\n");
      perror("ERRNO SET");
      exit(EXIT_FAILURE);
   }
}
```

Anwendung von prog32 und prog64

Die beiden Programme wurden auf folgende Dateien im Verzeichnis */mnt33/dir1* angewandt:

```
$ ls -l /mnt33/dir1
total 10245088
-rw-r--r--   1 BACH     OS315    2621440000 Feb 27 18:26 bigfile1
-rw-r--r--   1 BACH     OS315    2621440000 Mar  6 15:06 bigfile2
-rw-r--r--   1 BACH     OS315         18 Mar 15 13:56 smallfile1
-rw-r--r--   1 BACH     OS315         26 Mar 15 13:57 smallfile2
drwxr-xr-x   2 BACH     OS315       2048 Mar 15 13:58 subdir1
drwxr-xr-x   2 BACH     OS315       2048 Mar 15 13:58 subdir2
```

Ergebnisse einiger Programmläufe:

```
$ prog32 /mnt33/dir1/smallfile1  128
reading from file </mnt33/dir1/smallfile1> with offset 128 and length 32
offset 128 is greater than filelength 18, termination

$ prog32 /mnt33/dir1/bigfile1  128  28
reading from file </mnt33/dir1/bigfile1> with offset 128 and length 32
lseek to end of file not successful, termination
ERRNO SET: Value too large for defined data type


$ prog64 /mnt33/dir1/smallfile1  128
reading from file </mnt33/dir1/smallfile1> with offset 128 and length 32
offset 128 is greater than filelength 18, termination
```

```
$ prog64 /mnt33/dir1/smallfile1  10 28
reading from file </mnt33/dir1/smallfile1> with offset 10 and length 32
data of size 8 (expected 32) were read
91 A5 A2 A5 A2 A5 95 15


$ prog64 /mnt33/dir1/bigfile1  128  0
reading from file </mnt33/dir1/bigfile1> with offset 128 and length 32
data of size 32 (expected 32) were read
F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3 F3
F3 F3 F3 F3 F3 F3


$ prog64 /mnt33/dir1/bigfile1  2500000000
reading from file </mnt33/dir1/bigfile1> with offset 2500000000 and length 32
data of size 32 (expected 32) were read
F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1 F1
F1 F1 F1 F1 F1 F1


$ prog64 /mnt33/dir1/bigfile1  500000000000
reading from file </mnt33/dir1/bigfile1> with offset 5000000000 and length 32
offset 5000000000 is greater than filelength 2621440000, termination
```

4 BS2000-Softwareprodukte im Umfeld von POSIX

Dieses Kapitel wendet sich an alle POSIX-Benutzer. Es gibt Ihnen einen kurzen Überblick über die Softwareprodukte im Umfeld von POSIX. Die Abschnitte „Binder-Lader-System“ auf Seite 85 bis „SORT“ auf Seite 98 behandeln BS2000-Produkte, die an POSIX angepasst sind und mit POSIX-Dateien arbeiten können. Ab Abschnitt „interNet Services“ auf Seite 99 werden Produkte vorgestellt, die mit POSIX ins BS2000/OSD portiert wurden. Die Menge der portierten Produkte wächst ständig.

4.1 Binder-Lader-System

Zum Binden und Laden ausführbarer Programme verwendet POSIX das Binder-Lader-System des BS2000/OSD, dessen wichtigste Bestandteile der Binder BINDER und der dynamische Bindelader DBL sind (DBL ist Teil von BLSSERV). Das Binder-Lader-System wird von POSIX immer dann aufgerufen, wenn ein POSIX-Benutzer ein Programm aufruft oder wenn er beim Übersetzen eines Quellprogrammes ein ausführbares Programm als Ergebnis anfordert.
Eine externe Benutzerschnittstelle, wie sie BINDER im BS2000/OSD hat, ist in der POSIX-Umgebung nicht verfügbar. Für das Binden und Laden von Programmen in der POSIX-Umgebung gelten die Konventionen, die durch die Compiler und durch POSIX-übliche Programmaufrufe festgelegt sind.

Wenn in der POSIX-Umgebung Benutzerprogramme geladen werden, die noch unbefriedigte Externverweise („unresolved externals“) enthalten, werden Meldungen des Binder-Lader-Systems in der POSIX-Shell ausgegeben.

Informationen über das Binder-Lader-System des BS2000/OSD können Sie dem Handbuch „BLSSERV“ [11] entnehmen.

Zum Binden in der POSIX-Shell stehen die Kommandos *cc*, *c89*, *cobol* und *CC* zur Verfügung. Sie sind im Handbuch „POSIX-Kommandos des C/C++-Compilers“ [4] ausführlich beschrieben.

4.2 C/C++-Compiler

Die BS2000-Compiler C/C++ können sowohl aus der BS2000-Umgebung (mit SDF) als auch aus der POSIX-Umgebung (POSIX-Shell) aufgerufen und mit Optionen gesteuert werden.

Compilersteuerung über die SDF-Schnittstelle

Sämtliche Compiler-Ein-/Ausgaben sind sowohl im BS2000-Dateisystem (DMS/PLAM) als auch im POSIX-Dateisystem möglich:

- Eingabe von Quellprogrammen
- Eingabe von Include-Dateien
- Ausgabe von LLMs
- Ausgabe von wiederübersetzbaren Quellprogrammen
- Ausgabe von Übersetzungslisten
- Ausgabe von Meldungslisten
- Ausgabe von CIF-Informationen

Beliebige Mischfälle, d.h. die Ein- und Ausgabe sowohl von BS2000- als auch von POSIX-Dateien in einem Übersetzungslauf, sind möglich.

Die SDF-Schnittstelle des Compilers ist im Handbuch „C/C++ V3.x (BS2000/OSD), C/C++-Compiler“ beschrieben (x = jeweilige Version) beschrieben:

Compilersteuerung über die POSIX-Shell-Schnittstelle

Für die Steuerung der C/C++-Compiler aus der POSIX-Umgebung stehen folgende POSIX-Kommandos zur Verfügung:

cc, c89	Übersetzen von C-Sourcen
CC	Übersetzen von C++-Sourcen
cclistgen	Aufruf des globalen Listengenerators in C/C++

Diese Kommandos sind im Handbuch „POSIX-Kommandos des C/C++-Compilers“ [4] ausführlich beschrieben.

Sämtliche Compiler-Ein-/Ausgaben erfolgen ausschließlich im POSIX-Dateisystem.

In die Aufrufkommandos *cc*, *c89* und *CC* ist auch eine Binde-Phase integriert, in der die übersetzten Objekte zu einer ausführbaren Einheit gebunden werden können.

Mit den Optionen und Operanden der oben genannten Aufrufkommandos sind weitgehend die Leistungen und Funktionen abgedeckt, die mit der Compiler-Steuerung über die SDF-Schnittstelle zur Verfügung stehen. Die Syntax der POSIX-Kommandos ist an der Definition im XPG4-Standard bzw. an den in UNIX-Systemen üblichen Shell-Kommandos orientiert.

Der Compiler unterstützt arithmetische Datentypen der Länge 64 (long long). Das ist eine Voraussetzung, um Zugriffe auf große Dateien in POSIX zu programmieren.

Hinweise zu CRTE

Da CRTE (Common RunTime Environment) die Laufzeitumgebung für C- und C++-Programme bereitstellt, ist es Voraussetzung für die Verwendung des C/C++-Compilers.

Die benötigte CRTE-Version hängt von der Version des eingesetzten Compilers ab. Nähere Informationen finden Sie in der Freigabemitteilung des C/C++-Compilers.

Zeichensätze für Ein-/Ausgabe-Dateien

Die Quellprogramme und Include-Dateien können im EBCDIC- oder ASCII-Code vorliegen. Somit ist es auch möglich, Quellprogramme aus Dateisystemen zu übersetzen, die auf einem fernen UNIX-System liegen. Alle Dateien eines Dateisystems (lokales POSIX-Dateisystem oder eingehängtes fernes Dateisystem) müssen im selben Zeichensatz vorliegen. D.h. im POSIX-Dateisystem müssen alle Dateien im EBCDIC-Code und im fernen UNIX-Dateisystemen müssen alle Dateien im ASCII-Code vorliegen.

Die Umgebungsvariable *IO_CONVERSION* muss mit dem Wert „YES“ belegt sein (siehe dazu auch Seite 37).

Der Ausgabe-Zeichensatz der Textdateien (Listen etc.) richtet sich nach dem Zeichensatz des Zieldateisystems.

Es wird generell EBCDIC-Ablaufcode erzeugt.

4.3 COBOL85 / COBOL2000 Compiler

Die BS2000-Compiler COBOL85 ab V2.3 und COBOL2000 können sowohl aus der BS2000-Umgebung (mit SDF) als auch aus der POSIX-Umgebung (Shell) aufgerufen und mit Optionen gesteuert werden.

Die Benutzung der Compiler ist in folgenden Benutzerhandbüchern beschrieben:

– „COBOL85 (BS2000/OSD)“ [13]
– „COBOL2000 (BS2000/OSD)“ [12]

Compilersteuerung über die SDF-Schnittstelle

Sämtliche Compiler-Ein-/Ausgaben sind sowohl im BS2000-Dateisystem (DMS/PLAM) als auch im POSIX-Dateisystem möglich:

– Eingabe von Quellprogrammen
– Eingabe von Copy Elementen
– Ausgabe von LLMs
– Ausgabe von Übersetzungslisten
– Ausgabe von Meldungslisten

Mischfälle, d.h. die Ein- und Ausgabe sowohl von BS2000- als auch von POSIX-Dateien in einem Übersetzungslauf, sind möglich.

Compilersteuerung über die POSIX-Shell-Schnittstelle

Für die Steuerung der COBOL-Compiler aus der POSIX-Umgebung steht das POSIX-Kommando *cobol* zur Verfügung.

Sämtliche Compiler-Ein-/Ausgaben erfolgen ausschließlich im POSIX-Dateisystem.

In das Kommando *cobol* ist auch eine Binde-Phase integriert, in der die übersetzten Objekte zu einer ausführbaren Einheit gebunden werden können.

Mit den Optionen und Operanden der oben genannten Aufrufkommandos sind weitgehend die Leistungen und Funktionen abgedeckt, die mit der Compiler-Steuerung über die SDF-Schnittstelle zur Verfügung stehen. Die Syntax des POSIX-Kommandos ist an den in UNIX-Systemen üblichen Shell-Kommandos orientiert.

Hinweise zu CRTE

Mit CRTE (Common RunTime Environment) wird die Laufzeitumgebung für COBOL-Programme bereitgestellt. CRTE ist auch Voraussetzung für die Verwendung des COBOL-Compilers.

Nähere Informationen finden Sie in der Freigabemitteilung des COBOL-Compilers.

Zeichensätze für Ein-/Ausgabe-Dateien

Die Quellprogramme und Include-Dateien können im EBCDIC- oder ASCII-Code vorliegen. Somit ist es auch möglich, Quellprogramme aus Dateisystemen zu übersetzen, die auf einem fernen UNIX-System liegen. Alle Dateien eines Dateisystems (lokales POSIX-Dateisystem oder eingehängtes fernes Dateisystem) müssen im selben Zeichensatz vorliegen. D.h. im POSIX-Dateisystem müssen alle Dateien im EBCDIC-Code und im fernen UNIX-Dateisystemen müssen alle Dateien im ASCII-Code vorliegen.

Die Umgebungsvariable *IO_CONVERSION* muss mit dem Wert „YES“ belegt sein (siehe dazu auch Seite 37).

Der Ausgabe-Zeichensatz der Textdateien (Listen etc.) richtet sich nach dem Zeichensatz des Zieldateisystems.
Es wird generell EBCDIC-Ablaufcode erzeugt.

Ablauf von COBOL Anwendungen

Der Zugriff auf Dateien im POSIX-Dateisystem mittels COBOL-IO erfordert die Übersetzung des COBOL-Programms mit der Option COMOPT ENABLE-UFS=YES, die vom POSIX-Kommando *cobol* implizit gesetzt wird. So übersetzte Programme können dann sowohl BS2000- als auch POSIX-Dateien verarbeiten.

Es gelten ansonsten die im COBOL-Benutzerhandbuch beschriebenen Besonderheiten für COBOL-Programme beim Ablauf unter POSIX-Shell, d. h. es werden folgende Sprachmittel nicht unterstützt:
- CALL-Bezeichner,
- ENTRY,
- READ PREVIOUS und
- Dateiverarbeitung mit CODESET STANDARD.

Weitere Unterschiede in der Funktionalität ergeben sich aus Systemunterschieden bei der Dateiverarbeitung, speziell Bandverarbeitung mit Kennsätzen, Simultanverarbeitung von Dateien, Fixpunktausgabe und bei der Zuweisung von Dateien. Im Ausnahmefall wird anstelle des BS2000 DMS-Codes der PROSOS SIS-Code über die Filestatus-Datenfelder zur Verfügung gestellt.

Zur Steuerung der Anwendung von außen stehen Jobvariable sowie Auftrags- und Benutzerschalter nicht zur Verfügung. Als Erweiterung gibt es Sprachmittel zum Zugriff auf die Kommandozeile.

4.4 BS2000/OSD Environment For Java (JENV)

Mit dem BS2000/OSD Environment For Java (JENV) können Java-Programme, die auf beliebigen Plattformen erstellt wurden, auf BS2000-Systemen zum Ablauf gebracht werden. Dies entspricht dem Java-Konzept „write once, run everywhere“. Ebenso sind Java-Anwendungen, die für BS2000/OSD entwickelt wurden, auch auf anderen Plattformen ablauffähig.

Der Funktionsumfang entspricht dem jeweils zugrunde liegenden Java2 Standard Edition SDK von JavaSoft. Dies wird durch die Validierung bei Sun sichergestellt und dadurch das Recht erworben, das „Java Compatible Logo“ für BS2000/OSD zu verwenden.

JENV wird normalerweise innerhalb der POSIX-Umgebung (POSIX-Shell) verwendet. Es lässt sich aber auch aus der BS2000-Umgebung über Prozeduren steuern.

Bei der POSIX-Installation von JENV ist zu beachten, dass relativ viel Speicherplatz im POSIX-Filesystem benötigt wird. Dies ist nach Möglichkeit schon beim Design des POSIX-Filesystems zu berücksichtigen. Angaben über benötigte Größen befinden sich in der Freigabemitteilung.

JENV ist Lieferbestandteil von BS2000/OSD-BC und ist auf allen Hardware-Plattformen ablauffähig. Für die Nicht-/390-Plattformen gibt es spezielle hochperformante Varianten, die dann nur auf der jeweiligen Hardware ablauffähig sind.

Weitere Informationen sind dem jeweiligen Handbuch und der Freigabemitteilung zu entnehmen.

4.5 EDT

Mit dem EDT können Sie POSIX-Dateien erzeugen und bearbeiten. Dazu muss das Subsystem POSIX gestartet sein.

Der EDT im Unicode-Modus kann im BS2000 mit /START-EDTU oder in der POSIX-Shell mit dem Kommando *edtu* aufgerufen werden.

Der EDT im Kompatibilitätsmodus kann im BS2000 mit /START-EDT oder in der POSIX-Shell mir dem Kommando *edt* aufgerufen werden. Das Shell-Kommando *edt* verwendet die Version 16.6 des EDT.

Für weitere Informationen siehe Handbücher „POSIX (BS2000/OSD) - Kommandos“ [1], „EDT (BS2000/OSD) - Unicode-Modus Anweisungen“ [15] und „EDT (BS2000/OSD) - Anweisungen“ [14].

4.6 File-Transfer openFT für BS2000

Das File Transfer-Produkt openFT für BS2000 unterstützt die Übertragung von POSIX-Dateien. Optionale Komponenten sind openFT-FTAM zur Realisierung der FTAM-Funktionalität, openFT-AC für den mit der FTAC-Funktionalität gebotenen Zugangs- und Zugriffsschutz und ggf. openFT-FTP zur Unterstützung der FTP-Funktionalität.

Das Subsystem POSIX muss gestartet sein, damit die POSIX-Funktionen von openFT genutzt werden können. Außerdem muss POSPRRTS gestartet sein.

Angabe einer POSIX-Datei

Eine POSIX-Datei muss in den openFT-Kommandos durch eine besondere Syntax angegeben werden: Dateinamen, die mit / oder ./ beginnen, werden als voll- bzw. teilqualifizierte POSIX-Dateinamen interpretiert. Dateien, die nicht mit einem dieser Zeichen beginnen, gelten als BS2000-Dateien.

Nähere Informationen können Sie dem Handbuch „openFT für UNIX“ [19] entnehmen.

4.7 HSMS

HSMS ermöglicht auch das Sichern, Archivieren und Restaurieren von Dateien, die auf fernen Rechnern im Netz liegen. Das zu bearbeitende Dateisystem kann entweder das lokale BS2000-UFS sein oder ein fernes UNIX-Dateisystem, das der BS2000-Systemverwalter am Dateiverzeichnis „HSMS“ des lokalen BS2000-UFS einhängen muss. Das Dateiverzeichnis „HSMS“ befindet sich direkt unter dem Root-Verzeichnis „/“, das in jedem UNIX-Dateisystem vorhanden ist und als zentraler Einstieg dient.

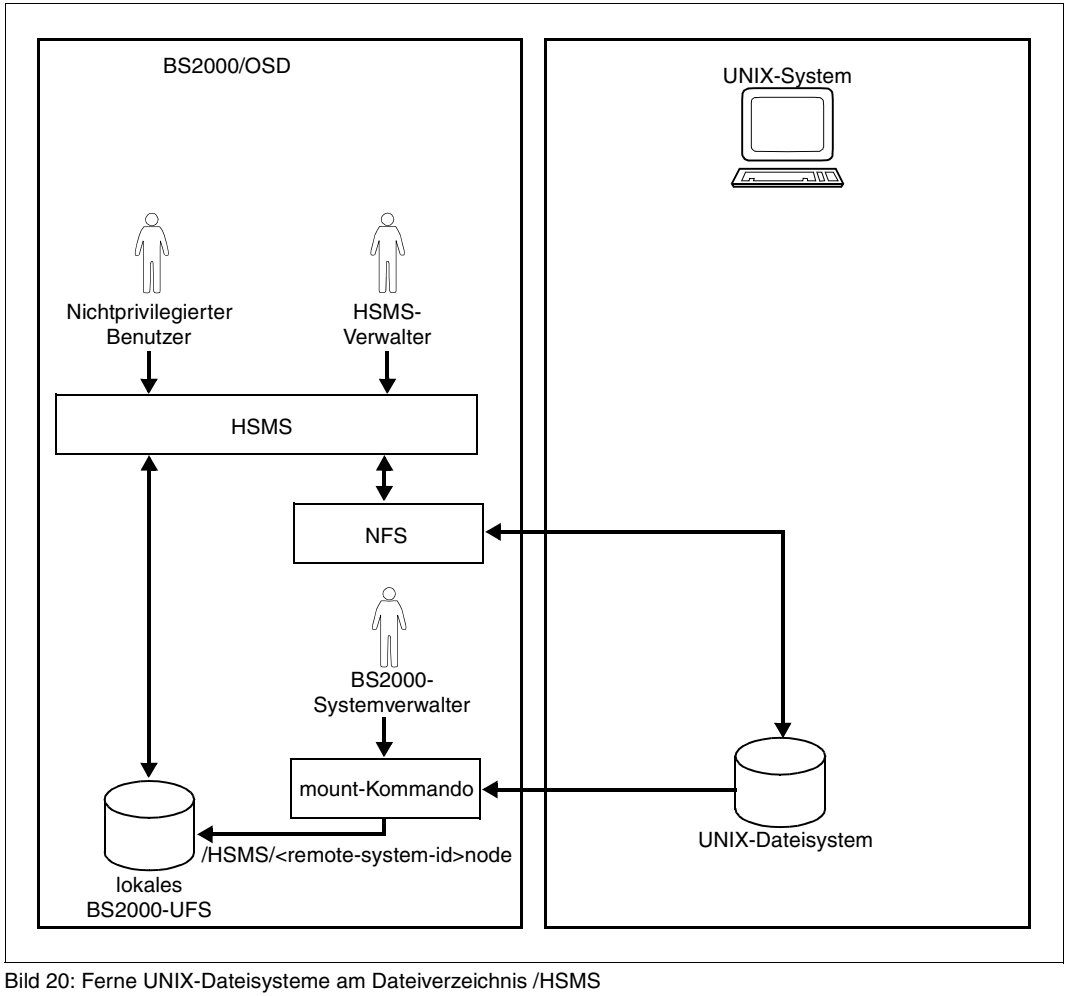

Bild 20: Ferne UNIX-Dateisysteme am Dateiverzeichnis /HSMS

Die folgenden HSMS-Anweisungen stehen dem HSMS-Verwalter und - in eingeschränktem Umfang - auch dem nichtprivilegierten Benutzer zur Verfügung, um Dateien eines UNIX-Dateisystems bearbeiten zu können:

- ARCHIVE-NODE-FILES
  Knotendateien archivieren
- BACKUP-NODE-FILES
  Knotendateien sichern
- COPY-NODE-SAVE-FILE
  Knoten-Sicherungsdatei kopieren
- MODIFY-NODE-PARAMETERS
  Parameter eines Knoten-S0 ändern
- RESTORE-NODE-FILES
  Knotendateien restaurieren
- SELECT-NODE-FILES
  Knotendateien auswählen
- SHOW-NODE-PARAMETERS
  Parameter von Knoten-S0 ausgeben

Nähere Informationen können Sie dem Handbuch „HSMS (BS2000/OSD)“ [21] entnehmen.

4.8 NFS

Um mit den Dateisystemen eines fernen Rechners arbeiten zu können, muss das Softwareprodukt NFS (Network File System) auf dem fernen und lokalen Rechner installiert sein. Am fernen Rechner (NFS-Server) muss das einzuhängende Dateisystem mit dem NFS-Kommando *share* bereitgestellt und am lokalen Rechner (NFS-Client) mit dem NFS-Kommando *mount* eingehängt werden. Danach kann auf das ferne Dateisystem vom lokalen Rechner aus zugegriffen werden. Umgekehrt kann natürlich auch der lokale Rechner der NFS-Server sein und der entfernte Rechner der NFS-Client.

Nähere Informationen können Sie dem Handbuch „NFS (BS2000/OSD)“ [8] entnehmen.

4.9 SECOS

POSIX verwendet für die Verwaltung und Zugangskontrolle der POSIX-Benutzer den Baustein SRPM des Softwareprodukts SECOS.

Wenn SECOS in Ihrem System nicht installiert ist, so ist doch der für POSIX relevante Teil von SRPM im Grundausbau des BS2000 enthalten.

Nähere Informationen zur BS2000-Verwaltung der POSIX-Benutzer finden Sie in diesem Handbuch im Kapitel „POSIX-Benutzer verwalten“ auf Seite 181.
Die Zugangskontrolle für Anwender, die sich von einem UNIX-System mit dem Kommando *rlogin* an einen BS2000-Rechner anschließen wollen, ist im Abschnitt „Zugang von einem Zeichenterminal“ auf Seite 64 beschrieben.

Wenn SECOS im Einsatz ist, stehen Ihnen für POSIX die folgenden zusätzlichen Möglichkeiten zur Verfügung:

- Verwendung des Privilegs POSIX-ADMINISTRATION für ausgewählte Benutzerkennungen (SRPM).
- Protokollierung und Auswertung sicherheitsrelevanter Ereignisse, die POSIX betreffen, mit SAT.
  Neben den allgemeinen Möglichkeiten der Überwachung von Benutzerkennungen, DVS-Dateiobjekten und Ereignissen sind folgende Ereignisse speziell für POSIX definiert:
  - Ereignis JFK: POSIX-Task erzeugen
  - Ereignis UPA: Kommando /MODIFY-POSIX-USER-ATTRIBUTES
  - Ereignis UPD: Kommando /MODIFY-POSIX-USER-DEFAULTS

  Die sicherheitsrelevanten Ereignisse der Privilegienverwaltung - z. B. das Privileg POSIX-ADMINISTRATION vergeben - werden stets mit SAT protokolliert.
- Protokollierung von ca. 50 sicherheitsrelevanten POSIX-Events, gruppiert nach:
  - Dateizugriffen (POSIX-FILE-and-Directory)
  - Prozessattributen (POSIX-Process)
  - fork (POSIX-CHILD-Process)
  - semaphore, shared memory (POSIX-SYSTEM-Resources)
- Eigene Zugangsklassen für rechnerübergreifende POSIX-Dienste (*rlogin*, *rcp*, ...).

Nähere Informationen können Sie den SECOS-Handbüchern „Security Control System - Zugangs- und Zugriffskontrolle“ [9] und „Security Control System - Beweissicherung“ [10] entnehmen.

4.10 SOCKETS/XTI (POSIX-SOCKETS)

Mit der Liefereinheit POSIX stehen auch die SOCKETS/XTI-Schnittstellen zur Verfügung. Es handelt sich dabei um Schnittstellen zur Programmierung von Netzfunktionen, mit denen der Zugang zum Internet über TCP/IP und UDP/IP ermöglicht wird. Diese Schnittstellen gewährleisten damit den Zugang zur offenen Netzwerk-Welt.

Die SOCKETS/XTI-Schnittstellen werden mit POSIX-SOCKETS ausgeliefert und sind in einer eigenen Bibliothek definiert. Wenn diese Bibliothek in eine POSIX-Anwendung eingebunden ist, stellen die SOCKETS/XTI-Schnittstellen über das Subsystem POSIX und das Transportsystem BCAM die Verbindung zum Netzwerk her.

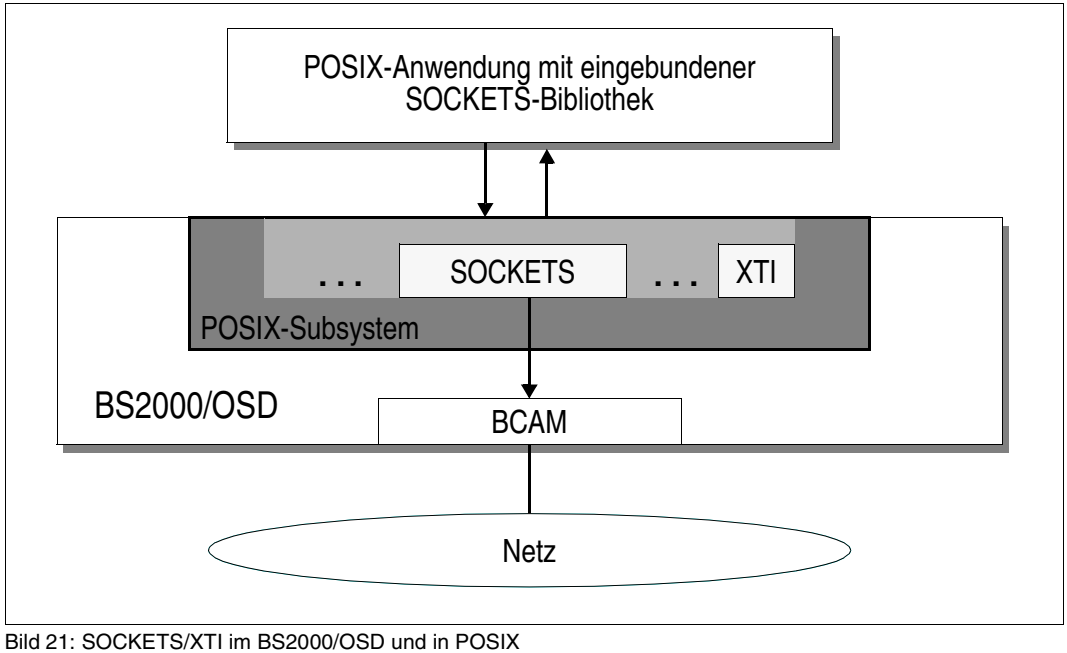

Bild 21: SOCKETS/XTI im BS2000/OSD und in POSIX

Nähere Informationen können Sie dem Handbuch „SOCKETS/XTI für POSIX“ [3] entnehmen.

4.11 SPOOL

Mit SPOOL können Sie POSIX-Dateien mit BS2000- oder POSIX-Kommandos ausdrucken.

Beispiel

Drucken der POSIX-Datei */home/psxroot/usr1* aus dem BS2000:

```
/PRINT-DOCUMENT FROM-FILE='/home/psxroot/usr1',...
```

Drucken derselben POSIX-Datei aus der POSIX-Shell:

```
/START-POSIX-SHELL
.
.
.
/home/psxroot> lp usr1
/home/psxroot> exit
```

POSIX-Dateien können Sie mit dem POSIX-Kommando *lp* auf dem angeschlossenen BS2000-Drucker ausdrucken. Das Kommando *lp* verwendet den BS2000-SPOOL zum Ausdrucken. Es wird keine ID für einen Druckauftrag vergeben.
Die Druckaufträge können nur über den BS2000-SPOOL verwaltet werden.

Nähere Informationen können Sie den Handbüchern zu „SPOOL“ ([32] und [33]) entnehmen.

4.12 TLI (POSIX-NSL)

Neben den SOCKETS-Schnittstellen (siehe Seite 95) stehen auch die TLI-Netzwerk-Schnittstellen zur Verfügung. Sie ermöglichen ebenfalls den Zugriff auf das Internet auf Basis von TCP/IP und UDP/IP. Die TLI-Schnittstellen sind in der Komponente POSIX-NSL enthalten; POSIX-NSL wird mit POSIX ausgeliefert.

Die TLI-Schnittstellen bestehen - ebenso wie die SOCKETS-Schnittstellen - aus einer Reihe von Bibliotheksfunktionen, die über das Transportsystem BCAM die Verbindung zum Netzwerk herstellen.

4.13 AID

Mit der Dialog-Testhilfe AID können Sie neben reinen BS2000-Programmen auch reine POSIX-Programme und gemischte Programme testen. Gemischte Programme benutzen sowohl BS2000-Programmschnittstellen als auch POSIX-Programmschnittstellen. Das Testen wird ermöglicht durch Erweiterungen in den AID-Kommandos %AID und %STOP sowie durch das POSIX-Kommando *debug*.

Das AID-Kommando %AID wurde um zwei neue Operanden FORK={OFF | NEXT | ALL} und EXEC={OFF | ON} ergänzt. Diese bewirken, dass unmittelbar nach einem *fork()*- bzw. *exec()*-Aufruf das Programm unterbrochen wird und in den Testmodus wechselt, so dass Sie wie gewohnt AID-Kommandos zum Testen Ihres Programms eingeben können.

Das AID-Kommando %STOP wurde ebenfalls um zwei neue Operanden T=tsn (Task Sequence Number) und PID=pid (Process Identification) erweitert, über die Sie eine durch *fork()* entstandene Task unterbrechen können.

AID meldet sich mit der Prozessnummer (pid) der unterbrochenen Task, und Sie können den weiteren Verlauf dieser Task über AID-Kommandos kontrollieren.

Das POSIX-Kommando *debug* ermöglicht es Ihnen, in der POSIX-Shell ein Programm mit LSD zu laden oder einen bereits laufenden Prozess zu unterbrechen und in den Testmodus zu versetzen:

`debug progname`
: Das Programm wird mit LSD in einer Fork-Task geladen und in den Testmodus versetzt; Sie können AID-Kommandos eingeben. Das Kommando 'debug progname' in der POSIX-Shell entspricht somit dem BS2000-Kommando
`LOAD-PROGRAM progname, ... TEST-OPTIONS=*YES`
in der BS2000-Umgebung.

`debug -p pid`
: Der Prozess mit der angegebenen pid wird von AID übernommen und unterbrochen. 'debug -p pid' in der POSIX-Shell entspricht dem oben erwähnten AID-Kommando %STOP PID=pid, das Sie im BS2000-Systemmodus oder im Testmodus einer Task eingeben können.

Dumps von gemischten oder POSIX-Programmen werden wie bisher im BS2000 abgelegt und können dort bearbeitet werden. Falls AID zum Dump eines POSIX-Programms die LSD über das AID-Kommando %SYMLIB nachladen soll, müssen Sie beachten, dass %SYMLIB nicht auf POSIX-Dateien zugreifen kann. Die entsprechende Datei muss zunächst mit dem POSIX-Kommando *bs2cp* als L-Element in eine PLAM-Bibliothek im BS2000 kopiert werden und kann dann mit %SYMLIB zugewiesen werden.

Wie POSIX-Programme und gemischte Programme mit AID getestet werden, ist ausführlich in den Handbüchern zu AID [35] bzw. [36] beschrieben.

4.14 SORT

POSIX-Dateien können in der SORT-Steueranweisung ASSIGN-FILES oder im Kommando SORT-FILE als Eingabedatei (Operand INPUT-FILES) oder als Ausgabedatei (Operand OUTPUT-FILES) zugewiesen werden.

Zur Unterscheidung von BS2000-Dateinamen müssen POSIX-Dateinamen in den Operanden INPUT-FILES und OUTPUT-FILE in Hochkommata angegeben werden.

Arbeits- und Hilfsdateien dürfen keine POSIX-Dateien sein.

Die Daten in den POSIX-Dateien liegen im Text-Format vor, das von SORT nicht unmittelbar verarbeitet werden kann. Sie werden daher vor der Bearbeitung durch die Sortierroutinen von SORT in Sätze variabler Länge umgewandelt, denen jeweils ein Satzlängenfeld vorangestellt wird.

Nach dem Sortiervorgang wandelt SORT die sortierte Ausgabedatei wieder in das Text-Format um, wenn sie im POSIX-Dateisystem gespeichert werden sollen.

Die interne Verwendung von Sätzen variabler Satzlänge bewirkt, dass sich die Position der Benutzerdaten im Satz um das Satzlängenfeld verschiebt. Für den Anwender von POSIX-Dateien hat das im Normalfall jedoch keine Auswirkungen. SORT berechnet die Feldpositionen bei Sätzen aus POSIX-Dateien standardmäßig relativ zum Anfang der Benutzerdaten.

Will der Anwender dennoch auf das interne Satzlängenfeld zugreifen, z.B. um die Sätze nach ihrer Länge zu sortieren, steht der Operand IGNORE-LENGTH-FIELD in der Anweisung SET-SORT-OPTIONS und im Kommando SORT-FILE zur Verfügung.

Die Angabe IGNORE-LENGTH-FIELD=*NO bewirkt, dass sowohl bei variabel langen Sätzen in BS2000-Dateien, als auch bei Sätzen in POSIX-Dateien die Positionen innerhalb des Satzes ab Satzanfang berechnet werden. Damit beginnen die Benutzerdaten an Position 5 im Satz.

Die Verschlüsselung des Satzende-Kennzeichens wird durch den Operanden CODE in der Anweisung ASSIGN-FILES und im Kommando SORT-FILE bestimmt. Bei CODE=EBCDIC wird das Satzende-Kennzeichen als X'0A' verschlüsselt, bei CODE=ASCII als X'15'.

Bei der Verwendung von POSIX-Dateien als Ausgabedatei ist darauf zu achten, dass die Ausgabesätze keine Zeichen enthalten, die als Satzende-Kennzeichen interpretiert werden. Das bedeutet im einzelnen:

– In der Anweisung SORT-RECORDS oder im Kommando SORT-FILE dürfen keine Konstantenfelder angegeben werden, die Satzende-Kennzeichen enthalten.

– Die Sätze einer BS2000-Eingabedatei dürfen keine Satzende-Kennzeichen enthalten, wenn die Ausgabedatei eine POSIX-Datei sein soll.

– Die Sortierart „Adresslistensortieren“ darf nicht verwendet werden, da nicht sichergestellt werden kann, dass die Adressfelder keine als Satzende-Kennzeichen interpretierbaren Zeichen enthalten.

Das Kommando *sort*, das in einer POSIX-Shell aufgerufen werden kann, ist nicht identisch mit einem Aufruf des Produkts SORT.

Nähere Informationen können Sie dem Handbuch „SORT (BS2000/OSD)“ [34] entnehmen.

4.15 interNet Services

Das Produkt interNet Services stellt folgende TCP/IP-Services bereit:

– File Transfer Protocol (FTP)
– Terminal Service (TELNET)
– Domain Name Service (DNS) Resolver (DNSD)
– Domain Name Service (DNS) Server (NAMED)
– Network Time Protocol (NTP)
– Secure Shell Server (SSHD) und Secure Shell Clients
– Mail-Server (Postfix) und IMAP-/POP3-Server

Mit interNet Services ab Version 2.5A stehen in POSIX ein ftp-Client und ein telnet-Client zur Verfügung. Diese können Sie mit dem POSIX-Kommando *ftp* bzw. *telnet* aufrufen.

Mit interNet Services ab Version V3.0B stehen in POSIX über OpenSSH eine Reihe von Tools zur Verfügung, welche als Ersatz für die unsicheren r-Utilities (*rlogin*, *rsh*, *rcp*) dienen.

Mit interNet Services ab Version V3.1A steht in POSIX ein Mail-Server auf der Portierungsbasis Postfix zur Verfügung. Ferner steht ein IMAP-/POP3-Server zur Verfügung, der einem Mail-Client den Zugriff auf die Mailboxen gestattet.

Im Folgenden wird näher auf die Komponente FTP eingegangen.

Mit FTP kann von fernen Systemen (UNIX-Systeme, Windows, BS2000/OSD) auf POSIX-Dateiverzeichnisse zugegriffen werden. Voraussetzung dafür ist, dass auf dem BS2000/OSD-System der FTP-Server-Task mit dem Kommando START-FTP-DEMON gestartet wurde.

Mit der Pfadangabe %POSIX bei *cd* oder *lcd* in einer FTP-Sitzung kann in das POSIX-UFS gewechselt werden.

Beispiel

Das folgende Beispiel zeigt einen Ausschnitt einer FTP-Sitzung:

```
FTP> open BS2SERVER ──────────────────────────────────────────────── (1)
Connected to BS2SERVER, port 21.
220 BS2SERVER FTP server (Version ... ) ready.
Name (BS2SERVER:USR): user1.
331 Password required for user1.
Password (BS2SERVER:red):                                          (2)
332 Account required.
Account: m0815xyz
230 User USER1 logged in.
Ftp> cd %POSIX ──────────────────────────────────────────────────── (3)
250 "/home/user1" is current directory now
Ftp> ...
Ftp> bye ────────────────────────────────────────────────────────── (4)
221 Goodbye.
```

(1) Geben Sie beim Kommando *open* den BS2000/OSD-Server an, auf dem das POSIX-Dateisystem installiert ist.

(2) Geben Sie Benutzerkennung, Passwort und Account Ihrer BS2000-Kennung auf diesem Server ein.

(3) Die Angabe %POSIX bei *cd* bewirkt, dass Sie vom BS2000-Dateisystem in das POSIX-Dateisystem wechseln. Danach befinden Sie sich in dem HOME-Verzeichnis der POSIX-Kennung, die Ihrer BS2000-Kennung zugeordnet ist.

(4) Mit dem FTP-Befehl *bye* verlassen Sie das POSIX-Dateisystem, melden sich bei BS2000 ab und schließen die FTP-Sitzung.

Nähere Informationen können Sie den Handbüchern zu „interNet Services“ [40] und [41] entnehmen.

4.16 APACHE Webserver auf BS2000/OSD

Mit dem Webserver-Produkt APACHE für BS2000/OSD steht der weltweit meistgenutzte Webserver APACHE auch für BS2000/OSD bereit.

Das kostenfreie Produkt enthält die volle Funktionalität des Original-APACHE-Webservers, einschließlich der Unterstützung des https-Protokolls, und wurde erweitert um zahlreiche integrierte Zusatzkomponenten zur Web-Programmierung. Dazu gehören die Skriptinterpreter PHP und Perl, der Anschluss für Java-Servlets und Java Server Pages via TOMCAT sowie das Dokumentenverwaltungssystem WebDAV. Darüber hinaus werden Perl- und PHP-Interpreter als Standalone-Programme angeboten.

APACHE (BS2000/OSD) läuft im POSIX-Subsystem ab. Zugriff auf BS2000-SAM- und -ISAM-Dateien sowie auf SESAM/SQL- und ORACLE-Datenbanken sowie die Ausführung von BS2000-Kommandos ist über PHP möglich.

4.17 SNMP-Basic-Agent und SNMP-Standard-Collection.

SNMP-Basic-Agent BS2000 (SBA-BS2) und SNMP-Standard-Collection BS2000 (SSC-BS2) bieten die Basis-Funktionalität für BS2000/OSD-Systeme, um in SNMP-basierte Managementumgebungen eingebunden werden zu können. SBA-BS2 und SSC-BS2 erlauben Netz-System- und Anwendungsmanagement über SNMP von einer zentralen Management-Station aus.

Das Produkt SNMP-Standard-Collection BS2000 (SSC-BS2) erweitert die Möglichkeiten des SNMP Basic-Agent BS2000 um:

- die Überwachung zentraler Systemressourcen wie die CPU-Auslastung (Performance Basisüberwachung) sowie Speicher, Geräte, Filesysteme, Pubsets und Platten gemäß Host Resources MIB.
- das Management wesentlicher Komponenten und Produkte des BS2000 wie AVAS, HIPLEX-AF, OMNIS, openFT, der Print-Service (Spool und RSO), SESAM/SQL, das Storage Management und HSMS.

Weitere Informationen zum SNMP Management finden Sie im Handbuch „SNMP Management“ [42].

5 POSIX installieren

Dieses Kapitel wendet sich an die Systemverwalter von BS2000 und POSIX, die POSIX und weitere POSIX-Programmpakete installieren wollen. Es informiert Sie über

- den Lieferumfang von POSIX
- das Konzept der POSIX-Installation
- die Installationsschritte für POSIX bei Erstinstallation und bei Upgrade-Installation
- das Installationsprogramm von POSIX (Dialog und Batch)
- die Protokolldateien von POSIX
- die Informationsdatei von POSIX (SYSSSI)

5.1 Lieferumfang

Die grundlegenden POSIX-Funktionen und -Kommandos sind Bestandteil des BS2000-Grundausbaus BS2000/OSD-BC und werden in einer eigenen technischen Liefereinheit „POSIX“ ausgeliefert. Diese Liefereinheit besteht aus folgenden Produktkomponenten:

- POSIX-BC (Subsystem POSIX und Basis-Shell)
- POSIX-SH (erweiterte Shell)
- POSPRRTS (Laufzeitsystem für den privilegierten Teil von POSIX)
- POSIX-SOCKETS (SOCKETS/XTI-Netzwerkschnittstellen)
- POSIX-NSL (TLI-, RPC- und XDR-Funktionen)
- POSIX-ADDON-LIB (Schnittstellen, die nicht zum XPG4-Standard gehören)

Mit POSIX steht der vollständige Kommandoumfang gemäß dem XPG4-Standard zur Verfügung.

Die Programmierschnittstellen für POSIX werden als Bibliotheksfunktionen für die Programmiersprache C/C++ im Rahmen des Softwareprodukts CRTE freigegeben und installiert.

Die Dateinamen der einzelnen Lieferbestandteile entnehmen Sie bitte der Freigabemitteilung (SYSFGM) zu POSIX-BC.

5.2 Konzept der POSIX-Installation

Die Installation von POSIX und POSIX-Produkten verläuft in zwei Schritten:

1. Installation im BS2000 über das IMON/SOLIS-Verfahren.

   Die SOLIS-Lieferung wird im BS2000 über IMON installiert. Dabei werden u.a. der Subsystemkatalog und das Software Configuration Inventory (SCI) aktualisiert. Auf die Installation im BS2000 wird im Rahmen dieses Handbuchs nicht näher eingegangen.

   Nach der Installation im BS2000 sind die Programme noch nicht ablauffähig. Sie müssen in einem zweiten Schritt im POSIX-Dateisystem installiert werden.

2. Installation im POSIX-Dateisystem.

   Für die Installation im POSIX steht ein eigenes POSIX-Installationsprogramm zur Verfügung, das mit dem Kommando /START-POSIX-INSTALLATION gestartet wird. Das POSIX-Installationsprogramm kann im Dialog oder im Batch (d.h. gesteuert über eine Parameterdatei) aufgerufen werden kann.

   Bei der Installation im POSIX kann das SCI wahlweise ausgewertet werden (siehe Abschnitt „Das Installationsprogramm im Zusammenspiel mit IMON“ auf Seite 108).

Dieses Unterkapitel geht näher auf folgende Themen der POSIX-Installation ein:

- die wichtigsten Eigenschaften des POSIX-Installationsprogramms
- das Format der Programmpakete
- die Installation mit IMON-Unterstützung
- die multimodale Installation, d.h simultane Installation für mehrere Plattformen (/390, SPARC, X86)
- die Installation ohne IMON-Unterstützung
- die Installation von privaten Programmpaketen

5.2.1 Eigenschaften des POSIX-Installationsprogramms

Mit dem POSIX-Installationsprogramm von POSIX können Sie eine Erstinstallation oder eine Upgrade-Installation von POSIX durchführen, POSIX-Dateisysteme verwalten sowie Software-Programmpakete hinzufügen oder löschen.

Das POSIX-Installationsprogramm wird mit /START-POSIX-INSTALLATION gestartet und kann sowohl im Dialog als auch im Batch aufgerufen werden:

– Im Dialogbetrieb geben Sie die notwendigen Daten in Masken ein. Fehlerhafte Eingaben und Inkonsistenzen werden im Dialog gemeldet, sodass Sie Ihre Eingaben umgehend korrigieren können. Näheres finden Sie im Abschnitt „POSIX-Installationsprogramm im Dialog“ auf Seite 126.

– Im Batchbetrieb schreiben Sie die zur Installation notwendigen Daten in Parameterdateien, deren Layout genau festgelegt ist. Fehlerhafte Parameterdateien führen zum Abbruch des POSIX-Installationsprogramms. Näheres siehe Abschnitt „Automatisierter Ablauf des POSIX-Installationsprogramms“ auf Seite 136.

Die Aktionen des Installationsprogramms werden protokolliert, siehe Abschnitt „Protokollierung der Installation“ auf Seite 144.

Welche Arbeitsschritte bei Erst- oder Upgrade-Installations im Einzelnen notwendig sind, finden Sie in Abschnitt „Erstmalige Installation von POSIX“ auf Seite 115 bzw. Abschnitt „Upgrade-Installation von POSIX“ auf Seite 124.

5.2.2 Format der Programmpakete

Programmpakete zur Installation unter POSIX werden für die von Fujitsu gelieferte Software als BS2000-PLAM-Bibliotheken mit dem Standardnamen <Präfix>LIB.<product>.<version>. <package> geliefert und unter einer beliebig wählbaren Ablagekennung eingespielt.

Programmpakete können wahlweise aus dem Software Configuration Inventory, d.h. nach dem offiziellen Lieferverfahren mit IMON-Support, siehe Abschnitt „Das Installationsprogramm im Zusammenspiel mit IMON“ auf Seite 108, oder von einer beliebigen Ablagekennung installiert werden, siehe Abschnitt „Produktinstallation ohne IMON-Unterstützung“ auf Seite 110. Das gilt auch für private Programmpakete, siehe Abschnitt „Private Programmpakte zur Installation vorbereiten“ auf Seite 110, z.B. private POSIX-Anwendungen, da auch diese im Software Configuration Inventory registriert werden können. Mittels Mehrfachinstallation kann mehr als eine Version eines Produktes bzw. eines Korrekturstandes auf POSIX installiert sein.

Präfixe für unterschiedliche Plattformen

Für die unterschiedlichen Plattformen werden die PLAM-Bibliotheken mit unterschiedlichen Präfixen ausgeliefert.

– Präfix SIN bzw. SYS* für die /390-Plattform

– Präfix SPU bzw. SPM* für die SPARC-Plattform

– Präfix SKU bzw. SKM* für die X86-Plattform

* Diese Präfixe gelten nur für SOCKETS und NSL.

Installationsskripts

Die PLAM-Bibliotheken enthalten Installationsskripts, die bei der Installation mit dem POISIX-Installationsprogramm üblicherweise folgende Aktionen durchführen:

– Dateiverzeichnisse in POSIX-Dateisystemen anlegen

– Textdateien (Parameter, Skripts, ...) in POSIX-Dateisysteme kopieren

– Verweise auf ausführbare Objekte (LLM) in der PLAM-Bibliothek anlegen

– Ausführbare Objekte (LLMs) in POSIX-Dateisysteme kopieren

– ksh-Skripts für komplexere Verarbeitungen ausführen

Die von Fujitsu ausgelieferten Produkte enthalten passende Installationsskripts.

Wenn Sie Fremdprodukte oder eigene Programmpakte in POSIX installieren möchten, dann müssen Sie weitere Anpassungen vornehmen und eventuell eigene Installationsskripts erstellen, siehe Abschnitt „Private Programmpakte zur Installation vorbereiten“ auf Seite 110.

5.2.3 Das Installationsprogramm im Zusammenspiel mit IMON

Wenn Sie POSIX-Komponenten mittels IMON-Unterstützung installieren (siehe POSIX-Installationsprogramm, IMON-Support = Y), dann gibt es 3 verschiedene Arten der Nutzung:

1. Installation des von IMON voreingestellten Korrekturstandes des Produktes

   In diesem Fall gibt der POSIX-Verwalter weder Version noch Korrekturstand an. Es wird entweder der höchste Korrekturstand des Produktes oder - falls vor dem Start des POSIX-Installationsprogramms ein Kommando /SELECT-PRODUCT-VERSION eingegeben wurde - der durch dieses Kommando ausgewählte Korrekturstand auf POSIX installiert.

2. Installation mit frei gewähltem Korrekturstand des Produktes

   Der POSIX-Verwalter gibt neben dem Produktnamen die Produktversion im Format Vmm.n (m,n: Ziffern) und zusätzlich den Korrekturstand im Format aso (a: Buchstabe; s,o: Ziffern) in der von IMON geforderten Schreibweise an.

3. Installation mit frei gewählter Version und mit dem von IMON voreingestellten Korrekturstand

   Der POSIX-Verwalter gibt neben dem Produktnamen die Produktversion im Format Vmm.n (m,n: Ziffern) an. Es wird entweder der höchste Korrekturstand der angegebenen Version oder - falls vor dem Start des POSIX-Installationsprogramms ein Kommando /SELECT-PRODUCT-VERSION eingegeben wurde - der durch dieses Kommando ausgewählte Korrekturstand auf POSIX installiert.

5.2.4 Multimodale Installation

Multimodale Installation bedeutet, dass ein Produkt so installiert wird, dass es auf unterschiedlichen Plattformen (/390 und SPARC) ablaufen kann bzw. dass Programme für unterschiedliche Plattformen erzeugt werden können (z.B. bei POSIX-SOCKETS). Multimodale Installation ist nur mit IMON-Unterstützung möglich.

Für die multimodale Installation gibt es zwei Installationsvarianten:

- Installationsvariante A

  Es wird nur **ein** Installationsskript ausgeführt, unabhängig davon, welche Plattformen das Produkt unterstützt. Werden mehrere Plattformen unterstützt, dann sind alle Installationsskripts identisch. Für diese Variante gelten folgende Konventionen:

IMON Target	**Logical Id**	**Name des Installationsskripts**
beliebig	SINLIB	INSTALL.<produktname>.<produktversion>

Geeignet ist diese Installationsvariante für alle Produkte, deren Objekte in POSIX plattformunabhängig sind wie z.B. NFS und CRTE.

Bitte beachten Sie, dass Verweise in POSIX auf ausführbare Objekte (LLM) in PLAM-Bibliotheken plattformunabhängig sind, nicht aber ein LLM, das in ein POSIX-Dateisystem kopiert wurde.

- Installationsvariante B

  Für jede vom Produkt unterstützte Plattform wird ein Installationsskript ausgeführt. Die Installationsskripts können sich z.B. dadurch unterscheiden, dass LLMs in unterschiedliche Pfade in POSIX installiert werden müssen. Beispiele dafür sind POSIX-NSL, POSIX-SOCKETS, POSIX-SH.

  Bei Installationsvariante B müssen in jeder PLAM-Bibliothek plattformspezifische Installationsskripts vorhanden sein. Für diese Variante gelten folgende Konventionen:

IMON Target	Logical Id	Name des Installationsskripts
S	SINLIB	INSTALL.<produktname>.<produktversion>.390
P	SINLIB	INSTALL.<produktname>.<produktversion>.SP04
K	SINLIB	INSTALL.<produktname>.<produktversion>.X86

Für die Produkte POSIX-NSL und POSIX-SOCKETS gilt aus Kompatibilitätsgründen folgende Ausnahme:

IMON Target	Logical Id
S, P, K	SYSLIB

Zusammenspiel mit dem POSIX-Installationsprogramm

Bei der Installation mit dem POSIX-Installationsprogramm genügt die Angabe des Produktnamens, dies entspricht Variante 1. in Abschnitt „Das Installationsprogramm im Zusammenspiel mit IMON“ auf Seite 108. Welche Installationsvariante ausgeführt wird, hängt vom IMON-Target ab:

- Ist Target A (ANY) belegt, wird immer die Installationsvariante A ausgeführt, auch wenn noch weitere Targets vorhanden sind. Die Verweise auf ausführbare Objekte (LLM) in PLAM-Bibliotheken werden entsprechend eingerichtet, sodass in der aktuellen Ablaufumgebung immer die für die aktuelle Plattform definierten LLMs bzw. die LLMs des Targets A gestartet werden. Das Produkt ist in jeder Umgebung einsetzbar.
- Ist Target A (ANY) nicht belegt, dann gilt:
  - Installationsvariante A wird ausgeführt, wenn nur ein Target belegt ist oder keine plattformspezifischen Installationsskripts existieren.

– Installationsvariante B wird ausgeführt, wenn mehr als ein Target belegt ist und wenn plattformspezifische Installationsskripts existieren. Maßgeblich für die Existenz plattformspezifischer Installationsskripts ist die PLAM-Bibliothek des ersten belegten Targets in obiger Reihenfolge.

Unabhängig von der ausgeführten Installationsvariante ist das Produkt nur in den Umgebungen einsetzbar, die in IMON definiert waren. In anderen Umgebungen wird von POSIX die Errno ENOEXEC gemeldet, wenn es sich um einen Programmaufruf über einen Verweis auf ausführbare Objekte (LLM) in PLAM-Bibliotheken handelt.

5.2.5 Produktinstallation ohne IMON-Unterstützung

Wenn Sie POSIX-Produkte ohne IMON-Unterstützung installieren möchten (siehe POSIX-Installationsprogramm, IMON-Support = N), dann verlangt das POSIX-Installationsprogramm den Produktnamen, die Produktversion und die BS2000-Kennung, um daraus den Namen der PLAM-Bibliothek zu bilden, aus der die Produktteile in POSIX installiert werden.

Bei einer Produktinstallation ohne IMON-Unterstützung ist keine multimodale Installation möglich, d.h. das Produkt wird immer nur für eine Plattform installiert.

5.2.6 Private Programmpakte zur Installation vorbereiten

Private Programmpakete oder Programmpakete anderer Hersteller müssen zuerst in die nachfolgend beschriebene Form gebracht werden, bevor sie mit dem POSIX-Installationsprogramm installiert werden können:

- Die Produktbestandteile müssen in einer in einer PLAM-Bibliothek vorliegen.
- Die PLAM-Bibliothek muss den produktspezifischen Namen <Präfix>LIB.<product>.<version>.[<package>] besitzen.
- Die ausführbaren Programme müssen als L-Elemente abgelegt sein.
- Header-Files, Shellskripts und sonstige Bausteine wie Textdateien müssen als S-Elemente abgelegt sein.
- Die PLAM-Bibliothek muss folgende produktspezifischen Installations- und Deinstallationsskripts als S-Element enthalten:
  – INSTALL.<product>.<version> .[<package>]
  – DELETE.<product>.<version>.[<package>]

  Diese Skripts beschreiben den Ablageort jedes Produktbestandteils im POSIX-Dateisystem und liefern weitere Informationen, die die Ablage betreffen. Der Aufbau dieser Skripts ist im nächsten Abschnitt beschrieben.

Format von Installations- und Deinstallationsskript

Die einzelnen Zeilen der Installations- und Deinstallationsskripts müssen folgendes Format besitzen:

```
Element:Kennbuchstabe:Pfadname:Linkname:Zugriff:Benutzernummer:Gruppennummer
```

Die Spaltenbreite ist variabel. Das Trennzeichen „:“ muss auch angegeben werden, wenn kein Wert angegeben wird. Kommentarzeilen beginnen mit „#“.

Die Einträge für `Kennbuchstabe` und `Zugriff` werden im Folgenden näher erläutert.

- `Kennbuchstabe` kennzeichnet die Installations-Teilfunktion (alphabetisch geordnet):

  b Eine Binärdatei (Plamelement-Typ X) wird unter dem angegebenen Namen angelegt

  d Das Dateiverzeichnis, das im Pfadnamen angegeben ist, wird neu eingerichtet

  f Das Kommando wird unter dem angegebenen Pfadnamen angelegt (Verweis auf Element in PLAM-Bibliothek)

  i Gibt den Installationspfad an. Der i-Eintrag muss die **erste** Anweisungszeile sein!

  l Für den angegebenen Linknamen wird ein Hard-Link angelegt

  m Das Kommando wird unter dem angegebenen Pfadnamen angelegt (als LLM in UFS)

  o Eintrag für Dateien, die entfernt werden sollen

  p Eine Prozedur (`Element`) wird unter dem angegebenen Pfadnamen angelegt

  r Das Skript (Prozedur) mit dem angegebenen Pfadnamen wird ausgeführt

  s Für den angegebenen Linknamen wird ein symbolischer Link angelegt

  u Codierte T-Dateien für *iconv*

  v Eintrag für Dateiverzeichnisse, die entfernt werden sollen

- `Zugriff` stellt die Zugriffsberechtigung für Eigentümer, Gruppe und Andere dar (oktal).

Umgebungsvariablen im Installations- und Deinstallationsskript

In Installationsskripts können Sie folgende zusätzlichen Umgebungsvariablen verwenden:

USER BS2000-Benutzerkennung, unter der die Installation gestartet wurde.

IPATH Installationspfad im POSIX-Dateisystem.
Wenn IPATH gleich der leeren Zeichenkette ist, ist der Installationspfad gleich "/". Wenn kein Installationspfad durch Angabe von :i: definiert wurde (siehe unten), dann wird $IPATH nicht interpretiert, d.h. der String „$IPATH" ist dann Teil eines Pfad- bzw. Linknamens.

IUID BS2000-Installationskennung
Entweder die *installation userid* aus der Dialogmaske *BS2000 POSIX package installation* oder die BS2000 -Benutzerkennung aus dem vollständigen Dateinamen, den IMON für die Logical-ID%SINLIB des zu installierenden Produktes meldet. Das führende Dollarzeichen ist Bestandteil des Strings.

Installationspfad im Installations- und Deinstallationsskript

Mittels der Kennzeichens „i" geben Sie den Installationspfad für die Komponenten eines Produktes im POSIX-Dateisystem an. Die Definition eines Installationspfades muss die erste Anweisungszeile im Installationsskript sein, sonst ist sie wirkungslos.

Ein solcher Eintrag hat folgende Syntax:

```
:i:installationspfad:Zugriff:Benutzernummer:Gruppennummer
```

Dabei bedeuten:

`installationspfad` Vollständiger Pfadname eines Verzeichnisses im POSIX-Dateisystem.

`Zugriff` Zugriffsrechte, mit denen `installationspfad` versehen wird.

`Benutzernummer` POSIX User-Id des Eigentümers von `installationspfad`

`Gruppennummer` POSIX Group-Id des Eigentümers von `installationspfad`

Mit diesem Mechanismus ist es möglich, ein Produkt mehrfach im POSIX-Dateisystem zu installieren.

Die Angabe eines Installationspfades im Installationsskript bewirkt Folgendes:

- Bei der Installation im Dialog wird dieser Installationspfad in der Installationsmaske des POSIX-Installationsprogramms angezeigt. Sie können den Installationspfad dort nach Belieben ändern. Wirksam wird der Wert, der zuletzt in der Dialogmaske steht.

- Bei Installation im Batch gilt dieser Installationspfad, wenn die Parameterdatei keinen Installationspfad enthält. Wurde in der Parameterdatei in der entsprechenden Anweisungszeile ein Installationspfad angegeben, dann wird der Installationspfad aus der Parameterdatei übernommen, siehe Abschnitt „Install Packages on POSIX (Programmpakete hinzufügen)“ auf Seite 142.
- Die Variable $IPATH im Installationsskript wird durch den aktuellen Wert des Installationspfades ersetzt. Diese gilt für jede Angabe eines Pfad- bzw. Linknamens, in der $IPATH vorangestellt ist (außer natürlich in der Definition des Installationspfades).

Beispiel

```
:i:/opt/C/022A10::0755:2:2 # default installpath
:d:$IPATH/bin::0755:2:2 # subdirectory
C89:f:$IPATH/bin/c89::0755:2:2 # command
```

Im Deinstallationsskript ist die Definition eines Installationspfades nicht sinnvoll. Die Angabe :i: wird ignoriert, wenn sie syntaktisch korrekt ist.

Die Variable $IPATH im Deinstallationsskript wird wie folgt ersetzt:

- im Dialog durch Auswahl der entsprechenden Position in der Deinstallationsmaske, in der alle aktuell installierten Produkte aufgelistet sind.
- im Batch durch den Installationspfad, der in der Anweisungszeile der Parameterdatei angegeben ist.

Meldungen, Eingaben und Rückkehrcodes bei Installationsskripts

In Installationsskripts können Shellskripts ausgeführt werden. Aus diesen Shellskripts heraus können Meldungen ausgegeben werden und in diese Shellskripts können Eingaben getätigt werden. Für die Ein-/Ausgaben gilt je nach Art der Installation Folgendes:

- Bei Installation im Dialog werden die Eingaben vom Terminal gelesen und die Ausgaben auf das Terminal ausgegeben, d.h. *stdin*, *stdout* und *stderr* werden auf das Terminal umgelenkt.
- Bei Installation im Dialog werden die Eingaben aus einer so genannten Response-Datei eingelesen. Diese Datei muss im selben Verzeichnis stehen, aus dem heraus das Shellskript gestartet wird. Der Name muss aus dem Namen des Shellskripts mit angehängtem *.response* bestehen.

  Die Ausgaben (*stdout* und *stderr*) werden immer auf SYSOUT ausgegeben.

Steueranweisungen

Ein Shellskript kann Steueranweisungen enthalten, die im Falle eines Rückkehrcodes ungleich 0 den Endestatus melden und den weiteren Verlauf der Installation bestimmen. Diese Steueranweisungen beginnen immer in der 1. Zeile mit einer Raute und einem Ausrufezeichen und können an beliebiger Stelle im Shellskript stehen.

Folgende Steueranweisungen sind möglich:

`#!REPORT_SHELLSCRIPT_ERROR={ON | OFF}`

`ON` Ist der Rückkehrcode ungleich 0, dann wird eine Meldung ausgegeben (Standard). Diese Meldung hat folgendes Format:

"`shell script` *skriptname* `reports error` *exitvalue*"

Die Bedeutung muss im jeweiligen Kontext in der Produktbeschreibung erläutert werden. Um Mehrdeutigkeiten zu vermeiden, sollten die Rückkehrcodes der Shellskripts zwischen 128 – 255 liegen, da auch die POSIX-Shell Rückkehrcodes setzt.

`OFF` Es wird keine Meldung ausgegeben.

`#!EXIT_ON_SHELLSCRIPT_ERROR={ON | OFF}`

`ON` Ist der Rückkehrcode ungleich 0, dann wird die Installation des Produkts abgebrochen.

`OFF` Die Installation des Produkts wird fortgesetzt (Standard).

Beispiel

Das Produkt *Beispiel* (Version *123*) wird aus der Produktbibliothek *SINLIB.BEISPIEL.123* installiert. Das Installationsskript *INSTALL.BEISPIEL.123* enthält u. a. folgende Zeilen:

```
:i:/tmp/beispiel.install::0755:2:2
beispiel.sh:p:SIPATH/script::0555:2:2
beispiel.rs:p:SIPATH/script.response::0555:2:2
#!REPORT_SHELLSCRIPT_ERROR=ON
#!EXIT_ON_SHELLSCRIPT_ERROR=OFF
:r:SIPATH/script::::
```

5.3 Erstmalige Installation von POSIX

Bei der Erstinstallation werden ein neues POSIX-root- und -var-Dateisystem eingerichtet.

Die Verfahrensschritte bei einer Erstinstallation sind:

1. SOLIS-Lieferung von POSIX einspielen mit IMON

   Nach der Installation des Produkts im BS2000 mit dem SOLIS/IMON-Verfahren sind u.a. der Subsystemkatalog, das Software Configuration Inventory (SCI), die Message- und SDF-Syntaxdateien aktualisiert.

2. Vorbereitungen treffen

   Bei der erstmaligen Installation von POSIX auf einer Anlage, auf der noch **kein** POSIX lief, sind nach der Produktinstallion im BS2000 noch einige Vorkehrungen zu treffen, siehe Abschnitt Abschnitt „Vorbereitende Schritte zur Erstinstallation“ auf Seite 116.

3. POSIX-Erstinstallation mit dem POSIX-Installationsprogramm durchführen, siehe Abschnitt Abschnitt „Erstinstallation mit dem POSIX-Installationsprogramm durchführen“ auf Seite 117.

5.3.1 Vorbereitende Schritte zur Erstinstallation

Wenn auf der Anlage noch kein POSIX lief, dann müssen Sie folgende Schritte unternehmen:

– Entfernen Sie vor dem Start des Subsystems POSIX das CPU-Limit der Systemkennung SYSROOT.

  Dies ist notwendig, damit die als Batch-Tasks unter der Kennung SYSROOT gestarteten POSIX-Dämonen ohne CPU-Limit (NTL) und somit während der gesamten Lebensdauer des POSIX-Subsystems laufen.

  Es gibt zwei Möglichkeiten:

  1. Stellen Sie das Account-Privileg der Kennung SYSROOT auf NO-CPU-LIMIT= *YES ein, falls die Standard-Batch-Jobklasse (bzw. ab OSD V9.0 optional die Standard-POSIX-Jobklasse) der Kennung SYSROOT mit NO-CPU-LIMIT=*NO definiert ist.

     Beispiel

     ```
     /MODIFY-USER-ATTRIBUTES USER-IDENTIFICATION=SYSROOT, -
     /       ACCOUNT-ATTRIBUTES=*MODIFY(ACCOUNT=SYSACC, -
     /           PRIVILEGE=*PARAMETERS(NO-CPU-LIMIT=*YES))
     ```

  2. Alternativ können Sie die der Kennung SYSROOT zugewiesenen Standard-Jobklassen (Batch und ab OSD V9.0 POSIX) mit NO-CPU-LIMIT=*YES definieren.

– Entsperren Sie die Kennung SYSROOT:

  ```
  /UNLOCK-USER USER-ID=SYSROOT
  ```

5.3.2 Erstinstallation mit dem POSIX-Installationsprogramm durchführen

Nach der Installation von POSIX im BS2000 müssen Sie noch POSIX und die gewünschte Software über das POSIX-Installationsprogramm unter der Benutzerkennung TSOS im POSIX-Dateisystem installieren und nötigenfalls Dateisysteme erzeugen und bearbeiten. Bei der Erstinstallation von POSIX wird die root-Berechtigung (Benutzernummer 0, Gruppennummer 0) automatisch an TSOS vergeben.

Die Erstinstallation müssen Sie in folgenden Schritten unter der Benutzerkennung TSOS durchgeführen:

1. Sicherstellen, dass das POSIX-Subsystem **nicht** gestartet ist und dass die POSIX-Informationsdatei SYSSSI.POSIX-BC.<version> schreibbar ist (ACCESS=*WRITE).
2. POSIX-Installationsprogramm aufrufen:

   ```
   /START-POSIX-INSTALLATION
   ```

   Sie können das Programm auch im Batch aufrufen. Die genaue Beschreibung der Masken und Parameter finden Sie auf Seite 126 (Dialog) bzw. Seite 136 (Batch).
3. Anlegen und Einhängen der neuen Dateisysteme root und var:

   Im POSIX-Installationsprogramm die Option „Install POSIX subsystem“ auswählen und in der Maske die Angaben zur Behälterdatei und zum root-Dateisystem eintragen. Das root-Dateisystem muss mindestens 4096 PAM-Seiten groß sein und muss unter SYSROOT angelegt werden. Wenn mehrere Produkte installiert werden, muss es größer angelegt werden (empfohlene Größe: 20 000 PAM-Seiten).

   Danach wird dieselbe Maske wieder angezeigt, um die Angaben zum var-Dateisystem einzutragen. Das var-Dateisystem muss mindestens 4096 PAM-Seiten groß sein und sollte unter SYSROOT des HOME-Pubsets angelegt werden. Wenn mehrere Produkte installiert werden, muss es größer angelegt werden (empfohlene Größe: 20 000 PAM-Seiten).

   Nähere Angaben zu den Dateisystemen root und var finden Sie in den entsprechenden Produkthandbüchern und in der Freigabemitteilung zu POSIX-BC.

Anschließend werden wichtige Dateiverzeichnisse und Dateien automatisch aus dem generischen root-Dateisystem in das neu angelegte root- und var-Dateisystem kopiert. Die generierten Dateiverzeichnisse, Geräte- und Verwaltungsdateien werden protokolliert. Nachdem alle Dateiverzeichnisse und Dateien kopiert worden sind, ist das Arbeiten mit POSIX über die Basis-Shell möglich. Nachdem der automatische Restart des POSIX-Subsystems erfolgt ist, kann das BS2000-Kommando /START-POSIX-SHELL (siehe Seite 263) eingegeben werden.

5.3.3 Installation weiterer Software

Nach der Installation von POSIX-BC können Sie noch weitere Produkte in POSIX installieren.

Folgende Tabelle gibt einen Überblick über alle Liefereinheiten, die in POSIX installierbare Programmpakete (Release Units) enthalten. Die Release-Units enthalten Bibliotheken zur Installation unter POSIX mit dem Präfix <xxx>LIB, wobei <xxx> plattformabhängig ist und SIN, SPU, SKU oder (nur im Falle von POSIX-SOCKETS und POSIX-NSL) SYS, SPM, SKM sein kann.

Es gibt mittlerweile eine Reihe von Produkten, die sich im Rahmen der SOLIS/IMON-Produktinstallation automatisch in POSIX installieren lassen. Die Liefereinheiten dieser Produkte enthalten sog. POSIX-Items vom Typ *PS.

Liefereinheit	**Release Unit**	**Kurzbeschreibung / Funktionalität in POSIX**	***PS Item ?**
APACHE (GA)	APACHE	Apache Webserver	N
	PERL	Scriptsprache perl	N
	TOMCAT	JAVA-Servlet-Unterstützung	N
BS2OSD (GA)	SANCHECK (ab 2.0)	Überprüfung der SAN-Konfiguration (Storage Area Network)	Y
COBOL2000	COBOL2000	COBOL2000-Compiler (cobol2000)	Y
COBOL85	COBOL85	COBO85L-Compiler (cobol85)	N
CPP	CPP	C/C++-Compiler (cc, c89, CC)	Y
CRTE	CRTE	Common RunTime Environment für C, C++ und Cobol, Include Header	Y
CRTE-BAS (GA)	POSIX-HEADER	Include-Header für POSIX-Bibliotheksfunktionen	Y
DPRINTCL	DPRINTCL	Distributed Print Services, Gateway-Komponente für BSD-LPD-Clients	N
HIPLEX-AF	HIPLEX-AF	Highly Integrated System Complex, Failover-Manager, MirrorView- oder Live-Monitor-Funktion	Y
IMON (GA)	IMON-BAS	Installations Monitor, rc-Skript für automatische POSIX-Paketinstallation	Y

Liefereinheit	Release Unit	Kurzbeschreibung / Funktionalität in POSIX	*PS Item ?
INETSERV	MAIL – IMAP – POSTFIX	 Internet Message Access Protocol SMTP-Server (Simple Mail Transfer Protocol)	N
	TCP-IP-AP	File Transfer Protokoll ftp	N
	TCP-IP-SV – DNS – NAMED – NTP – OPENSSH	 DNS Resolver DNS Server Client und Server (Network Time Protocol) Secure Shell	N
JENV (GA)	JENV	Java Environment	Y
NFS	NFS	Network File System	Y
OMNIS	OMNIS	Web-Oberfläche für Internet-Zugang	N
ONETSERV	BCAM	BCAM-Subagent (SNMP)	N
OPENFT	OPENFT	ft-Client für POSIX-Schnittstellen ncopy, ft etc.	Y
POSIX (GA)	POSIX-BC	Basis Shell	Y
	POSIX-ADDON-LIB	UNIX-/BS2000-spezifische Erweiterungen zu den POSIX-Bibliotheksfunktionen	Y
	POSIX-NSL	TLI-, RPC- und XDR-Funktionen	Y
	POSIX-SH	erweiterte Shell	Y
	POSIX-SOCKETS	Socket- und XTI-Funktionen	Y
SBA-BS2	SBA-BS2	SNMP-Basic-Agent	Y
SCCA-BS2	SCCA-BS2	Storage Control Center Agent	Y
SESAM-SQL	SESAM-SQL – SNMP-SA	SNMP-Subagent SESAM-SQL	N
SHC-OSD	STORMAN	FibreCAT-CX-Unterstützung (SQ/SX-Server)	Y
	SYMAPI	Symmetrix Application Programming Interface	Y
SM2-TOOLS	SM2-TOOLS	SNMP-Subagent OpenSM2	N
SSA-OUTM-BS2	SSA-OUTM-BS2	SNMP-Subagent OpenUTM	Y
SSA-SM2-BS2	SSA-SM2-BS2	SNMP-Subagent SM2	Y
SSC-BS2	SSC-BS2	SNMP-Standard-Collection, Erw. Basic Agent	Y
WEBTRANS-OSD (GA)	WEBTRANS-OSD	BS2000-Dialoganwendungen in das WWW anbinden	N
WEBTRANS-UTM	WEBTRANS-UTM	OpenUTM-Anwendungen in das WWW anbinden	N

Erweiterte POSIX-Shell installieren

Es wird empfohlen, generell nach einer Erstinstallation von POSIX eine Paket-Installation von POSIX-SH (erweiterte Shell) durchzuführen. Nur so ist der komplette Umfang der POSIX-Shellkommandos verfügbar.

C/C++-Programmierumgebung installieren

Wollen Sie C/C++-Programme in der POSIX-Shell entwickeln (POSIX-Kommandos *cc*, *c89* oder *CC*), so müssen Sie zusätzlich CRTE, CPP und POSIX-HEADER installieren.

5.3.4 Hinweise zur automatischen POSIX-Paketinstallation mit IMON

Voraussetzungen und allgemeines Konzept

Voraussetzung für die automatische Paketinstallation ist, dass

– IMON-BAS in POSIX installiert ist,

– beim Generieren der IMON-Installationsprozedur mit INSTALL-UNITS oder im Menü-Modus für die Liefereinheit das "POSIX-Processing" eingeschaltet wird und

– die Liefereinheit des Produkts POSIX-Items vom Typ *PS enthält.

Bei Lieferung einer neuen Produktversion erfolgen generell die Schritte

1. Deinstallation der in POSIX installierten Vorgängerversion
2. Installation der neuen Produktversion in POSIX

Standardmäßig werden auch Produkte in POSIX installiert, die zuvor in POSIX noch nicht installiert waren. In diesem Fall entfällt die Deinstallation.

Die automatische POSIX-Paketinstallation lässt sich bei Einspielen der SOLIS-Lieferung (INSTALL-UNITS) ausschalten.

Ablauf der IMON-Installation im BS2000

Beim Installieren der Produkte im BS2000 werden von der IMON-Installationsprozedur folgende Schritte durchgeführt:

1. Die Datei $SYSROOT.POSIX.CONFIGURATION wird erzeugt bzw. aktualisiert. Sie enthält eine Liste aller Pakete, die in POSIX installiert sind.
2. Es werden Dateien angelegt, die als Eingabedateien für die automatische POSIX-Paketinstallation mit dem POSIX-Installationsprogramm (im Batch-Modus) dienen:

   $SYSROOT.IMON.ACTIONS.REM (Delete-Packages)
   $SYSROOT.IMON.ACTIONS.ADD (Install-Packages)

   Für den Aufbau der REM-Datei ist der Inhalt der CONFIGURATION-Datei relevant. Alle genannten Dateien werden in der Kennung SYSROOT auf dem Home-Pubset erstellt.

Durchführen der automatischen Paketinstallation in POSIX

Die Paketinstallation selbst (Deinstallation der alten Pakete und Installation der neuen Pakete) erfolgt automatisch durch IMON entweder beim nächsten POSIX-Neustart oder im Rahmen der dynamischen Aktivierung mit ACTIVATE-UNITS. Treten bei der Paketinstallation Probleme auf, werden diese in der Datei */var/sadm/pkg/insterr* protokolliert (siehe „Protokollierung von Fehlern in der Parameterdatei (MAINCODE = POS2956):“ auf Seite 262).

- Paketinstallation beim POSIX-Neustart:

  Die Deinstallation/Installation erfolgt über das Skript */etc/rc2.d/S05imon*. Eingabedateien für das POSIX-Installationsprogramm sind die mit der Installationsprozedur aufgebauten IMON.ACTIONS-Dateien. Alle dort enthaltenen Produkte müssen aktiviert sein. Dies ist nach einem BS2000-Neustart gegeben.

  Nach Abschluss der Installation werden die IMON.ACTIONS-Dateien stets mit dem Suffix ".SAVE" versehen.

- Paketinstallation bei ACTIVATE-UNITS:

  ACTIVATE-UNITS für die Liefereinheit BS2GA.POSIX bewirkt (sofern das POSIX-Subsystem in der Lieferung enthalten ist) einen automatischen POSIX-Neustart. Die Paketinstallation erfolgt dann anschließend beim POSIX-Neustart (siehe oben).

  Bei allen anderen Liefereinheiten läuft die Paketinstallation wie folgt ab: Bei Ausführen der Aktivierungsprozedur verschiebt IMON die Einträge der zu aktivierenden Produkte in temporäre IMON.ACTIONS-Dateien (mit dem Dateinamens-Suffix #) und führt die Deinstallation bzw. Installation der Pakete anhand dieser Dateien durch.

  Im Erfolgsfall werden die Einträge für die installierten Produkte aus den von der IMON-Installationsprozedur aufgebauten IMON.ACTIONS-Dateien entfernt. Der ursprüngliche Stand dieser Dateien steht dann in einer Sicherungskopie mit dem Suffix ".SAV". Im Fehlerfall bleiben die IMON.ACTIONS-Dateien unverändert.

  

  ACHTUNG:

  Die dynamische Aktivierung der Liefereinheit POSIX sollte immer erst im Anschluss an die übrigen Produkte durchgeführt werden, da sonst alle auf die POSIX-Liefereinheit folgenden Produkte zum Zeitpunkt des automatischen POSIX-Neustarts noch nicht aktiviert sind und ihre Paketinstallation deswegen fehl schlägt.

Vorgehensweise bei der dynamischen Aktivierung im laufenden Betrieb

Es muss grundsätzlich unterschieden werden, ob die Lieferung die Installationseinheiten POSIX-BC (Liefereinheit BS2GA.POSIX) oder CRTE-BASYS (Liefereinheit BS2GA.CRTE-BAS) enthält oder nicht. Abhängig davon sind folgende Varianten möglich:

1. Bei Lieferungen ohne POSIX-BC und CRTE-BASYS ist kein POSIX-Neustart erforderlich. Die automatische Paketinstallation erfolgt im Rahmen von ACTIVATE-UNITS im laufenden POSIX-Betrieb.
2. Bei Lieferungen mit POSIX-BC oder CRTE-BASYS ist ein POSIX-Neustart erforderlich. Hier sind zwei Vorgehensweisen möglich:

   a) Gemeinsame Installation aller Produkte
      – POSIX beenden
      – alle Liefereinheiten im BS2000 installieren und mit ACTIVATE-UNITS aktivieren
      – POSIX neu starten (automatische Paketinstallation erfolgt im Rahmen des POSIX-Neustarts)

   b) Getrennte Installation von POSIX-BC/CRTE-BASYS und anderen Produkten
      – Bei laufendem POSIX alle Liefereinheiten, außer BS2GA.POSIX und BS2GA.CRTE-BAS, im BS2000 installieren
      – diese Liefereinheiten mit ACTIVATE-UNITS aktivieren (automatische Paketinstallation erfolgt im Rahmen von ACTIVATE-UNITS)
      – POSIX beenden
      – BS2GA.POSIX und/oder BS2GA.CRTE-BAS im BS2000 installieren und mit ACTIVATE-UNITS aktivieren
      – POSIX neu starten (automatische Paketinstallation beim POSIX-Neustart).

5.4 Upgrade-Installation von POSIX

Die POSIX-Upgrade-Installation ist erforderlich, falls Sie POSIX bereits installiert haben und die bestehenden Dateisysteme root und var beibehalten wollen.

Die folgenden Verfahrensschritte beschreiben sowohl den Upgrade auf einen neuen Korrekturstand von POSIX (siehe Abschnitt „Upgrade-Installation bei einem neuen POSIX-Korrekturstand“ unten) als auch einen POSIX-Versionswechsel, der z.B. durch einen Wechsel der BS2000/OSD-Version notwendig ist (siehe Abschnitt „Upgrade-Installation bei einer neuen POSIX-Version“ auf Seite 125).

5.4.1 Upgrade-Installation bei einem neuen POSIX-Korrekturstand

Dazu sind folgende Schritte notwendig:

1. Spielen Sie die SOLIS-Auslieferung des neuen POSIX-Korrekturstandes ein.

   Die neu ausgelieferte POSIX-Parameterdatei wird unter dem Namen SYSSSI.POSIX-BC.<version>.NEW abgelegt, d.h. es wird standardmäßig die alte SYSSSI-Parameterdatei (ohne das Suffix NEW) verwendet, in der u.a. bereits der Name des Root-Dateisystems unter ROOTFSNAME eingetragen ist.

2. Starten Sie das Subsystems POSIX (/START-SUBSYSTEM POSIX).

   Die neue POSIX-Version wird dabei mit den alten Dateisystemen hochgefahren.

3. Nach 'POSIX ready' (Meldung an Konsole):

   – Starten Sie das POSIX-Installationsprogramm (/START-POSIX-INSTALLATION) und wählen Sie die Option 'Install packages on POSIX'.

   – Installieren Sie das Produkt 'POSIX-BC'.

   – Beenden Sie danach das POSIX-Installationsprogramm.

5.4.2 Upgrade-Installation bei einer neuen POSIX-Version

Die Upgrade-Installation braucht gleichzeitigen Zugriff auf die POSIX-Produkt-Dateien <oldversion> und <newversion>.

Dazu sind folgende Schritte notwendig:

1. Erstellung eines neuen HOME-Pubsets mit weiterer Nutzung der POSIX-Daten.

   Kopieren Sie die alten root- und var-Dateisysteme sowie die alten POSIX-Produktdateien (Dateinamen S*.POSIX-BC.<oldver>*) auf das neue HOME-Pubset.

2. Spielen Sie die SOLIS-Auslieferung der neuen POSIX-Version ein und gehen Sie wie folgt vor:

   – Tragen Sie in der Parameterdatei SYSSSI.POSIX-BC.<newversion> in der Zeile des Parameters ROOTFSNAME den Dateinamen des root-Dateisystems ein.

   – Aktivieren Sie die im Subsystem geänderte Version des Subsystems POSIX, indem Sie das BS2000-Systems beenden und wieder hochfahren.

3. Starten Sie das Subsystem POSIX (/START-SUBSYSTEM POSIX).

   Die neue POSIX-Version wird dabei mit den alten Dateisystemen hochgefahren.

4. Nach 'POSIX ready' (Meldung an Konsole):

   – Starten Sie das POSIX-Installationsprogramm (/START-POSIX-INSTALLATION) und wählen Sie die Option 'Install packages on POSIX'.

   – Installieren Sie das Produkt 'POSIX-BC' mit der neuen Versionsnummer <newvers>.

   – Beenden Sie danach das POSIX-Installationsprogramm.

   Anschließend können die Produktdateien der alten POSIX-Version gelöscht werden.

5.5 POSIX-Installationsprogramm im Dialog

Das POSIX-Installationsprogramm können Sie offline (POSIX nicht gestartet) und online (POSIX gestartet) aufrufen. Die Hauptmasken unterscheiden sich dabei.

- Installationsprogramm offline aufrufen

```
/START-POSIX-INSTALLATION
```

```
                    BS2000 POSIX installation program


                              Please select

                    Install POSIX subsystem

                    Expand POSIX filesystems



Select:  MAR + DUE                                  Finish installation: F2
Help  :  F1
```

Bild 22: Hauptmaske des POSIX-Installationsprogramms - offline

Das POSIX-Installationsprogramm stellt dann folgende Optionen zur Verfügung:

– Install POSIX subsystem (Subsystem POSIX neu einrichten ≙ Erstinstallation)

– Expand POSIX filesystems (POSIX-Dateisystem erweitern)

- Installationsprogramm online aufrufen

```
/START-POSIX-INSTALLATION
```

```
                    BS2000 POSIX installation program


                              Please select


                    Administrate POSIX filesystems

                    Install packages from POSIX

                    Delete packages from POSIX



Select:  MAR + DUE                                  Finish installation: F2
Help  :  F1
```

Bild 23: Hauptmaske des POSIX-Installationsprogramms - online

Wenn das Subsystem POSIX aktiv ist, gibt es folgenden Optionen:

– Administrate POSIX filesystems (POSIX-Dateisystem verwalten)

– Install packages on POSIX (Programmpakete hinzufügen)

– Delete packages from POSIX (Programmpakete entfernen)

Um eine der Optionen auszuwählen, müssen Sie die Option mit der Schreibmarke anwählen, durch Angabe eines Zeichens oder mit der MAR-Taste markieren und die Auswahl mit DUE bestätigen.

Mit der Funktionstaste F1 erhalten Sie Hilfe zur Maske, mit der Funktionstaste F2 beenden Sie das Installationsprogramm. Meldungen und Informationen des Installationsprogramms erhalten Sie in der letzen Bildschirmzeile.

Install POSIX subsystem (Subsystem POSIX neu einrichten)

Mit dieser Option können Sie ein neues Subsystem POSIX einrichten.

Das root- und das var-Dateisystem werden mit den Benutzerangaben angelegt, Dateiverzeichnisse und Dateien werden im POSIX-Dateisystem installiert und das Subsystem POSIX wird mit dem soeben angelegten root-Dateisystem gestartet.

```
                    Definition of BS2000 Container File

BS2000 filename:  $SYSROOT.FS.ROOT

BS2000 filesize:  20000     PAM-Pages

POSIX filesystem? (y/n): Y

================================================================================

                        Definition of POSIX filesystem

Size of filesystem:          PAM pages       Journaling? (y/n):

POSIX mountpoint: /

Automount? (y/n): Y           Mountoptions:

Overwrite existing filesystem? (y/n):       POSIX filesystem marker (y/n): Y

================================================================================
Save definitions:  DUE
Help            :  F1                                              terminate: F2
Indicate name of BS2000 container for the root filesystem
```

Bild 24: Folgemaske zu „Install POSIX subsystem“

Angaben zur BS2000-Behälterdatei

BS2000 filename
Name der PAM-Datei, die als Behälterdatei für das root- bzw. var-Dateisystem benutzt werden soll. Der Dateiname muss die Benutzerkennung SYSROOT enthalten. Wenn die Datei noch nicht existiert, wird sie in der angegebenen Größe erzeugt.

BS2000 filesize
Größe der Behälterdatei in PAM-Seiten (Einheit: 2 KB). Die Mindestgröße muss 4096 PAM-Seiten betragen. Für neu einzurichtende Behälterdateien müssen Sie die gewünschte Größe eintragen. Wenn die Behälterdatei bereits existiert, wird die tatsächliche Größe in das Feld übernommen. In diesem Fall können Sie den Wert nicht ändern.

POSIX filesystem? (y/n)
Beantworten Sie die Frage mit y (ja) oder n (nein). Normalerweise soll die Behälterdatei ein POSIX-Dateisystem enthalten. In besonderen Fällen können Sie aber den Zugriff über das POSIX-Dateisystem umgehen und direkt auf den Datei-Inhalt zugreifen (*raw*-Zugriff).
Die Felder für das root- und var-Dateisystem sind mit y (ja) vorbelegt.

Angaben zum POSIX-Dateisystem:

Size of filesystem
Dieses Feld zeigt die Größe eines bereits vorhandenen Dateisystems in der BS2000-Behälterdatei in PAM-PAGES (2 KB) an. Wenn noch kein Dateisystem existiert, wird die Größe der BS2000-Behälterdatei angezeigt.
Das Feld kann nicht überschrieben werden, da ein Dateisystem in einer BS2000-Behälterdatei immer die Größe der BS2000-Behälterdatei hat.

Journaling? (y/n)
In diesem Feld können Sie festlegen, ob das Dateisystem mit (y) oder ohne (n) Journal montiert werden soll. Wenn Sie keine Angabe machen, ist der Standardwert = n.

POSIX mountpoint
Dateiverzeichnis, in das das POSIX-Dateisystem eingehängt werden soll.
Sie müssen den absoluten Pfadnamen des Dateiverzeichnisses eingeben, wobei der Name mit einem Schrägstrich (/) beginnen muss. Wenn das Dateiverzeichnis noch nicht existiert, wird es vom Programm angelegt.
Die Felder für das root- und var-Dateisystem sind mit / vorbelegt.

Automount? (y/n)
Wenn das Dateisystem sofort und bei jedem Subsystem-Start automatisch eingehängt werden soll, müssen Sie y (ja) eintragen. Wenn Sie das Dateisystem nur einrichten, aber noch nicht benutzen wollen, müssen Sie n (nein) eintragen.
Die Felder für das root- und var-Dateisystem sind mit y (ja) vorbelegt.

Mountoptions
Sie können das Einhängen des Dateisystems parametrisieren. Die entsprechenden Optionen finden Sie beim Kommando *mount* im POSIX-Handbuch „Kommandos“ [1]. Mehrere Optionen sind durch Kommas zu trennen.

Overwrite existing filesystem? (y/n)
Beantworten Sie die Frage mit y (ja) oder n (nein).
Das Feld ist nur aktiviert, wenn die Behälterdatei bereits ein POSIX-Dateisystem enthält. Sie müssen entscheiden, ob das Dateisystem unverändert übernommen werden soll oder ob ein neues Dateisystem erzeugt werden soll. Beim Einrichten von root- oder var-Dateisystem wird dieses Feld nicht aktiviert, da diese Dateisysteme immer überschrieben werden.

POSIX filesystem marker (y/n)
Bedeutung: Dateisystem in POSIX/BS2000 erzeugt.
Die Felder für das root- und var-Dateisystem sind mit Y (ja) vorbelegt.

Wenn der Marker nicht gesetzt ist, wird das Dateisystem unter POSIX als ASCII-Dateisystem betrachtet. D.h., es findet in Abhängigkeit von der Umgebungsvariablen *IO_CONVERSION* eine ASCII-EBCDIC-Konvertierung statt (siehe Abschnitt „Kopieren und Konvertieren von Dateien“ auf Seite 37).

Expand POSIX filesystems (POSIX-Dateisystem erweitern)

Mit dieser Option können Sie jedes POSIX-Dateisystem einschließlich *root* und *var* im Offline-Modus vergrößern.

Sie können alle POSIX-Dateisysteme außer *root* und *var* auch im Online-Modus vergrößern (siehe Seite 132, Kommando *expand*).

```
                        Expand of POSIX filesystem

BS2000 filename: _

   Characteristics         before expand            after expand
------------------------------------------------------------------------------
   BS2000 filesize      ......... PAM-Pages      ......... PAM-Pages
size of filesystem      ......... PAM-Pages      ......... PAM-Pages
            inodes      .........                .........
       free inodes      .........                .........
        datablocks      ......... (4 KB)         ......... (4 KB)
   free datablocks      ......... (4 KB)         ......... (4 KB)

Best value for expand is ......... PAM pages + N * ......... PAM pages (N >= 0)

------------------------------------------------------------------------------


                    Expand value: ......... PAM pages


==============================================================================

Execute expand: DUE              Help: F1                  Terminate: F2
```

Bild 25: Folgemaske zu „Expand POSIX filesystems“

BS2000 filename
In diesem Feld geben Sie den Namen der Behälterdatei für das zu vergrößernde Dateisystem an.

Characteristics before expand / after expand
In diesen Spalten werden die aktuellen Kenndaten des Dateisystems vor und nach der Vergrößerung angezeigt.

Best value for expand is ...
In dieser Zeile wird der optimale Wert für eine Vergrößerung angezeigt (um ungenutzte oder nur teilweise genutzte PAM-Seiten zu vermeiden).

Expand value
In diesem Feld geben Sie an, um wieviele PAM-Seiten das Dateisystem erweitert werden soll. Ein erneutes DUE nach erfolgreicher Erweiterung bewirkt keine erneute Erweiterung, sondern wird ignoriert.

Nach erfolgreicher Erweiterung kann ein neues Dateisystem angegeben werden oder mit *F2* in die Startmaske zurückgekehrt werden.

Administrate POSIX filesystems (POSIX-Dateisysteme verwalten)

Mit dieser Option können Sie vorhandene POSIX-Dateisystem-Einträge ändern, erweitern und löschen sowie neue POSIX-Dateisystem-Einträge erzeugen.

```
                        BS2000 POSIX filesystem table
                                                                  F J A P
   BS2000 filename                                          size  S O M F

_ $SYSROOT.FS.ROOT                                         50001  Y N Y N
_ $SYSROOT.FS.HOME1                                       100008  Y N Y N
_ $SYSROOT.FS.VAR                                          50001  Y N Y N
_ $SYSROOT.FS.BACH                                        100008  Y N Y N
_
_
_
_
_
_
_
_
_
_
_
edit commands:  'a'=append  'm'=modify  'd'=delete  'e'=expand
function keys:  F1=help  F2=terminate              scroll commands: '+'/'-'
command  ====>  ____________________________________________________ more:
```

Bild 26: Folgemaske zu „Administrate POSIX filesystems“

Die Folgemaske zeigt sämtliche BS2000-Dateien, die im Subsystem POSIX als lokale Behälterdateien eingetragen sind. Mit „+“ und „-“ können Sie in der Maske blättern. Jeder Eintrag enthält:

- den BS2000-Dateinamen der Behälterdatei.
- die Größe der Behälterdatei in PAM-Seiten (Einheit: 2 KB).
- die aktuelle Einstellung folgender Parameter:
  FS = Dateisystem-Inside (y/n),
  JO = Journaling (y/n),
  AM = Automount (y/n) und
  PF = POSIX-Dateisystem-Marker (y/n)

Im unteren Teil der Maske befindet sich das Eingabefeld *command*, in dem Sie eines der folgenden Verwaltungskommandos eintragen können:

‘a‘=append
Neuen Eintrag erzeugen
Behälterdatei und ggf. Dateisystem in Liste aufnehmen. Markieren Sie zuvor den Eintrag, hinter den Sie den neuen Eintrag anfügen wollen; andernfalls wird der neue Eintrag an das Ende der Liste angehängt.

Beim Anlegen eines Dateisystems mit der Funktion append hat der Systemverwalter die Wahl, ob er das Dateisystem als „von POSIX erzeugt“ (also als EBCDIC-Dateisystem) markieren will oder nicht (ASCII-Dateisystem). Das root- und das var-Dateisystem werden bei der Erstinstallation automatisch als „von POSIX erzeugt“ markiert.

‘m‘=modify
Markierten Eintrag ändern
Bestimmte Felder des Dateisystem-Eintrags können Sie nachträglich ändern.

‘d‘=delete
Markierten Eintrag löschen
Es wird nur der Eintrag aus der Liste entfernt. Die Behälterdatei und das Dateisystem bleiben unverändert und können später wieder eingetragen oder gelöscht werden.

‘e‘=expand
Markiertes Dateisystem erweitern
Mit diesem Kommando können Sie das ausgewählte Dateisystem vergrößern, vorausgesetzt, es lässt sich demontieren. Bei der Eingabe dieses Kommandos wird dieselbe Maske angezeigt, wie bei der Option „Expand POSIX filesystems (POSIX-Dateisystem erweitern)“ auf Seite 130.

Im Gegensatz zum Offline-Modus gilt:

– Die Dateisysteme *root* und *var* können hier nicht vergrößert werden, da sie immer belegt sind und sich nicht demontieren lassen.

– Das Feld *BS2000 filename* ist bereits mit dem in der Maske *BS2000 POSIX filesystem table* markierten Namen vorbelegt und kann nicht geändert werden.

– Das Dateisystem wird nach erfolgreicher Erweiterung und Betätigen von *F2* wieder montiert, wenn es vor der Erweiterung montiert war. Dies geschieht unabhängig von der Automount-Einstellung.

– Nach Betätigen von *F2* verzweigt das Programm zurück zur Vorgängermaske *BS2000 POSIX filesystem table*.

Install packages on POSIX (POSIX-Programmpakete hinzufügen)

Mit dieser Option können Sie POSIX-Anwenderprogramme und Programmpakete im POSIX-Dateisystem installieren, siehe auch Seite 118.

```
                    BS2000 POSIX package installation


    IMON support ?     : Y  (y) mandatory for official package
                            (n) private package (SINLIB...)

    name of product    :
    package of product :                      (optional for certain products)

    version of product :          (format Vmm.n or mmn)

    correction state   :          (format aso, optional for IMON support)

    installation userid:          (mandatory for no IMON support)




install:  DUE
help   :  F1                                              terminate: F2
```

Bild 27: Folgemaske zu „Install packages on POSIX“

IMON support? (y/n)
Legt fest, ob aus dem SCI installiert wird (IMON support: y) oder von einer privaten Ablagekennung (IMON support: n).
Voreinstellung ist IMON support: y.

name of product
Produktname (Release Unit).

package of product
Paketname, falls das Produkt in Pakete zerlegt ist.

version of product (format Vmm.n or mmn)
Produktversion:
– bei 'IMON support: y' im Format Vmm.n oder mmn (m,n: Ziffern) oder leer
– bei 'IMON support: n' im Format mmn (m,n: Ziffern)

correction state (format aso)
Nur bei 'IMON support: y' und nur zusammen mit 'version of product':
Angabe des Korrekturstandes im Format aso (a: Buchstabe; s,o: Ziffern)
Das Feld muss leer bleiben, wenn 'version of product' leer ist, siehe Fall 1 im Abschnitt „Das Installationsprogramm im Zusammenspiel mit IMON“ auf Seite 108.

installation userid (no IMON support)
Nur bei 'IMON support: n' (sonst leer):
User-ID der privaten Ablagekennung

Bei fehlerhafter Eingabe (z.B. Eingabe von Zeichen in ein „leeres" Feld) wird eine Fehlermeldung ausgegeben und die Maske wird erneut zur Änderung ausgegeben.

Vor der Installation einer neuen Version sollte die alte Version des Programmpakets mit der Option „Delete packages from POSIX“ gelöscht werden. Beachten Sie, dass die erweiterte Shell (Paket-Name POSIX-SH) dabei nicht gelöscht werden kann.

Delete packages from POSIX (POSIX-Programmpakete entfernen)

Mit dieser Option können Sie POSIX-Anwenderprogramme und Programmpakete entfernen.

```
                        BS2000 POSIX package delete


   Product                   Version Package             Date of installation
   NFS                       030                         Jan 27 12:23:41 2012
   /
   POSIX-BC                  080                         Jan 27 11:33:27 2012
   /
   POSIX-SH                  080                         Jan 27 11:35:13 2012
   /
   POSIX-SOCKETS             080                         Jan 27 12:53:51 2012
   /
   POSIX-NSL                 080                         Jan 27 12:58:30 2012
   /









scroll: + (%/+/-/$)
delete: mark product with 'x' and DUE                  help: F1  terminate: F2
```

Bild 28: Folgemaske zu „Delete packages from POSIX“

Die Komponenten, die Sie deinstallieren möchten, müssen Sie markieren. Die Komponenten POSIX-BC und POSIX-SH können nicht deinstalliert werden.

5.6 Automatisierter Ablauf des POSIX-Installationsprogramms

Sie rufen das POSIX-Installationsprogramm für den automatisierten Ablauf (Batch-Modus) wie folgt auf:

```
/START-POSIX-INSTALLATION -
/  INPUT-INTERFACE=*FILE( -
/     FILE-NAME=<parameterdatei>, -
/     ERROR-HANDLING=*PARAMETERS(...) -
/  )
```

Beim Operanden FILE-NAME müssen Sie den Namen einer Parameterdatei angeben, die die Installationsinformationen in der nachfolgend beschriebenen Form enthält.

Mit dem Operanden ERROR-HANDLING kann das weitere Verhalten beim Auftreten eines Fehlers gesteuert werden (ausführliche Informationen hierzu siehe Beschreibung des Kommandos START-POSIX-INSTALLATION auf Seite 260).

Aufbau der Parameterdateien

Eine Parameterdatei besteht aus einer Identifikationszeile, aus einer oder mehreren Anweisungszeilen und wahlweise aus Kommentarzeilen.

Kommentare

Kommentare und Kommentarzeilen sind optional. Sie müssen immer mit „#“ beginnen.

Identifikationszeile

Die erste Zeile in der Parameterdatei, die keine Kommentarzeile ist, muss die Identifikationszeile sein. Dadurch wird ein Zweig für die Installation ausgewählt:

[FirstInstallation]	Subsystem POSIX neu einrichten
oder	
[ExpandFileSystem]	POSIX-Dateisystem erweitern
oder	
[FileSystemAdministration]	POSIX-Dateisystem verwalten
oder	
[PackageInstallation]	POSIX-Programmpakete hinzufügen
oder	
[DeletePackage]	POSIX-Programmpakete entfernen

Die eckigen Klammern müssen Sie immer angeben. Die Zeichenfolge zwischen den Klammern können Sie abkürzen; die Eindeutigkeit muss aber gewährleistet sein. Groß- und Kleinbuchstaben dürfen Sie beliebig verwenden.

Anweisungszeilen

Nach der Identifikationszeile folgen eine oder mehrere Anweisungszeilen, die die notwendigen Parameter abhängig vom Zweig enthalten. Das Trennzeichen “;“ (Semikolon) müssen Sie angeben, auch wenn Sie für einen Parameter keinen Wert angeben.

Install POSIX subsystem (Subsystem POSIX neu einrichten)

Identifikationszeile: **[FirstInstallation]**

Anweisungszeile: **<file>;<size>;<jo>**

Dabei bedeuten:

<file> BS2000-Dateiname der Behälterdatei

<size> BS2000-Dateigröße der Behälterdatei (= Größe des Dateisystems)

<jo> Journaling ? (Y/N) Standard : N

Es sind genau zwei Anweisungszeilen vorgeschrieben:

– Die erste Anweisungszeile definiert das root-Dateisystem

– Die zweite Anweisungszeile definiert das var-Dateisystem.

Bei mehr als zwei Anweisungszeilen werden die überzähligen ignoriert. Bei fehlenden Anweisungszeilen wird die Installation abgebrochen. Die Angabe, ob die Dateisysteme das Journaling unterstützen, ist optional. Standardmäßig ist kein Journaling eingestellt.

Die beiden Anweisungszeilen enthalten die Angaben zur BS2000-Behälterdatei eines POSIX-Dateisystems. Dabei sind zusätzliche Angaben, die zur vollständigen Beschreibung eines Dateisystems nötig sind, entsprechend folgender Tabelle vorgegeben:

Parameter	**root-Dateisystem**	**var-Dateisystem**
Dateisystem-Inside	Y	Y
POSIX-Dateisystem-Marker	Y	Y
POSIX-Mountpoint	/	/var
Automount	Y	Y
Mount-Optionen	-	-
Overwrite	N	N

Beispiel

```
#
# Batch Installationsdatei
#
[FirstInstallation]                  # POSIX Erstinstallation
# <file>;<size>

$SYSROOT.FS.ROOT;20000  # Installation des root-Dateisystems ohne Journaling
$SYSROOT.FS.VAR;50000;Y # Installation des var-Dateisystems mit Journaling
```

Expand POSIX filesystems (POSIX-Dateisysteme erweitern)

Identifikationszeile: **[ExpandFileSystem]**

Anweisungszeile: **<file>;<size>**

Dabei bedeuten:

<file> BS2000-Dateiname der Behälterdatei

<size> Größe der Erweiterung

Die Anweisung wird nur für Dateisysteme ausgeführt, die demontierbar bzw. nicht montiert sind. Um z. B. *root* und *var*-Dateisysteme zu erweitern, muss das POSIX-Subsystem beendet werden, da diese Dateisysteme im laufenden Betrieb nicht demontierbar sind.

Beispiel

```
#
# Batch Installationsdatei
#
[ExpandFileSystem]                  # POSIX-Dateisysteme erweitern
# <file>;<size>

# root-Dateisystem um 10.000 PAM-Seiten erweitern
$SYSROOT.FS.ROOT;10000

# var-Dateisystem um 20.000 PAM-Seiten erweitern
$SYSROOT.FS.VAR;20000
```

Administrate POSIX filesystems (POSIX-Dateisysteme verwalten)

Identifikationszeile: **[FileSystemAdministration]**

Anweisungszeile:
<op>;<file>;<size>;<flag>;<mark>;<mdir>;<auto>;<mopt>;<ov>;<jo>

Dabei bedeuten:

<op>	Editierkommando: a(ppend), m(odify) oder d(elete)
<file>	BS2000-Dateiname der Behälterdatei
<size>	BS2000-Dateigröße der Behälterdatei (=Dateisystem-Größe)
<flag>	Dateisystem-Inside? (Y/N)
<mark>	POSIX-Dateisystem-Marker? (Y/N)
<mdir>	POSIX-Mountpoint
<auto>	Automount? (Y/N)
<mopt>	Mount-Optionen
<ov>	POSIX-Dateisystem überschreiben? (Y/N)
<jo>	Journaling ? (Y/N)

Jede Anweisungszeile enthält das Editier-Kommando und die Angaben zur BS2000-Behälterdatei und zum POSIX-Dateisystem. Sie müssen nicht alle Parameter bei jedem Editier-Kommando angeben. So kann z. B. beim Editier-Kommando m(odify) u.a. die BS2000-Dateigröße nicht geändert werden. Die folgende Tabelle zeigt, welcher Parameter bei welchem Editier-Kommando angegeben werden muss:

Parameter	**a(ppend)**	**m(odify)**	**d(elete)**
BS2000-Dateiname	x	x	x
BS2000-Dateigröße	x	-	-
POSIX-Dateisystem-Inside	dy	-	-
POSIX-Dateisystem-Marker	xy	-	-
POSIX-Mountpoint	xm	o	-
Automount	dy	o	-
Mount-Optionen	dl	o	-
POSIX-Dateisystem überschreiben	xo	-	-
Journaling	dn	o	-

Dabei bedeuten:

-	wird ignoriert oder ist wirkungslos, wenn syntaktisch korrekt angegeben
dy	Standardwert ist Y, wenn der Wert fehlt
dn	Standardwert ist N, wenn der Wert fehlt
dl	Standardwert ist die leere Zeichenkette, wenn der Wert fehlt
o	optional; wenn der Wert fehlt, gilt die aktuelle Einstellung
x	Pflichtangabe
xy	Pflichtangabe, wenn bei POSIX-Dateisystem-Inside Y angegeben wird; sonst ignoriert
xm	Pflichtangabe, wenn bei Automount Y angegeben ist; sonst ignoriert
xo	Pflichteingabe bei Overwrite-Situation; sonst ignoriert

Beispiel

```
#
# Batch Installationsdatei
#
[FileSystemAdministration]           # POSIX-Dateisysteme verwalten
# <op>;<file>;<size>;<flag>;<mark>;<mdir>;<auto>;<mopt>;<ov>

# Erzeuge neues BS2000-POSIX-Dateisystem
# Ein existierendes Dateisystem wird ueberschrieben
append;$SYSROOT.FS.USR;50000;;y;/usr;y;;y

# Loesche Dateisystem
delete;$SYSROOT.FS.USR;

# Erzeuge neues BS2000-POSIX-Dateisystem
# Ein existierendes Dateisystem wird nicht ueberschrieben
append;:PUB:$SYSROOT.FS.USR;50000;;y;/HIPLEX/PUB/usr;y;;n

# Existierendes Dateisystem an /usr/home einhaengen
modify;$SYSROOT.FS.HOME;;;;/usr/home/;y;;y
```

Install Packages on POSIX (Programmpakete hinzufügen)

Identifikationszeile: **[PackageInstallation]**

Anweisungszeile: **<prod>;<imon>;<vers>;<corr>;<uid>;<ipath>**

Dabei bedeuten:

<prod>	Name des Softwarepakets zusammengesetzt aus Produktname und optionalen Paketnamen, falls das Produkt in Pakete zerlegt ist. Syntax: <Produktname>[:<Paketname>]
<imon>	IMON flag, legt fest, ob aus dem SCI installiert wird (Y) oder nicht (N)
<vers>	Produktversion des Softwarepakets: – bei IMON flag= 'Y' im Format Vmm.n oder mmn (m,n: Ziffern) oder leer – bei IMON flag= 'N' im Format mmn (m,n: Ziffern)
<corr>	Angabe des Korrekturstandes im Format aso (a: Buchstabe; s,o: Ziffern) Das Feld muss leer bleiben, wenn 'vers' leer ist, siehe Fall 1 im Abschnitt „Das Installationsprogramm im Zusammenspiel mit IMON“ auf Seite 108.
<uid>	BS2000-Benutzerkennung der Ablagekennung, Pflicht für IMON flag= 'N', wird ignoriert für IMON flag = 'Y'.
<ipath>	Optionaler Installationspfad, falls das Softwarepaket dies unterstützt. Standardwert ist '/'.

Bei IMON flag = 'N' wird der Name der BS2000-PLAM-Bibliothek, die das Softwarepaket enthält, so gebildet wie in Abschnitt „Private Programmpakte zur Installation vorbereiten“ auf Seite 110 beschrieben.

Beispiel

```
#
# Batch Installationsdatei

[PackageInstallation]   # Programmpakete installieren
# <product[:package]>;<imon>;<version>;<corr>;<uid>;<ipath>

#Installation der erweiterten Shell mit IMON
POSIX-SH;Y

#Installation von NFS ohne IMON
NFS;N;030;;TSOS

# Installation von C89 mit IMON
CPP;Y;;;;/opt/C

# Installation von OPENSSH mit IMON
TCP-IP-SV:OPENSSH;Y
```

Delete Packages from POSIX (Programmpakete entfernen)

Identifikationszeile: **[DeletePackage]**

Anweisungszeile: **<prod>;<vers>;<ipath>**

Dabei bedeuten:

<prod>	Name des Softwarepakets zusammengesetzt aus Produktname und optionalen Paketnamen, falls das Produkt in Pakete zerlegt ist. Syntax: <Produktname>[:<Paketname>]
<vers>	Produktversion des Softwarepakets
<ipath>	Optionaler Installationspfad, falls das Softwarepaket dies unterstützt. Standardwert ist '/'.

Beispiel

```
#
# Batch Installationsdatei
#
[DeletePackage]     # Programmpakete loeschen
# <product[:package]>;<version>;<ipath>

# Loeschen von NFS
NFS;030

# Loeschen von C89 auf bestimmtem Installationspfad
CPP;032;/opt/C
```

5.7 Protokollierung der Installation

POSIX protokolliert die Paket-Installation in der Logging-Datei */var/sadm/pkg/instlog*.

Pro Vorgang wird ein Eintrag geschrieben, der folgende Informationen enthält:

Zeile 1	Kennzeichen, ob Installation oder Löschung (in der 1. Spalte: I (install) oder D (delete)) Name des Produkts oder Pakets Version des Produktes Datum und Uhrzeit der Installation bzw. Löschung
Zeile 2	nur bei Installation: Installationsbibliothek
Zeile 3	Installationspfad

Beispiel

```
I JENV.050                Wed Sep 19 13:15:46 2007   ...
I BCAM.190                Mon Jun  9 11:35:10 2008   ...
I NFS.030                 Thu Nov 20 12:23:41 2008   ...
I POSIX-BC.070            Tue Jan 27 11:33:27 2009   ...
I POSIX-SH.070            Tue Jan 27 11:35:13 2009   ...
I POSIX-SOCKETS.070       Tue Jan 27 12:53:51 2009   ...
I POSIX-NSL.070           Tue Jan 27 12:58:30 2009   ...
```

Informationen über die installierten Pakete können mit dem Shell-Kommando *pkginfo* ausgegeben werden.

Treten bei der Installation im Batchbetrieb Probleme auf, werden diese in der Datei */var/sadm/pkg/insterr* protokolliert (siehe „Protokollierung von Fehlern in der Parameterdatei (MAINCODE = POS2956):“ auf Seite 262.

Information über installierte POSIX-Packages (pkginfo)

Das Kommando *pkginfo* zeigt Informationen über Software-Pakete an, die im POSIX installiert sind. Ein im POSIX installiertes Software-Paket wird beschrieben durch:

- Name des Software-Produkts
- Paket (Package) aus dem Software-Produkt (optional)
- Version des Software-Produkts
- Pfad, unter dem das Software-Paket installiert ist (Standard: /)
- BS2000-Bibliothek (SINLIB), aus der das Software-Paket installiert wurde
- Datum der (letzten) Installation

Dieses Kommando ist auch jedem nicht privilegierten POSIX-Benutzer zugänglich.

5.8 POSIX-Informationsdatei

In der POSIX-Informationsdatei SYSSSI.POSIX-BC.<version> wird mit Steuerparametern die Größe der Systemtabellen festgelegt. Dadurch werden die Ressourcen kontrolliert, die das System und die Benutzer beanspruchen können.

Jedem Steuerparameter ist ein Standard-, ein Minimal- und ein Maximalwert zugeordnet. Die Standardwerte sind so gewählt, dass das POSIX-Subsystem in beliebiger Umgebung ablaufen kann, ohne das Gesamtsystem durch übermäßigen Ressourcenverbrauch zu belasten. Sie können diese Steuerparameter jedoch bei Bedarf an die Gegebenheiten Ihres Systems anpassen.

Die POSIX-Informationsdatei wird nur beim Hochfahren des POSIX-Subsystems ausgewertet, Änderungen werden daher erst nach dem nächsten Hochfahren wirksam. Einige Steuerparameter können jedoch auch mit dem Kommando *usp* im laufenden Betrieb geändert werden, siehe POSIX-Handbuch „Kommandos“ [1].

5.8.1 Inhalt der POSIX-Informationsdatei

Die nachfolgende Tabelle listet alle Parameter in alphabetischer Reihefolge auf. **Halbfett** dargestellte Parameter können mit dem Kommando *usp* dynamisch modifiziert werden. Die Spalte *Kategorie* gibt an, zu welchem Bereich der Parameter gehört:

Allgemein	Allgemeiner Systemparameter
Dateisystem	Dateisystemparameter
IPC	Steuerparameter für die Interprozesskommunikation
POSIX	Spezieller Parameter für POSIX

Bei der Auslieferung der Informationsdatei ist für den Steuerparameter ROOTFSNAME kein Wert eingetragen. Bei der POSIX-Erstinstallation wird der Name des root-Dateisystems vom POSIX-Installationsprogramm automatisch eingetragen.
Die Systemkennung SYSROOT ist als Eigentümer des root-Dateisystems obligatorisch und muss deshalb nicht angegeben werden.
Numerische Parameterwerte dürfen in den Einheiten Kilo (K, entspricht 1024) und Mega (M, entspricht 1048576) angegeben werden.

Parameter	Beschreibung	Standard	Min.	Max.	Kategorie
BINDANY	BCAM / mode flag	0	0	1	POSIX
BUFHWM	high-water-mark of buffer cache	2000	200	2000	Dateisystem
DBLPOOL	memory pool in class 6 memory	0	0	1024	POSIX
DBLSTATE	state of the loader	0	0	1	POSIX
FDFLUSHR	fsflush time interval	5	1	5	Dateisystem
FILESIZE	max. size of file	UNLIMITED	64	UNLIMITED	Allgemein
FLCKREC	max. # of active file records locks	1000	100	2000	Allgemein
FORCEDTERM	controlling termination	0	0	1	POSIX
HDPTNI	# of partition table entries	200	16	300	Allgemein
HDSTNI	# of hard disk server tasks	4	1	16	Allgemein
HEAPSZ	size of heap-segment	4M	2M	64M	Allgemein
KMAHWM	kma daemonkmem high water mark	2M	1M	2M	Allgemein
MAXTIMERC	max. wait time for rc term procs	660	120	1200	POSIX
MAXUP	max. # of processes per user	50	15	500	Allgemein
MINPAGEFREE	pageout daemon / min. # of free pages	0	0	0	Allgemein
MSGMAP	# of entries in msg map	200	10	200	IPC
MSGMAX	max. message size	2048	512	2048	IPC
MSGMNB	max. # bytes on queue	16384	4096	16384	IPC
MSGMNI	# of message queue identifiers	150	50	150	IPC
MSGSEG	# of msg segments (MUST BE < 32768)	2048	1024	32768	IPC
MSGSSZ	msg segment size	8	8	8	IPC
MSGTQL	# of system message headers	160	40	160	IPC
NAUTOUP	age of a delayed-write buffer	60	10	120	Dateisystem
NBUF	# of I/O buffers	200	100	2000	Dateisystem
NHBUF	buffer cache size for metadata	256	32	1024	Dateisystem
NOFILES	max. # of file descriptors	2048	2048	4096	Allgemein
NOPTY	max. # of ptys	64	4	1024	POSIX
NOSTTY	max. # of sttys	64	4	1024	POSIX
NOTTY	max. # of ttys	64	4	1024	POSIX
NPBUF	number of physical I/O buffers	20	20	40	Allgemein
NPROC	max. # of processes	200	50	2000	Allgemein
NRNODE	max. # of incore remote nodes (nfs)	600	400	600	Dateisystem
PGOVERFLOW	overflow buffers for pageout	32	32	32	Allgemein

Parameter	Beschreibung	Standard	Min.	Max.	Kategorie
PORTMON	port monitoring (nfs)	1	0	1	POSIX
ROOTFSNAME	name of root-file system	---	---	---	---
SEGMAPSZ	# of buffer cache entries	256	256	22500	Dateisystem
SEMAEM	adjust on exit max value	16384	16384	16384	IPC
SEMMAP	# of entries in semaphore map	150	10	150	IPC
SEMMNI	# of semaphore identifiers	150	10	150	IPC
SEMMNS	# of semaphores in system	200	60	200	IPC
SEMMNU	# of undo structures in system	200	30	200	IPC
SEMMSL	max. # of semaphores per id	25	25	25	IPC
SEMOPM	max. # of operations per semop call	20	10	20	IPC
SEMUME	max. # of undo entries per process	20	10	20	IPC
SEMVMX	semaphore maximum value	32767	32767	32767	IPC
SHMMAX	max. size of a shared memory segment	16M	131072	16M	IPC
SHMMIN	min. size of a shared memory segment	1	1	1	IPC
SHMMNI	# of shared memory headers	100	100	100	IPC
SHMSEG	max. # of segments per process	16	6	16	IPC
UFSNINODE	# of inodes	1000	600	1000	Dateisystem

5.8.2 Beschreibung der Steuerparameter

In vielen Fällen sind die Standardwerte der Steuerparameter ausreichend. Manchmal kann es aber sinnvoll sein, dass der BS2000-Systemverwalter Steuerparameter an die spezielle POSIX-Anwendung und an den Ressourcenvorrat des Gesamtsystems anpasst. Im folgenden Teil sind die Steuerparameter aufgeführt, für die eine Änderung sinnvoll sein kann. Zusätzlich ist bei jedem Steuerparameter die Bedeutung angegeben.

Allgemeine Systemparameter

FILESIZE Maximale Größe einer Datei beim Anlegen und Schreiben. Der voreingestellte Maximalwert liegt bei 1024 Gbyte. Aus Kompatibilitätsgründen kann statt UNLIMITED auch UNLIMITED64 angegeben werden.

FLCKREC Anzahl der vom System verwendeten Sperrstrukturen für Datensätze (recordlocks).

HDPTNI Maximale Anzahl gemounteter lokaler Dateisysteme.

HDSTNI Anzahl der Servertasks zur Durchführung asynchroner I/Os.

HEAPSZ Maximal möglicher Wert bei *brk()*-Systemcall.

KMAHWM Überschreitet die dynamische CI-4-Speicherbelegung in POSIX den angegebenen Wert, so wird der Kernel-Memory-Dämon zur Reorganisation und Freigabe des Speichers aktiviert.

MAXUP Maximale Anzahl der Prozesse, die ein nichtprivilegierter Benutzer gleichzeitig starten kann (nicht pro Terminal, sondern insgesamt).

MINPAGEFREE
Bedeutungslos, da nicht einstellbar. Implizit ist MINPAGEFREE auf 128 K gesetzt, d.h. wenn weniger als 128 K im Buffercache frei sind, wird pageout aktiviert.

NOFILES Maximale Anzahl offener Dateien im System.

NPBUF Maximale Anzahl der I/O-Puffer für physikalische I/Os. Dieser Wert sollte mindestens 4 * HDSTNI sein.

NPROC Maximale Anzahl der Benutzerprozesse, die im System erlaubt sind.

PGOVERFLOW
Anzahl fest reservierter I/O-Puffer für pageout auch bei Speicherengpass.

Dateisystemparameter

BUFHWM Größe des Speichers (in Kilobytes), der durch die Ein-/Ausgabepuffer belegt werden kann.

FDFLUSHR Zeitintervall (in Sekunden) zwischen zwei Aktivierungen eines Prozesses. *fsflush* schreibt Daten aus dem Cache-Puffer auf die Festplatte und stellt damit die Konsistenz der Daten auf der Festplatte sicher. Ein kleiner Wert für FDFLUSHR bringt größere Sicherheit gegen Datenverlust bei einem Systemausfall, geht aber zu Lasten der Systemleistung.

NAUTOUP Angabe (in Sekunden), wie lange ein Puffer im Speicher „altern“ muss, bevor er durch *fsflush* zurückgeschrieben wird. Dieser Wert betrifft nur den Inhalt des Cache-Puffers.

NBUF Anzahl von Ein-/Ausgabepuffern des Cache-Puffers, die vom Systemkern zugewiesen werden, wenn keine mehr frei sind.

NHBUF Anzahl von Hash-Ankern für den schnellen Zugriff auf Puffer des Cache-Puffers über Geräte- und Blocknummern.

NRNODE Maximale Anzahl von NFS-rnode-Strukturen. Dies sind spezielle Deskriptoren für offene Dateien aus NFS-Dateisystemen, d.h. diese Dateien liegen auf fernen Rechnern.

SEGMAPSZ Maximale Größe des Cache-Puffers im Klasse-4-Speicher (in Einheiten von 8KB). Dieser Parameter hat nur u.U. eine Auswirkung auf den Ein-/Ausgabedurchsatz. In der Regel, d.h. auf Hardware mit Dataspace-Unterstützung, wird der Cache-Puffer des POSIX-Kernel in Dataspaces und nicht im Klasse-4-Speicher gehalten und dieser Parameter ist wirkungslos.

UFSNINODE Maximale Anzahl von UFS-Indexeinträgen im Systemkern.

Steuerparameter für die Interprozess-Kommunikation

Nachrichten-Warteschlangen und Semaphore werden über sogenannte Resourcemaps verwaltet. Resourcemaps führen darüber Buch, wieviel Speicherplatz von Meldungen und Semaphoren verbraucht wurde. Die Anzahl der belegten Einträge einer Resourcemap zu einem bestimmten Zeitpunkt ist ein Maß für die aktuelle Stückelung des für Meldungen verfügbaren Speicherbereichs oder der verfügbaren Semaphore.
Wenn Steuerparameter wie MSGSEG oder SEMMNS erhöht werden, sollte die Größe der entsprechenden Resourcemap ebenfalls erhöht werden.

MSGMAP Anzahl der Einträge in der Resourcemap für Nachrichten-Warteschlangen.

MSGMAX Maximale Größe einer Meldung (in Bytes).

MSGMNB Maximale Gesamtgröße aller Meldungen einer Nachrichten-Warteschlange (in Bytes).

MSGMNI	Maximale Anzahl von Nachrichten-Warteschlangen systemweit.
MSGSEG	Anzahl von Meldungs-Segmenten im System. Wenn der Wert von MSGSSZ mit dem Wert von MSGSEG multipliziert wird, erhält man den gesamten Speicherplatz, der für Meldungsdaten zur Verfügung steht.
MSGSSZ	Minimale Zuweisungsgröße für Meldungsspeicher (Segmentgröße in Bytes).
MSGTQL	Anzahl der Nachrichtenköpfe im System. Sie entspricht der Anzahl der ausstehenden Meldungen.
SEMAEM	Maximaler Undo-Wert für eine Semaphore.
SEMMAP	Anzahl der Einträge in der Resourcemap für Semaphor-Sätze.
SEMMNI	Maximale Anzahl von Semaphor-Sätzen.
SEMMSL	Maximale Anzahl von Semaphoren pro Satz.
SEMMNS	Maximale Anzahl von Semaphoren im System.
SEMMNU	Maximale Anzahl von Prozessen mit noch ausstehenden Undo-Operationen. Prozesse können festlegen, ob ihre Semaphor-Aktionen bei Prozessende automatisch rückgängig gemacht werden.
SEMOPM	Maximale Anzahl von Semaphor-Operationen, die pro *semop(2)*-Systemaufruf ausgeführt werden können.
SEMUME	Maximale Anzahl von Undo-Operationen pro Prozess.
SEMVMX	Maximaler Wert für eine Semaphore.
SHMMAX	Maximale Größe eines gemeinsam nutzbaren Speicherbereichs (in Bytes).
SHMMIN	Minimale Größe eines gemeinsam nutzbaren Speicherbereichs (in Bytes).
SHMMNI	Maximale Anzahl von gemeinsam nutzbaren Speicherbereichen.
SHMSEG	Maximale Anzahl von gemeinsam nutzbaren Speicherbereichen, die ein Prozess gleichzeitig verwendet.

Spezielle Parameter für POSIX

BINDANY Der Parameter BINDANY ist seit BCAM Version 13 bedeutungslos.

DBLPOOL Um die Ladevorgänge für die POSIX-Shellkommandos und andere POSIX-Programme mit Hilfe des POSIX-Laders zu beschleunigen, kann hier ein Wert größer als Null eingegeben werden (in Megabyte). Siehe auch Abschnitt „POSIX-Lader“ auf Seite 158.

DBLSTATE Gibt an, ob der POSIX-Lader beim Hochfahren des POSIX-Subsystems automatisch aktiviert ist:

0=nein (Default); 1=ja.

FORCEDTERM
Forciertes Beenden des POSIX-Subsystems. Mit diesem Parameter kann gesteuert werden, ob bei vorhandenen Connections ein zweites STOP-SUBSYSTEM-Kommando mit dem Zusatz SUB-PARAMETER='FORCED-BY-SUBSYSTEM' abgesetzt werden muss, oder ob das Subsystem ohne zweites STOP-SUBSYSTEM-Kommando sofort beendet wird.

FORCEDTERM=0 (bisheriges Verhalten)
FORCEDTERM=1 (forciertes Beenden)

MAXTIMERC Maximale Wartezeit für den Ablauf der rc-Beendigungsprozeduren bei der POSIX-Terminierung.

NOPTY Maximale Anzahl physikalischer Terminals (Gerät dev/pts). Dies entspricht der Zahl der zulässigen *rlogin*- und *telnet*-Zugänge.

NOSTTY Maximale Anzahl von Systemdatei-Terminals (Gerät dev/sf), die POSIX unterstützt. Dies entspricht der Zahl der zulässigen POSIX-Zugänge über BS2000-Prozeduren und -Programme.

NOTTY Maximale Anzahl von Blockterminals (Gerät dev/term), die POSIX unterstützt. Dies entspricht der Zahl der POSIX-Zugänge über BS2000-Dialogtasks (Kommando START-POSIX-SHELL).

PORTMON Ein- oder Ausschalten der Portüberwachung für NFS (0=ausgeschaltet, 1=eingeschaltet).

6 POSIX-Subsystem und POSIX-Lader

Dieses Kapitel wendet sich an die Systemverwalter von BS2000 und POSIX. Es informiert Sie über

- die Steuerung des POSIX-Subsystems (Starten, Beenden, Überwachen)
- den POSIX-Lader (Übersicht, Initialisierung, Linkvorgang, Ladevorgang, Administration)

6.1 POSIX-Subsystem steuern

Dieses Unterkapitel beschreibt das Starten, Beenden und Überwachen des POSIX-Subsystems und gibt Hinweise auf BCAM-Abhängigkeiten beim Starten und Beenden.

6.1.1 POSIX-Subsystem starten

Folgende Voraussetzungen müssen erfüllt sein, damit ein Benutzer mit dem Privileg SUBSYSTEM-MANAGEMENT oder OPERATING POSIX starten kann:

- Das Subsystem POSIX muss installiert sein (siehe Kapitel „POSIX installieren“ auf Seite 103).
- Eventuell muss die POSIX-Informationsdatei angepasst werden (siehe Seite 145).

  Der Name der Behälterdatei, in der sich das root-Dateisystem befindet, muss mit dem Steuerparameter ROOTFSNAME der POSIX-Informationsdatei übereinstimmen.

  Bei einer Erstinstallation wird der Name des neu erzeugten root-Dateisystems in die POSIX-Informationsdatei eingetragen. Deshalb ist in diesem Fall keine Kontrolle auf Übereinstimmung nötig.
- Der schreibende Zugriff auf die Behälterdatei des root-Dateisystems und aller anderen Dateisysteme, die während des POSIX-Starts eingehängt werden sollen, muss möglich sein (Attribut ACCESS=*WRITE im Dateikatalog).
- POSIX muss im Subsystemkatalog eingetragen sein.
  Bei Installation mit IMON wird der Eintrag automatisch vorgenommen.

POSIX wird entweder automatisch nach erfolgter Erstinstallation gestartet oder explizit mit folgendem BS2000-Kommando

```
/START-SUBSYSTEM SUBSYSTEM-NAME=POSIX
```

Wenn das Starten des Subsystems POSIX erfolgreich abgeschlossen wurde, erscheint an der Konsole folgende Meldung:

```
POS4100: INIT: THE POSIX SUBSYSTEM IS READY.
```

Wenn das Starten nicht erfolgreich abgeschlossen wurde, weil z.B. der gestartete Init-Prozess nicht beendet werden konnte, können Sie die Ursache der Protokolldatei des Init-Prozesses $SYSROOT.SYSLOG.POSIX-BC.<version>.INIT entnehmen.

Subsystem-Parameter

Um beim POSIX-Startup die Konsistenzprüfung und ggf. die Bereinigung von Dateisystemen zu erzwingen, werden zwei Parameter beim Kommando START-SUBSYSTEM unterstützt. Diese Parameter können in Groß- oder Kleinschreibung eingegeben werden, sie dürfen aber nicht abgekürzt werden.

- `/START-SUBSYSTEM POSIX,SUBSYSTEM-PARAMETER='CHECK-SYSTEM-FS'`

  Die folgenden Dateisysteme werden vor dem Einhängen mit *fsck* überprüft und wenn nötig bereinigt:

  – / (root-Dateisystem)
  – /var
  – /opt (nur falls vorhanden)

- `/START-SUBSYSTEM POSIX,SUBSYSTEM-PARAMETER='CHECK-ALL-FS'`

  Alle Dateisysteme werden vor dem Einhängen mit *fsck* überprüft und wenn nötig bereinigt.

Unterstützung von rc-Prozeduren

POSIX unterstützt zwar nicht den Runlevel-Mechanismus des Native UNIX, jedoch können in Anlehnung an UNIX rc-Prozeduren definiert werden, die beim Starten und Beenden von POSIX automatisch ablaufen. Wie bei UNIX müssen die rc-Prozeduren, die beim Start aktiviert werden sollen, im Verzeichnis */etc/rc2.d* abgelegt werden und die Prozeduren, die beim Beenden aktiviert werden, im Verzeichnis */etc/rc0.d*. Die rc-Prozeduren werden nacheinander in alphabetischer Reihenfolge beim Starten bzw. Beenden von POSIX durch die Shell-Skripts */etc/rc2* bzw. */etc/rc0* aufgerufen. Falls die Datei */etc/.trace.rc* existiert, werden diese Aufrufe an der BS2000-Konsole protokolliert.

Mit POSIX-BC und POSIX-SH werden rc-Prozeduren zum Starten bzw. Beenden folgender Dämonen eingerichtet: *shmd* (Shared-Memory-Dämon), *syslogd* (Syslog-Dämon), *fsmond* (Dämon zur Überwachung der Dateisystembelegung), *rpcbind* (Dämon für RPC-Dienste), *inetd* (Internet-Superdämon für Netzwerkdienste), *cron* (Dämon für die Kommandos *cron* und *at*). Zusätzliche Softwareprodukte wie z.B. NFS installieren eigene rc-Prozueduren zum Starten bzw. Beenden weiterer Dämonen.

Die maximale Wartezeit für den Ablauf der rc-Beendigungsprozeduren während der POSIX-Terminierung ist über den Parameter MAXTIMERC in der Informationsdatei (siehe Seite 151) steuerbar. Ist die Ausführung der rc-Beendigungsprozeduren nach Ablauf dieser Zeitspanne noch nicht abgeschlossen, wird die POSIX-Terminierung abhänging vom Wert des Parameters FORCEDTERM entweder abgebrochen oder im abnormalen Modus fortgesetzt.

6.1.2 POSIX beenden

POSIX wird entweder explizit durch einen Benutzer mit dem Privileg SUBSYSTEM-MANAGEMENT oder OPERATING beendet oder automatisch beim Shutdown des BS2000-Betriebssystems. Wenn ein schwerwiegender Fehler auftritt, wird POSIX abnormal beendet.

Subsystem POSIX explizit beenden

Das Subsystem POSIX wird von der BS2000-Systembetreuung oder dem Operating mit folgendem Kommando beendet:

```
/STOP-SUBSYSTEM SUBSYSTEM-NAME=POSIX
```

Wenn zum Zeitpunkt der POSIX-Beendigung noch Benutzer mit dem Subsystem POSIX verbunden sind und in der POSIX-Informationsdatei FORCEDTERM=0 gesetzt ist, bricht das DSSM die Beendigung ab. Sie können dann die Beendigung wie folgt erzwingen:

```
/STOP-SUBSYSTEM SUBSYSTEM-NAME=POSIX, -
/               SUBSYSTEM-PARAMETER='FORCED-BY-SUBSYSTEM'
```

Der Parameter kann beliebig in Groß- und Kleinbuchstaben geschrieben, aber nicht abgekürzt werden.

Wenn in der POSIX-Informationsdatei FORCEDTERM=1 gesetzt ist, dann wird POSIX immer sofort beendet, das zweite STOP-SUBSYSTEM-Kommando ist dann nicht nötig.

Das Subsystem POSIX kann nicht mit dem Kommando /HOLD-SUBSYSTEM POSIX angehalten werden. Das Kommando wird zurückgewiesen; die laufende POSIX-Sitzung wird dadurch nicht beeinträchtigt.

Die Beendigung des Subsystems POSIX wird an der Konsole gemeldet:

```
POS3010: SUBSYSTEM POSIX HAS BEEN TERMINATED.
```

Subsystem POSIX beim Shutdown des BS2000-Betriebssystems beenden

DSSM beendet das Subsystem POSIX während des Shutdowns implizit, damit die POSIX-Dateien in einen konsistenten Zustand gebracht werden.

Abnormale Beendigung von POSIX

Bei einem schwerwiegenden Fehler wird POSIX abnormal beendet. Dabei arbeiten die BS2000-Subsystem-Verwaltung und POSIX eng zusammen. Alle Programme, die POSIX verwenden, werden abnormal beendet. Die von POSIX belegten BS2000-Ressourcen werden freigegeben.

Wenn sich der Init-Prozess während einer POSIX-Sitzung beendet, wird eine abnormale POSIX-Beendigung eingeleitet, da der Init-Prozess eine zentrale Steuerungsfunktion in POSIX hat und deshalb für einen fehlerfreien Ablauf unentbehrlich ist.

6.1.3 Überwachen des POSIX-Subsystems über eine Monitor-Jobvariable

Das POSIX-Subsystem kann mit der Monitor-Jobvariable $.SYS.POSIXSTATUS überwacht werden. Dazu starten Sie das POSIX-Subsystems wie folgt:

```
/START-SUBSYSTEM POSIX,MONJV=$.SYS.POSIXSTATUS
```

In dieser Monitor-Jobvariable kann zu den von DSSM gesetzten Zuständen auch der Status des POSIX-Subsystems abgefragt werden. Es sind folgende Zustände des POSIX-Subsystems möglich:

Zeitpunkt	**Spalte 89 in MONJV:**
vor /START-SUBSYSTEM	NOT CREATED
vor Subsystem ready	IN CREATE
vor POSIX ready	CREATED
nach POSIX ready	*AVAILABLE
nach /STOP-SUBSYSTEM	IN DELETE
nach /STOP rejected	CREATED
Subsystem entladen	NOT CREATED

Wenn eine BS2000-Sitzung nicht ordnungsgemäß mit SHUTDOWN beendet wird, bleibt die Monitor-Jobvariable $SYS.POSIXSTATUS gesperrt. Bevor sie wieder zur Überwachung des POSIX-Subsystems verwendet werden kann, muss sie mit folgendem Kommando entsperrt werden:

```
/MODIFY-JV-ATTRIBUTES JV-NAME=$.SYS.POSIXSTATUS,
          PROTECTION=*PARAMETERS(MONJV-PROTECTION=*NO)
```

6.1.4 BCAM-Abhängigkeiten beim Starten und Beenden von POSIX

Das Subsystem POSIX kann erst nach ’BCAM READY’ gestartet werden.

Nach einem Neustart von BCAM muss auch das Subsystem POSIX beendet und neu gestartet werden. Auf diese Notwendigkeit wird bis zur Beendigung des Subsystems POSIX mit der Konsolmeldung POS1040 hingewiesen, die mit /SHOW-PENDING-MSG (/STATUS MSG) angezeigt werden kann.

6.2 POSIX-Lader

Dieses Unterkapitel informiert Sie über die Zweckbestimmung, die Bestandteile und den Funktionsumfang des POSIX-Laders. Eine kurze Einführung finden Sie im Kapitel „Arbeiten mit POSIX“ auf Seite 68. Die zur Verwaltung des POSIX-Laders bestimmten Kommandos *posdbl* und *pdbl* werden beispielhaft erklärt. Ihre ausführliche Beschreibung findet sich im POSIX-Handbuch „Kommandos“ [1].

6.2.1 Übersicht

Die Ladezeit eines POSIX-Programms hängt erheblich von der Größe des Programms ab und macht in der Regel mindestens ca. 35% der Gesamtausführungszeit aus. Zurückzuführen ist dies größtenteils auf Bibliothekszugriffe in DMS/PLAM, die auf Anforderung von BLS durchgeführt werden. Durch den Einsatz von DAB lassen sich die Programmladezeiten erheblich verkürzen. Dies allerdings nur, wenn alle Bibliotheken, die an einem Ladevorgang beteiligt sind, auch tatsächlich erfasst werden.

Als Alternative wurde der POSIX-Lader entwickelt. Dieser betrachtet nicht Bibliotheken, sondern ablaufbereite Core-Images im Speicher nach einem kompleten Ladevorgang. Die Core-Images werden in Programm-Caches gesichert und für jeden weiteren Ablauf in den Speicher kopiert. Dies verkürzt die Ladezeiten im Vergleich zu BLS ohne DAB um bis zu 80% und im Vergleich zu BLS mit DAB um bis zu 65%.

In der folgenden exemplarischen Abbildung sind die normierten Ladezeiten des Beispielprogramms *snet* in POSIX relativ zueinander dargestellt.

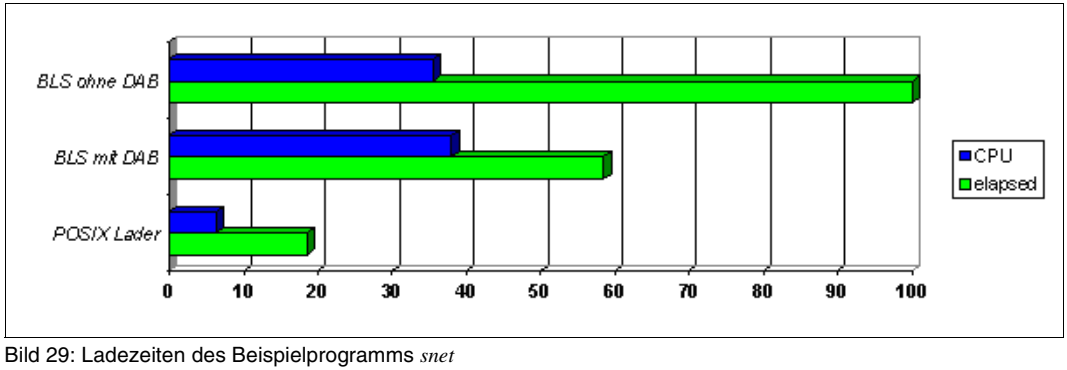

Bild 29: Ladezeiten des Beispielprogramms *snet*

Der POSIX-Lader bietet zusätzlich noch den Vorteil, dass einzelne Programme aus dem UFS-Dateisystem je nach individueller Anforderung zwischengespeichert werden können, und zwar systemglobal, benutzerspezifisch oder sitzungsspezifisch.

Der POSIX-Lader ist Bestandteil des POSIX-Subsystems und gliedert sich in zwei Teile:

- Privilegierter Teil

  Dieser Teil wird vom Superuser verwaltet. Gehalten wird ein globaler Programm-Cache in skalierbarer Größe zur Speicherung ablauffähiger Core-Images von POSIX-Programmen. Diese Core-Images werden entweder automatisch beim ersten Aufruf eines POSIX-Programms aus einer der festgelegten Bibliotheken in den Programm-Cache gespeichert ("impliziter Linkvorgang") oder explizit vom Superuser mit Hilfe des Kommandos *posdbl –b*. Zum Laden eines gespeicherten Programms steht der globale Programm-Cache allen Benutzern zur Verfügung.

- Nicht privilegierter Teil

  Dieser Teil kann von jedem Benutzer verwendet werden. Es können benutzerspezifische Programm-Caches gehalten werden, die vom jeweiligen Nutzer selbst zu verwalten sind. Der Scope eines benutzerspezifischen Programm-Caches ist wahlweise

  USERWIDE alle Prozesse einer User-ID sind angeschlossen oder

  SESSIONWIDE alle Prozesse einer Sitzung sind angeschlossen.

  Core-Images werden in benutzerspezifischen Programm-Caches mit Hilfe des Kommandos *pdbl* gespeichert. Zum Laden eines gespeicherten Programms werden die benutzerspezifischen Programm-Caches vorrangig herangezogen. Wird in keinem der Caches ein dem Programm entsprechendes Core-Image gefunden, so wird das Programm wie üblich über BLS geladen.

Folgende Vorgänge sind bei der Nutzung des POSIX- Laders zu unterscheiden:

- Initialisierung (Programm-Cache einrichten)

  Der globale Programm-Cache kann auf zwei Arten eingerichtet werden:

  – Wenn der Wert des DBLPOOL-Parameters in der POSIX-Informationsdatei größer als 0 ist, wird der globale Programm-Cache automatisch beim Start des Subsystems in der entsprechenden Größe (in MB) eingerichtet.

  – Ist der DBLPOOL-Wert gleich 0, kann der globale Programm-Cache explizit mit dem Kommando *posdbl* eingerichtet werden, und zwar in der zuvor mit dem Kommando *usp* vereinbarten Größe.

  Die benutzerspezifischen Programm-Caches richtet der jeweillige Benutzer mit Hilfe des Kommandos *pdbl* ein.

- Linkvorgang (Core-Image erstellen)

  Mit Hilfe der Kommandos *posdbl* bzw. *pdbl* wird das Programm geladen und das Core-Image in den Programm-Cache kopiert. Für den globalen Programm-Cache wird u. U. auch ein impliziter Linkvorgang beim Systemcall *exec()* durch den POSIX-Kernel ausgelöst.

- Ladevorgang (Core-Image laden und starten)

  In den Programm-Caches wird nach einem Core-Image gesucht und das Core-Image beim Systemcall *exec()* durch den POSIX-Kernel zum Ablauf in den Speicher kopiert.

- Administration (Programm-Caches verwalten)

  Mit Hilfe der Kommandos *posdbl* bzw. *pdbl* werden Programm-Caches aktiviert bzw. deaktiviert, der Status abgefragt, Core-Images aufgelistet und gelöscht sowie Programm-Caches insgesamt gelöscht.

  

  Die korrekte Funktion eines aus dem Programm-Cache geladenen Core-Images kann nicht gewährleistet werden, wenn dieses während des Ablaufs versucht, Programmteile dynamisch nachzuladen.

6.2.2 Initialisierung

Globalen Programm-Cache einrichten und aktivieren

Der globale Programm-Cache kann auf zwei Arten eingerichtet und aktiviert werden:

– Automatisch beim Hochfahren des POSIX-Subsystems, gesteuert über bestimmte Parameter in der POSIX-Informationsdatei

– Explizit mit dem Kommando *posdbl* während der laufenden POSIX-Session

In der POSIX-Informationsdatei sind zwei Parameter für den privilegierten Teil des POSIX-Laders definiert:

```
DBLSTATE      | initial state of POSIX loader         | status
DBLPOOL       | size of pool (MB) for POSIX loader    | größe
```

Ist die Größe *größe* des Programm-Cache gleich Null Megabyte, so wird kein globaler Programm-Cache eingerichtet. Der Anfangsstatus *status* des globalen Programm-Cache ist in diesem Fall irrelevant.

Ist die Größe *größe* des Programm-Cache größer als Null Megabyte, so wird ein globaler Programm-Cache in der angegeben Größe eingerichtet. Ist der Anfangstatus *status* auf „1“ gesetzt ist, so ist der globale Programm-Cache aktiviert. Ist der Anfangstatus *status* auf „0“ gesetzt, so ist der globale Programm-Cache deaktiviert.

Es wird ein Memory-Pool mit Scope *Global* in der angegebenen Größe eingerichtet. Der obere Grenzwert wird nicht von *posdbl*, sondern von den systemspezifischen Einstellungen bestimmt.

Falls die mit dem DBLPOOL-Parameter beim Start des POSIX-Subsystems festgelegte Größe 0 ist, kann der globale Programm-Cache auch nachträglich mit den Kommandos *posdbl* und *usp* folgendermaßen neu eingerichtet werden:
– Mit *usp* Größe in MB festlegen (usp –p dblpool –v *wert*)
– Mit *posdbl* Programm-Cache neu einrichten (posdbl –n)

Der globale Programm-Cache ist dann noch nicht aktiviert.

Der implizite Link- und Ladevorgang muss mit der Option *-e* des *posdbl*-Kommandos aktiviert werden.

Siehe hierzu Abschnitt „Administration“ auf Seite 164.

Benutzerspezifische Programm-Caches einrichten und aktivieren

USERWIDE

In jedem beliebigen Prozess einer User-ID *ruid* (reale POSIX User Identifikation) wird mit dem Kommandoaufruf

`pdbl –u –e` *größe*

ein Programm-Cache für alle existierenden und nachfolgenden Prozesse der User-ID eingerichtet und automatisch aktiviert. Ein Hintergrund-Prozess mit dem Programmnamen `dblu`*ruid* wird erzeugt, um den Programm-Cache zu halten.

größe ist die Größe des Programm-Caches in Megabyte. Es wird ein Memory-Pool mit Scope *Group* in der angegebenen Größe eingerichtet. Der obere Grenzwert wird nicht von *pdbl*, sondern von den system- und taskspezifischen Einstellungen bestimmt.

SESSIONWIDE

In jedem beliebigem Prozess wird mit dem Kommandoaufruf

`pdbl –s [`*sid*`] –e` *größe*

ein Programm-Cache für alle existierenden und nachfolgende Prozesse in der angegebenen Sitzung *sid* eingerichtet und automatisch aktiviert. Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Es wird ein Memory-Pool mit Scope *Group* in der angegebenen Größe eingerichtet. Der obere Grenzwert wird nicht von *pdbl*, sondern von den system- und taskspezifischen Einstellungen bestimmt. Die Größe des Pools kann das ADDRESS-SPACE-LIMIT der Benutzerkennung nicht überschreiten.

Ein Hintergrund-Prozess mit dem Programmnamen `dbls`*sid* wird erzeugt, um den Programm-Cache zu halten. Wird eine andere *sid* als die des aktuellen Prozesses angegeben, so muss die Sitzung existieren und in derselben User-ID aktiv sein wie die des aktuellen Prozesses, d.h. ein Benutzer kann sich nur auf seine eigenen Sitzungen beziehen.

Beispiel

```
$ pdbl -u -e 20     # Programm-Cache für alle Prozesse der User-ID einrichten

$ ps -ef
     UID   PID  PPID  C    STIME TTY      TIME CMD
    GAST   206     1  0 16:04:01 ?        0:00 dblu101
    GAST   204   203  0 15:59:59 pts/0    0:03 [sh]
    GAST   207   204  0 16:04:04 pts/0    0:05 [ps]

$ pdbl -s -e 20     # Programm-Cache für die aktuelle Sitzung einrichten

$ ps -ef
     UID   PID  PPID  C    STIME TTY      TIME CMD
    GAST   210     1  0 16:11:59 pts/0    0:00 dbls204
    GAST   206     1  0 16:04:01 ?        0:00 dblu101
    GAST   211   204  0 16:12:01 pts/0    0:01 [ps]
    GAST   204   203  0 15:59:59 pts/0    0:03 [sh]
```

6.2.3 Linkvorgang

Impliziter Linkvorgang in den globalen Programm-Cache

Beim ersten Aufruf eines POSIX-Programms, das in einer Bibliothek enthalten ist, für die der implizite Linkvorgang aktiviert ist, wird der Linkvorgang gestartet. Das Programm wird über die BLS-Schnittstelle *BIND* geladen. Vor dem Start des Programms wird das Core-Image des Programms im CI.6-Speicher analysiert. Alle benötigen Informationen über das geladene Programm (pro Slice: Adresse, Länge und Attribute) liefert BLS. Es werden alle Slices des geladenen Programms in den globalen Programm-Cache kopiert und anschließend wird das Programm gestartet.

Es gibt eine Reihe von Kommandos, die nur relativ selten benutzt werden und deshalb nicht automatisch in den globalen Programm-Cache geladen werden. Dabei handelt es sich z.B. um Dämonen oder Kommandos aus dem *mount*/*umount*-Komplex.

Expliziter Linkvorgang in den globalen Programm-Cache

Im Unterschied zum impliziten Linkvorgang kann der Superuser jedes beliebige POSIX-Programm in den globalen Programm-Cache aufnehmen. Mit dem Kommandoaufruf

`posdbl -b` *pfad*

wird der Linkvorgang initiiert und das ablaufbereite Core-Image des Programms mit dem Pfadnamen *pfad* in den globalen Programm-Cache kopiert.

Expliziter Linkvorgang in den benutzerspezifischen Programm-Cache

USERWIDE

Mit dem Kommandoaufruf

`pdbl -u -b` *pfad*

wird der Linkvorgang initiiert und das ablaufbereite Core-Image des Programms mit dem Pfadnamen *pfad* in den benutzerspezifischen Programm-Cache kopiert.

SESSIONWIDE

Mit dem Kommandoaufruf

`pdbl -s [`*sid*`] -b` *pfad*

wird der Linkvorgang initiiert und das ablaufbereite Core-Image des Programms mit dem Pfadnamen *pfad* in den benutzerspezifischen Programm-Cache der Sitzung *sid* kopiert. Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Der Benutzer muss für das mit *pfad* angebene Programm Ausführberechtigung haben.

6.2.4 Ladevorgang

Bei jedem Aufruf eines POSIX-Programms über den Systemcall *exec()* werden vorrangig folgende Bedingungen in der angebenen Reihenfolge geprüft:

- Ist die Task eine durch *fork* erzeugte Task?
- Wird das Programm nicht im Debug-Modus gestartet?
- Existiert ein Programm-Cache für die Sitzung oder für den Benutzer bzw. existiert ein globaler Programm-Cache und ist das entsprechende Core-Image des Programms dort gespeichert?

Sind nicht alle Bedingungen, so wird das Programm über BLS geladen und gestartet.

Sind alle Bedingungen erfüllt, wird das im ausgewählten Programm-Cache gespeicherte Core-Image des Programms direkt und unter Umgehung von BLS in den Speicher kopiert und gestartet.

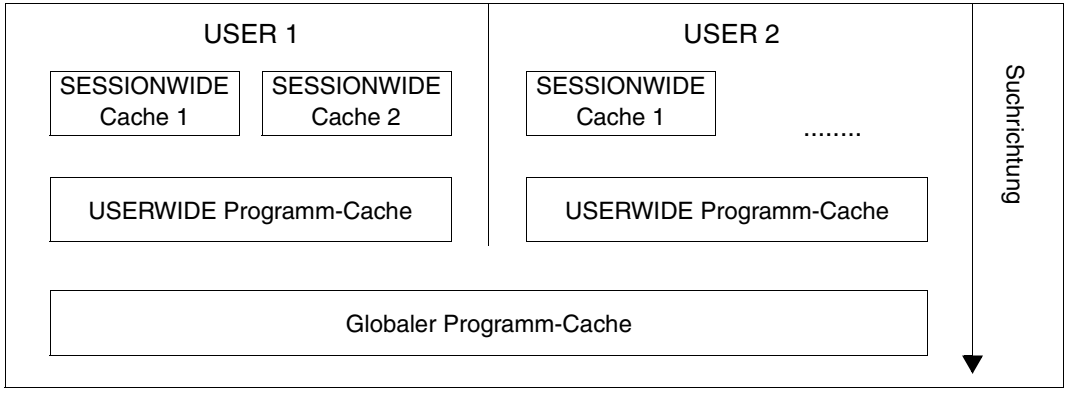

Bild 30: Programm-Caches, die einem Prozess zur Verfügung stehen

Die Umgehung von BLS und damit die unvollständige Einbettung der von POSIX gestarteten Programme in das BS2000-Programm-Environment bringt für Programme, die aus dem Programm-Cache geladen wurden, folgende Einschränkungen mit sich:

- Im Programm definierte Entries sind nach außen nicht sichtbar, d.h. das Program kann zwar Objekte dynamisch nachladen, das nachgeladene Objekt kann aber keine externen Verweise auf das Programm befriedigen.
- Es wird keine Lademeldung ausgegeben.
- Testen mit AID ist nicht möglich.
- Der Programm-Name in der Ausgabe des Kommandos */STATUS* fehlt.
- Beim Laden aus dem Programm-Cache verliert das Kommando *debug* seine Funktion, d.h. man sollte dieses Kommando nicht in den Programm-Cache laden.

6.2.5 Administration

Globalen Programm-Cache aktivieren/deaktivieren

Der Anfangsstatus des globalen Programm-Caches und des impliziten Linkvorgangs wird in der POSIX-Informationsdatei definiert:

```
DBLSTATE      | initial state of POSIX loader          | status
```

Ist der Anfangsstatus *status* auf „1“ gesetzt, sind sowohl der implizite Linkvorgang als auch der Ladevorgang aktiviert. Ist der Anfangsstatus *status* auf „0“ gesetzt, sind beide Vorgänge deaktiviert.

Spätere Statusänderungen können zusammenfassend oder auch getrennt für den impliziten Linkvorgang und den Ladevorgang mit dem Kommando *posdbl* vorgenommen werden.

- Impliziten Link- und Ladevorgang aktivieren/deaktivieren:

  ```
  posdbl {-e|-d} both
  ```

- Impliziten Linkvorgang aktivieren/deaktivieren:

  ```
  posdbl {-e|-d} linker
  ```

- Ladevorgang aktivieren/deaktivieren:

  ```
  posdbl {-e|-d} loader
  ```

Explizite Linkvorgänge werden unabhängig vom Status ausgeführt.

Benutzerspezifische Programm-Caches aktivieren/deaktivieren

Nach dem Einrichten ist ein benutzerspezifischer Programm-Cache aktiviert, d.h. bei Ladevorgängen wird der Programm-Cache nach geeigneten Core-Images durchsucht. Mit dem Kommando *pdbl* kann der benutzerspezifische Programm-Cache deaktiviert werden.

- USERWIDE (benutzerspezifischen Programm-Cache der User-ID deaktivieren):

  ```
  pdbl -u -d
  ```

- SESSIONWIDE (benutzerspezifischen Programm-Cache einer Sitzung deaktivieren):

  ```
  pdbl -s[sid] -d
  ```

  Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Wird ein leerer Programm-Cache deaktiviert, so wird dieser aufgelöst und kann nur durch erneutes Einrichten aktiviert werden. Wird ein belegter Programm-Cache deaktiviert, so wird dieser bis zur erneuten Aktivierung bei Ladevorgängen nicht berücksichtigt. Linkvorgänge werden unabhängig vom Status ausgeführt.

Der benutzerspezifische Programm-Cache wird mit dem Kommando *pdbl* wie folgt aktiviert:

- USERWIDE (benutzerspezifischen Programm-Cache der User-ID aktivieren):

  ```
  pdbl -u -a
  ```

- SESSIONWIDE (benutzerspezifischen Programm-Cache einer Sitzung aktivieren):

  ```
  pdbl -s[sid] -a
  ```

  Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Nach der Aktivierung wird der Programm-Cache wieder bei Ladevorgängen berücksichtigt.

Status des globalen Programm-Cache abfragen

Der Kommandoaufruf

```
posdbl -s
```

gibt den Status des globalen Programm-Caches und des impliziten Linkvorgangs sowie u. a. statistische Daten über Größe und Belegung wie im folgenden Beispiel aus:

```
$ posdbl -s              # Status ausgeben
POSIX-DBL:               linker ON       loader ON
Cache POSIX@DBL          CREATED: 01/22/09 14:16:23
                         SIZE: 24 MB     ENTRIES: 9
                         FREE PAGES: 2688
```

Status benutzerspezifischer Programm-Caches abfragen

Informationen über die benutzerspezifischen Programm-Caches werden mit dem Kommando *pdbl* ausgegeben. Im folgenden Beispiel wird erst ein Programm-Cache mit Scope *USERWIDE* in der Größe 10 MB angelegt und dann werden die aktuellen Informationen angefordert. Anschließend wird der Programm-Cache deaktiviert und die aktuellen Informationen werden erneut angefordert:

```
$ pdbl -u -e 10

$ pdbl -u -i
Cache DBLU2001           CREATED: 01/22/09 10:12:51       STATE: active
                         SIZE: 10 MB     ENTRIES: 0
                         FREE PAGES: 2559

$ pdbl -u -d

$ pdbl -u -i
pdbl: cache DBLU2001 not found
```

Für einen Programm-Cache mit Scope *SESSIONWIDE* werden die aktuellen Informationen mit dem Kommandoaufruf

```
pdbl -s[sid] -i
```

angefordert.

Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Core-Images im globalen Programm-Cache auflisten

Eine Liste aller im globalen Programm-Cache gespeicherten Core-Images wird mit dem Kommando *posdbl* angefordert. Das folgende Beispiel zeigt zwei Core-Images, die durch den impliziten Linkvorgang aus der Shell-Bibliothek im globalen Programm-Cache gespeichert wurden. Das dritte Core-Image wurde explizit mit dem Kommandoaufruf

```
posdbl -b /opt/C/bin/snet
```

im globalen Programm-Cache gespeichert.

```
$ posdbl -l
 PS                 53 Jan 23 13:45:08 $TSOS.SINLIB.POSIX-BC.090.SHELL
 SH                243 Jan 28 13:15:23 $TSOS.SINLIB.POSIX-BC.090.SHELL
+IN@RLOGIND        113 Jan 28 13:15:17 $TSOS.SINLIB.POSIX-BC.090.ROOT
```

Detailinformationen über ein Core-Image können mit dem Kommandoaufruf

```
posdbl -l element
```

angefordert werden. *element* ist der Name des Core-Image, wie in der Auflistung ausgegeben.

```
$ posdbl -l IN@RLOGIND
IN@RLOGIND      CREATED : 01/22/09 11:56:00     ACCESS: 01/28/09 13:15:17
                START AT: 0x01003CA0            CACHESIZE: 452 kB
                                                USECOUNT: 12
                ------------------------------------------------------------
                SLICES  : 1     LOADADDR:       SIZE:
                                0x01000000      452 kB
                ------------------------------------------------------------
                Loaded by command from:
                $TSOS.SINLIB.POSIX-BC.090.ROOT
$
```

Core-Images in benutzerspezifischen Programm-Caches auflisten

Eine Liste aller im benutzerspezifischen Programm-Cache gespeicherten Core-Images wird mit dem Kommando *pdbl* angefordert:

USERWIDE `pdbl -u -l`

SESSIONWIDE `pdbl -s[`*sid*`] -l`

Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Im folgenden Beispiel wird für Sitzung *541* ein Programm-Cache in der Größe 10 MB eingerichtet, das Core-Image des Programms *snet* gespeichert, die Statusinformationen und anschließend die Liste der gespeicherten Core-Images angefordert:

```
$ pdbl -s 541 -e 10

$ pdbl -s 541 -b /opt/C/bin/snet

$ pdbl -s 541 -i
Cache     DBLS90        CREATED: 01/23/09 13:12:32      STATE: active
                        SIZE: 10 MB     ENTRIES: 1
                        FREE PAGES: 153

$ pdbl -s 541 -l
SNET          2406 Jul 26 13:13:26 $TSOS.SINLIB.SNET.010
```

Detailinformationen über ein Core-Image im benutzerspezifischen Programm-Cache werden mit dem Kommando *pdbl* angefordert:

USERWIDE `pdbl -u -l` *element*

SESSIONWIDE `pdbl -s[`*sid*`] -l` *element*

Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

element ist der Name des Core-Images, wie in der Auflistung ausgegeben.

```
$ pdbl -s 541 -l SNET
SNET            CREATED : 01/23/09 16:43:41     ACCESS: 01/23/09 17:29:12
                START AT: 0x01000048            CACHESIZE: 9624 kB
                                                USECOUNT: 2
-------------------------------------------------------------------------
                SLICES  : 2     LOADADDR:       SIZE:
                                0x01000000      4272 kB
                                0x01500000      5352 kB
---------------------------------------------------------
Loaded from:
$TSOS.SINLIB.SNET.010
```

Für den impliziten Linkvorgang im globalen Programm-Cache definierte Bibliotheken auflisten

Die Bibliotheken, für die der implizite Linkvorgang aktiviert ist, werden mit dem Kommando *posdbl* aufgelistet:

```
# posdbl -L
$TSOS.SINLIB.POSIX-SH.080
$TSOS.SINLIB.POSIX-BC.090.SHELL
```

Impliziten Linkvorgang für eine Bibliothek im globalen Programm-Cache aktivieren

Ist der implizite Linkvorgang für eine Bibliothek aktiviert, werden alle darin enthaltenen Programme beim Ausführen automatisch in den Programm-Cache geladen.

Der implizite Linkvorgang für eine zusätzliche Bibliothek wird mit dem Kommando *posdbl* aktiviert:

```
# posdbl -A \$TSOS.SINLIB.POSIX-BC.090.ROOT
$TSOS.SINLIB.POSIX-BC.090.ROOT: Successfully added.
```

Impliziten Linkvorgang für eine Bibliothek im globalen Programm-Cache deaktivieren

Der implizite Linkvorgang für eine Bibliothek wird mit dem Kommando *posdbl* deaktiviert:

```
# posdbl -R \$TSOS.SINLIB.POSIX-BC.090.ROOT
$TSOS.SINLIB.POSIX-BC.090.ROOT: Successfully deleted.
```

Core-Images im globalen Programm-Cache löschen

Core-Images im globalen Programm-Cache werden einzeln oder insgesamt mit dem Kommando *posdbl* gelöscht:

```
posdbl -r element
```

Bei einer Einzellöschung ist *element* der Name des Core-Images, wie in der mit dem Kommando *posdbl -l* ausgegebenen Auflistung. Wird als *element* ein Stern ('*') angegeben, so werden alle Core-Images gelöscht.

Core-Images in benutzerspezifischen Programm-Caches löschen

Core-Images in benutzerspezifischen Programm-Caches werden einzeln oder insgesamt mit dem Kommando *pdbl* gelöscht:

USERWIDE `pdbl -u -r` *element*

SESSIONWIDE `pdbl -s[`*sid*`] -r` *element*

Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

Bei einer Einzellöschung ist *element* der Name des Core-Images, wie in der Auflistung von Core-Images ausgegeben. Wird als *element* ein Stern ('*') angegeben, so werden alle Core-Images gelöscht.

Größe des globalen Programm-Caches verändern

Bei Bedarf kann die Größe des Programm-Caches im laufenden Betrieb verändert werden. Führen Sie dazu folgende Anweisungen aus:

```
usp -p DBLPOOL -v wert
    oder
usp -P DBLPOOL -v wert ——————————————————————————— (1)
posdbl -S >skriptname ——————————————————————————— (2)
posdbl -D ——————————————————————————————————————— (3)
posdbl -n ——————————————————————————————————————— (4)
skriptname ——————————————————————————————————————— (5)
posdbl -e linker | -e loader | both ————————————— (6)
```

(1) Die neue Größe des Programm-Caches wird festgelegt. Bei Angabe der Option *-p* wird nach einem Neustart des POSIX-Subsystem die alte Größe wieder eingestellt. Bei Angabe von *-P* gilt die neue Größe auch nach einem Neustart.

(2) Das Skript *skriptname* wird erzeugt, mit dem der aktuelle Inhalt des Programm-Caches wiederhergestellt werden kann.

(3) Der aktuelle Programm-Cache wird gelöscht.

(4) Ein neuer Programm-Cache mit der in Schritt (1) angegebenen Größe wird angelegt.

(5) Das Skript *skriptname* wird aufgerufen, um den bisherigen Inhalt des Programm-Caches wiederherzustellen.

(6) Nach dem Neuanlegen mit posdbl -n sind Linker und Loader deaktiviert und müssen je nach Bedarf wieder aktiviert werden.

Falls der Programm-Cache mit der Anweisung *usp* in Schritt (1) **verkleinert** wird, ist die Wiederherstellung des ursprünglichen Inhalts durch Aufruf des Skripts in Schritt (5) u. U. nicht mehr in vollem Umfang möglich.

Globalen Programm-Cache auflösen

Der globale Programm-Cache bleibt während der Laufzeit des POSIX-Subsystems erhalten. Aufgelöst wird er bei der POSIX-Subsystem-Terminierung.

Benutzerspezifische Programm-Caches auflösen

Benutzerspezifische Programm-Caches können mit dem Kommando *pdbl* aufgelöst werden:

USERWIDE `pdbl -u -d`

Falls der benutzerspezifische Programm-Cache der User-ID leer ist, wird er aufgelöst. Ansonsten wird er deaktiviert.

Mit dem Kommandoaufruf

`pdbl -u -D`

wird der benutzerspezifische Programm-Cache der User-ID unbedingt aufgelöst.

SESSIONWIDE `pdbl -s[`*sid*`] -d`

Falls der benutzerspezifische Programm-Cache der Sitzung *sid* leer ist, wird er aufgelöst. Ansonsten wird er deaktiviert.

Mit dem Kommandoaufruf

`pdbl -s[`*sid*`] -D`

wird der benutzerspezifische Programm-Cache der User-ID unbedingt aufgelöst.

Wird *sid* nicht angegeben, dann wird automatisch die aktuelle Sitzung genommen.

7 Dateisysteme verwalten und überwachen

Dieses Kapitel wendet sich an die Systemverwalter von BS2000 und POSIX. Es informiert Sie über

- das Verwalten von Dateisystemen (Einrichten, Ändern, Löschen, Ein- und Aushängen)
- die Überwachung von Dateisystemen mit *fsmond* (File System Monitor Dämon)

7.1 Dateisysteme verwalten

Die folgende Tabelle zeigt die notwendigen Privilegien für die POSIX-Verwaltungsaufgaben und dafür vorgesehene Kommandos bzw. Programme.

Aufgabe	Privileg	Kommando etc.	Eingabe in
POSIX-Dateisysteme ein- und aushängen	Root-Berechtigung	mount, mountall; umount, umountall	POSIX-Shell
POSIX-Dateisysteme einrichten, ändern und löschen	TSOS mit Root-Berechtigung	POSIX-Installations-programm	BS2000

Mehrere POSIX-Dateisysteme können zusammen einen Dateibaum bilden. Während einer POSIX-Session sind immer mindestens zwei Dateisysteme eingehängt: Das root-Dateisystem und das var-Dateisystem (siehe Abschnitt „Erstmalige Installation von POSIX“ auf Seite 115).

Das root-Dateisystem hat die höchste Hierarchie im Dateibaum. Der BS2000-Systemverwalter muss bei der Installation angeben, welches POSIX-Dateisystem das root-Dateisystem sein soll. Das root-Dateisystem wird automatisch beim Starten des Subsystems POSIX geöffnet.

7.1.1 Ein- und Aushängen von Dateisystemen

Ein Dateibaum kann durch das Verbinden zusätzlicher Dateisysteme mit dem root-Dateisystem noch erweitert werden. Dieser Vorgang wird „Einhängen“ genannt. Als Einhängepunkt kann jedes Dateiverzeichnis mit Ausnahme des root-Verzeichnisses im Dateibaum gewählt werden. Das Einhängen geschieht

- automatisch beim POSIX-Start, wenn das Dateisystem in der Datei */etc/vfstab* bzw. bei „Administrate POSIX file systems“ mit dem Attribut *automount=yes* definiert ist oder
- explizit mit dem Kommando *mount*. Nur POSIX-Verwalter können POSIX-Dateisysteme einhängen.

Wenn Sie Dateisysteme einhängen, beachten Sie bitte Folgendes:

- Während des Einhängens von Dateisystemen mit dem POSIX-Installationstool sollten parallel keine *mount*-Kommandos in der Shell abgesetzt werden.
- Der ursprüngliche Inhalt von Verzeichnissen, die als Einhängepunkt verwendet werden, ist „verborgen“ (Dateien, Unterverzeichnisse). Daher sollte das Verzeichnis */usr* nicht als Einhängepunkt genommen werden.

POSIX-Verwalter können eingehängte Dateisysteme mit dem Kommando *umount* wieder aushängen.

Ist für ein einzuhängendes Dateisystem Journaling aktiv, so wird das Dateisystem mit Journaling eingehängt. Ist nicht genügend Platz im Dateisystem für das Journal vorhanden, so wird es ohne Journaling eingehängt und auf der BS2000-Konsole wird auf den Engpass hingewiesen. Das Dateisystem können Sie mit dem POSIX-Installationsprogramm (siehe Kapitel „POSIX installieren“ auf Seite 103) oder mit dem Kommando *fsexpand* erweitern, siehe POSIX-Handbuch „Kommandos“ [1].

Die Größe des Journals hängt wie folgt von der Größe des Dateisystems ab:

Größe des Dateisystems	**Größe des Journals**
< 100 MB	1 MB
100 MB - 1600 MB	1 % der Grösse des Dateisystems
> 1600 MB	16 MB

Ist für ein einzuhängendes Dateisystem Journaling nicht aktiv, so wird das Dateisystem ohne Journaling eingehängt und gegebenenfalls der Platz eines vormals existierenden Journals freigegeben.

Weitere Informationen zum Journaling sowie zu seiner Aktivierung finden Sie unter:
– „Journaling für Dateisysteme“ auf Seite 45
– „Install POSIX subsystem (Subsystem POSIX neu einrichten)“ auf Seite 128 und 138
– „Administrate POSIX filesystems (POSIX-Dateisysteme verwalten)“ auf Seite 140

7.1.2 Lokale POSIX-Dateisysteme verwalten

Lokale POSIX-Dateisysteme können mit dem POSIX-Installationsprogramm eingerichtet, geändert und gelöscht werden. Dazu steht Benutzern mit dem Privileg TSOS und zusätzlicher Root-Berechtigung die Unterfunktion *Administrate POSIX filesystems* zur Verfügung (siehe Seite 131).

Lokale POSIX-Dateisysteme, die POSIX mit dem POSIX-Installationsprogramm bekannt gemacht wurden, kann ein Root-Berechtigter ein- oder aushängen. Zum Einhängen dienen die POSIX-Kommandos *mount* und *mountall*, zum Aushängen die POSIX-Kommandos *umount* und *umountall*. Sie sind im POSIX-Handbuch „Kommandos“ [1] ausführlich beschrieben.

7.1.3 bs2fs-Dateisysteme verwalten

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.

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

Für das Ein- und Aushängen des bs2fs-Containers und von bs2fs-Dateisystemen, können Sie die entsprechenden Ausprägungen der Kommandos *mount*, *mountall*, *umount*, *umountall* bzw. die Datei */etc/vfstab* verwenden. Außerdem unterstützen die Kommandos *show_pubset_export* und *start_bs2fsd* die Verwaltung von bs2fs-Dateisystemen.

Näheres dazu finden Sie im POSIX-Handbuch „BS2000-Dateisystem bs2fs“ [2].

7.1.4 Verteilte Dateisysteme verwalten

Das Softwareprodukt NFS ermöglicht verteilte Dateisysteme in einem heterogenen Rechnernetz. Verteilte Dateisysteme bedeutet:

- Lokale Datenbestände können Sie für die Bearbeitung an fernen Rechnern bereitstellen. Sie können beliebige Ausschnitte aus der Hierarchie des POSIX-Dateisystems bereitstellen. Die bereitgestellten Ausschnitte dürfen sich aber nicht überlappen. Für das Bereitstellen und Zurücknehmen der Bereitstellung können Sie die Kommandos *share*, *shareall*, *unshare* und *unshareall* bzw. die Datei */etc/dfstab* verwenden.
- Von fernen Rechnern bereitgestellte Datenbestände können Sie am lokalen Rechner im POSIX-Dateisystem einhängen und bearbeiten. Der Benutzer merkt nicht, dass sich das eingehängte Dateisystem physikalisch auf einem anderen Rechner befindet. Er kann mit den Dateien dieses Dateisystems so arbeiten, als wenn sie sich im lokalen POSIX-Dateisystem befinden würden.

  Für das Ein- und Aushängen von Datenbeständen, die ferne Rechner bereitstellen, können Sie die NFS-spezifischen Ausprägungen der Kommandos *mount*, *mountall*, *umount*, *umountall* bzw. die Datei */etc/vfstab* verwenden.

Näheres dazu steht im Handbuch „NFS (BS2000/OSD)“ [8].

7.1.5 Dateisystem auf Konsistenz prüfen

Mit dem POSIX-Kommando *fsck* kann die Konsistenz eines Dateisystems geprüft werden. Inkonsistenzen können im Dialog mit dem Benutzer korrigiert werden.

Das POSIX-Kommando *fsck* ist im POSIX-Handbuch „Kommandos“ [1] ausführlich beschrieben. Die Konsistenzprüfung bestimmter oder aller Dateisysteme kann auch beim Starten des Subsystems POSIX erzwungen werden, siehe Abschnitt „POSIX-Subsystem starten“ auf Seite 154.

7.1.6 Dateisystem erweitern

POSIX bietet eine platzsparende direkte Methode, bei der das Dateisystem ohne vorheriges Kopieren auf BS2000-Ebene (BS2000-Container) vergrößert wird. Die internen Strukturen des POSIX-Dateisystems werden anschließend durch POSIX an die neue Größe angepasst. Dies hat den Vorteil, dass nur der zusätzliche Plattenspeicherplatz für das neue Dateisystem benötigt wird.

Diese Dateisystemerweiterung ist online, offline, im Dialog und im Batch möglich. Wird ein Dateisystem online erweitert, dann wird das Dateisystem nach erfolgreicher Erweiterung unabhängig von der Automount-Einstellung montiert, falls es vorher schon montiert war.

Im Einzelnen haben Sie folgende Möglichkeiten, um Dateisysteme zu erweitern:

- POSIX-Installationsprogramm im Dialogmodus, offline (POSIX nicht gestartet)

  Mit dieser Variante können Sie alle POSIX-Dateisysteme erweitern, auch das *root*- und *var*-Dateisystem. Gehen Sie wie folgt vor:

  – Rufen Sie das POSIX-Installationsprogramm auf mit `/START-POSIX-INSTALLATION`

  – Wählen Sie in der Startmaske die Funktion *Expand of POSIX filesystem* aus.

  – Tragen Sie in der Folgemaske das gewünschte Dateisystem und die neuen Größen ein, siehe Abschnitt „Expand POSIX filesystems (POSIX-Dateisystem erweitern)“ auf Seite 130.

- POSIX-Installationsprogramm im Dialogmodus, online (POSIX gestartet)

  Mit dieser Variante können Sie alle POSIX-Dateisysteme außer *root* und *var* erweitern, vorausgesetzt das Dateisystem lässt sich demontieren. Gehen Sie wie folgt vor:

  – Rufen Sie das POSIX-Installationsprogramm auf mit `/START-POSIX-INSTALLATION`

  – Wählen Sie in der Startmaske die Funktion *Administrate POSIX filesystems* aus.

  – Markieren Sie in der Maske *Administrate POSIX filesystems* das gewünschte Dateisystem und wählen Sie im Eingabefeld die Option E (expand), siehe Abschnitt „Administrate POSIX filesystems (POSIX-Dateisysteme verwalten)“ auf Seite 131.

  – Tragen Sie in der Folgemaske die neuen Größen ein, siehe Abschnitt „Expand POSIX filesystems (POSIX-Dateisystem erweitern)“ auf Seite 130.

- POSIX-Installationsprogramm im Batchmodus, offline oder online

  Beim offline-Aufruf lassen sich alle POSIX-Dateisystem erweitern, beim online-Aufruf nur die demontierbaren Dateisysteme; *root* und *var* sind nicht online erweiterbar, da sie bei gestartetem POSIX nicht demontierbar sind.

  Gehen Sie wie folgt vor:

  – Erstellen Sie eine Parameterdatei mit der Identifikationszeile *[ExpandFileSystem]*, siehe Abschnitt „Expand POSIX filesystems (POSIX-Dateisysteme erweitern)“ auf Seite 139.

  – Rufen Sie das POSIX-Installationsprogramm auf mit
  `/START-POSIX-INSTALLATION INPUT-INTERFACE=*FILE(<parameterdatei>`

  Beispiel

  ```
  #
  # Batch Installationsdatei
  #
  [ExpandFileSystem]
  # <file>;<size>
  $SYSROOT.FS.ROOT;10000    #  root filesystem 10 000 PAM pages more
  $SYSROOT.FS.VAR;20000     #  var filesystem 20 000 PAM pages more
  ```

- Kommando *fsexpand*

  Damit können alle Dateisysteme außer *root* und *var* erweitert werden.

  – Melden Sie sich an der POSIX-Shell an.

  – Geben Sie das Kommando *fsexpand* mit den gewünschten Optionen ein. Es besitzt folgende Syntax, siehe POSIX-Handbuch „Kommandos“ [1]:

  fsexpand[␣**-i**][␣**-p**␣pamseiten|␣**-c**␣zylindergruppen]␣gerät

  Für *gerät* geben Sie das zu erweiternde Dateisystem an.

7.2 Dateisysteme überwachen mit fsmond (File System Monitor Dämon)

Der File System Monitor Dämon *fsmond* wird über rc-Skripts beim Hochfahren des POSIX-Subsystems automatisch gestartet und beim Herunterfahren des POSIX-Subsystems automatisch beendet (siehe unten). Er überwacht die kritischen Dateisysteme */* (root) und */var*.

Der Dämon läuft periodisch alle *intervall* Sekunden und überprüft dabei die prozentuale Belegung der oben angegebenen Dateisysteme. Wird der Schwellwert *warnlimit* überschritten, so wird an der Konsole die Warnung *POS4030* ausgegeben. Wird der Schwellwert *fehlerlimit* überschritten, so wird an der Konsole die zu beantwortende Meldung *POS4031* ausgegeben.

Syntax **fsmond**[␣**-t**␣intervall][␣**-w**␣warnlimit][␣**-e**␣fehlerlimit]

-t intervall
Mit dieser Option wird die Periode in Sekunden angegeben, in der die Überprüfung des Dateisystems stattfinden soll. Der zulässige Wert liegt zwischen 1 und 3600. Standardwert ist 120.

-w warnlimit
Mit dieser Option wird in Prozent angegeben, ab welcher Belegung (eines) der Dateisysteme eine Warnmeldung an der Konsole ausgegeben werden soll. Falls sich dieser Wert beim nächsten Durchlauf von *fsmond* verschlechtert hat, wird wiederum eine Warnmeldung an der Konsole ausgegeben. Der zulässige Wert liegt zwischen 1 und 99. Standardwert ist 80.

-e fehlerlimit
Mit dieser Option wird in Prozent angegeben, ab welcher Belegung (eines) der Dateisysteme eine zu beantwortende Meldung an der Konsole ausgegeben werden soll. Der zulässige Wert liegt zwischen 1 und 99. Standardwert ist 90.

Der Wert von *fehlerlimit* muss um mindestens 5 größer sein, als der Wert von *warnlimit*.

Hinweis **Belegungsquote**

Die in Prozent angegebene Belegungsquote für *warnlimit* und *fehlerlimit* entspricht dem Füllgrad, wie er mit dem Kommandoaufruf `df -v` ausgegeben wird (Verhältnis von verfügbaren und belegten Blöcken für privilegierte Benutzer).

rc-Skripts

Die rc-Skripts zum automatischen Starten/Beenden des Dämon lauten wie folgt:
/etc/rc2.d/S14fsmond (Starten)
/etc/rc0.d/K14fsmond (Beenden)

Standardwerte ändern

Wenn der Dämon mit anderen Werten als den Standardwerten laufen soll, muß der Aufruf des Dämons im rc-Skript */etc/rc2.d/S14fsmond* geändert werden. Nachdem die Werte geändert wurden, muss das POSIX-Subsystem beendet und neu gestartet werden. Wenn die angegebenen Werte inkonsistent sind, wird der Dämon mit einer Fehlermeldung beendet, die in die syslog-Protokolldatei ausgegeben wird.

Status überprüfen

Beispiele für die Überprüfung, ob der Dämon ordnungsgemäß gestartet wurde:

```
/ # /etc/rc2.d/S14fsmond status
fsmond is running (pid=25)

/ # /etc/rc2.d/S14fsmond status
fsmond is not running

/ # ps –ef |grep fsmond
ROOT    25     1  0 10:24:32 ?        0:00 [fsmond]
```

Für den Fall, dass der Dämon nicht gestartet ist, steht die Fehlerursache in der Regel in der Datei */var/adm/syslog*.

Empfehlungen zum Betrieb des Dämons

Man sollte den Dämon generell automatisch durch das POSIX-Subsystem starten und beenden lassen. Der Dämon läuft in diesem Fall als Hintergrundprozess unter *$SYSROOT*.

Die Möglichkeit, den Dämon über rc-Skript-Aufrufe manuell zu starten und zu beenden, sollte nur in Ausnahmefällen und beispielsweise zu Testzwecken genutzt werden:

/etc/rc2.d/S14fsmond stop
/etc/rc2.d/S14fsmond start

Konsolmeldungen

```
POS4030 FSMOND: WARNING! (&00)% ALLOCATION EXCEEDS WARNLIMIT ((&01)%) ON FILE
SYSTEM "(&02)"

?POS4031 FSMOND: ATTENTION! (&00)% ALLOCATION EXCEEDS ERROR LIMIT ((&01)%) ON
FILE SYSTEM "(&02)"
```

Die Meldung *POS4031* kann mit einer beliebigen Eingabe (z. B. *tsn*.) beantwortet werden.

Solange die Meldung *POS4031* nicht beantwortet ist, ist der Dämon in Wartestellung und es unterbleibt die intervallmäßige Überprüfung des jeweils anderen Dateisystems.

8 POSIX-Benutzer verwalten

Dieses Kapitel wendet sich an BS2000-Systemverwalter, BS2000-Gruppenverwalter und POSIX-Verwalter.

Jeder BS2000-Benutzer ist gleichzeitig auch POSIX-Benutzer. Außer einer BS2000-Benutzerkennung mit gültigen individuellen POSIX-Benutzerattributen (siehe Seite 185) sind keine weiteren Bedingungen zu erfüllen, um Zugang zu POSIX und seinen Schnittstellen zu erhalten.

Die POSIX-Benutzerverwaltung ist in die BS2000-Benutzerverwaltung integriert. Dieses Kapitel beschreibt die Schnittstellen zur Verwaltung der POSIX-Benutzerattribute einer BS2000-Benutzerkennung. Diese Schnittstellen sind Bestandteil des Bausteins SRPM (**S**ystem **R**esources and **P**rivileges **M**anagement), der im BS2000-Grundausbau und im Softwareprodukt SECOS enthalten ist. Näheres zu SRPM finden Sie in den Handbüchern „Einführung in die Systembetreuung“ [16] und „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9]. Das Softwareprodukt SECOS muss nicht installiert sein, um mit POSIX arbeiten zu können.

8.1 Übersicht über Privilegien und Aufgaben

Für POSIX wird das Privileg POSIX-ADMINISTRATION neu eingeführt. Inhaber dieses Privilegs werden in diesem Handbuch kurz POSIX-Verwalter genannt. Sie haben folgende Aufgaben und Rechte:

- Verwaltung der POSIX-Benutzerattribute aller BS2000-Benutzerkennungen auf allen Pubsets (siehe Seite 185)
- Verwaltung der Standardwerte für die POSIX-Benutzerattribute auf allen Pubsets (siehe Seite 190)

Das Privileg POSIX-ADMINISTRATION ist automatisch an die Systemkennung SYSROOT geknüpft. Dieses Privileg kann SYSROOT nicht entzogen werden.

Der Sicherheitsbeauftragte (Privileg SECURITY-ADMINISTRATION) kann das Privileg POSIX-ADMINISTRATION auch anderen BS2000-Benutzerkennungen verleihen und entziehen. Dazu wird das Softwareprodukt SECOS benötigt.

SYSROOT ist das POSIX-Gegenstück zur Systemverwalterkennung *root* in UNIX-Systemen. SYSROOT wird beim Start des BS2000-Systems eingerichtet und erhält automatisch die Benutzernummer 0. SYSROOT kann keine andere Benutzernummer zugewiesen werden.

Inhaber des Privilegs USER-ADMINISTRATION erhalten zusätzlich die Berechtigung, die POSIX-Benutzerattribute zu verwalten. Sie sind diesbezüglich dem POSIX-Verwalter gleichgestellt.

Die Berechtigung des Gruppenverwalters der Gruppe *UNIVERSAL wird auf die POSIX-Benutzerattribute ausgedehnt. Er ist bei der Verwaltung der POSIX-Benutzerattribute auf dem von ihm verwalteten Pubset den Inhabern des Privilegs USER-ADMINISTRATION gleichgestellt. Deshalb gelten für ihn nicht die im folgenden aufgeführten Einschränkungen für Gruppenverwalter seiner Hierarchie.

Gruppenverwalter dürfen ebenfalls die POSIX-Benutzerattribute verwalten. Allerdings gelten für sie folgende Einschränkungen:

- Sie können nicht die Standardwerte für die POSIX-Benutzerattribute verwalten.
- Die Art der POSIX-Benutzerattribute, die ihrer Verwaltung unterstellt sind, hängt von ihrer Autorisierung ab (ADM-AUTHORITY).
- Der Wertebereich der POSIX-Benutzerattribute ist für sie eingeschränkt.
- Sie können nur die Gruppen- und Untergruppenmitglieder verwalten, die ihnen unterstellt sind.

Die folgende Tabelle gibt Ihnen einen Überblick über die Aufgaben und Tätigkeiten, die im Zusammenhang mit der POSIX-Benutzerverwaltung anfallen. Dazu sind bestimmte Privilegien erforderlich. Die Aufgaben müssen entweder auf der BS2000-Ebene, auf der Shell-Ebene oder auf beiden Ebenen erledigt werden.

Aufgabe/Tätigkeit	**Privileg**	**Kommando etc.**	**Eingabe in**	**siehe**
POSIX-Status anzeigen	SUBSYSTEM-MANAGEMENT	/SHOW-POSIX-STATUS	BS2000	Seite 247
BS2000-Kennungen das Privileg POSIX-ADMINISTRATION verleihen oder entziehen	SECURITY-ADMIN.	/SET-PRIVILEGE /RESET-PRIVILEGE	BS2000	Handbuch „SECOS“ [9]
POSIX-Benutzerattribute vergeben	USER-ADMIN. oder POSIX-ADMIN. oder BS2000-Gruppenverwalter (mit Einschränkungen)	/MODIFY-POSIX-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES	BS2000	Seite 185
Einer BS2000-Benutzerkennung eine individuelle Benutzernummer zuordnen	USER-ADMIN.	/MODIFY-POSIX-USER-ATTRIBUTES	BS2000	Seite 186
POSIX-Gruppen im BS2000 verwalten	USER-ADMIN. oder POSIX-ADMIN. oder Gruppenverwalter	/MODIFY-POSIX-USER-ATTRIBUTES: Benutzerattribut GROUP-NUMBER	BS2000	Seite 187, Seite 189
POSIX-Gruppen in POSIX verwalten	Root-Berechtigung	Datei /etc/group	POSIX-Shell	Seite 187, Seite 189
Neue POSIX-Benutzer eintragen	USER-ADMIN., für /ADD-POSIX-USER TSOS erforderlich	/ADD-USER und /ADD-POSIX-USER	BS2000	Seite 189
Standardwerte für POSIX-Benutzerattribute festlegen	USER-ADMIN. oder POSIX-ADMIN. oder BS2000-Gruppenverwalter (mit Einschränkungen)	/MODIFY-POSIX-USER-DEFAULTS /SHOW-POSIX-USER-DEFAULTS	BS2000	Seite 190

Aufgabe/Tätigkeit	Privileg	Kommando etc.	Eingabe in	siehe
Zugangsberechtigung für den Benutzer eines fernen Rechners erteilen	USER-ADMIN. oder BS2000-Gruppen-verwalter (mit Einschränkungen)	/SET-LOGON-PROTECTION /MODIFY-LOGON-PROTECTION /SHOW-LOGON-PROTECTION	BS2000	Seite 191
Abrechnungsnummer für den Systemzugang über einen fernen Rechner eintragen	USER-ADMIN. oder BS2000-Gruppen-verwalter (mit Einschränkungen)	/ADD-USER /MODIFY-USER-ATTRIBUTES /SHOW-USER-ATTRIBUTES	BS2000	Seite 192
POSIX-Benutzer löschen	POSIX-ADMIN.	/MODIFY-POSIX-USER-ATTRIBUTES	BS2000	Seite 192
POSIX-Benutzer löschen	Root-Berechtigung	rmdir	POSIX-Shell	Seite 192
Informationen über Einträge im Benutzerkatalog für die eigenen Benutzerkennungen ausgeben Benutzerinformationen per Programm lesen	STD-PROCESSING	/SHOW-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES Makro SRMUINF	BS2000	Seite 256 Seite 248 Seite 193

8.2 POSIX-Benutzerattribute vergeben

Die POSIX-Benutzerattribute charakterisieren den Benutzer, treffen Voreinstellungen und legen Berechtigungen fest. POSIX-Benutzerattribute sind: *Benutzernummer*, *Gruppennummer*, *Kommentar*, *Login-Verzeichnis* und *Programm*. Sie entsprechen den Einträgen im Benutzerkatalog */etc/passwd* eines UNIX-Systems. Die Datei */etc/passwd* existiert aber in POSIX nicht.

Die POSIX-Benutzerattribute werden beim Einrichten einer BS2000-Benutzerkennung mit Standardwerten belegt (siehe Seite 190). Die POSIX-Benutzerattribute einer BS2000-Benutzerkennung können mit dem Kommando /MODIFY-POSIX-USER-ATTRIBUTES (siehe Seite 228) geändert werden.

Bereits existierenden BS2000-Benutzerkennungen wird beim First-Start oder beim Versionsumstieg automatisch die Standard-Benutzernummer zugeordnet (siehe Seite 186). Die Standard-Benutzernummer einer existierenden BS2000-Benutzerkennung kann mit dem Kommando /MODIFY-POSIX-USER-ATTRIBUTES (siehe Seite 228) geändert werden.

Die POSIX-Benutzerattribute sind Bestandteil des BS2000-Benutzereintrags im BS2000-Benutzerkatalog SYSSRPM.

Benutzernummer

Sie legt unter POSIX fest, wer der Eigentümer von Dateien und Dateierzeichnissen ist. Im Gegensatz zum BS2000 ist hier die BS2000-Benutzerkennung - oder besser der Login-Name - zweitrangig. Deshalb muss jeder BS2000-Benutzerkennung, die POSIX nutzen möchte, eine individuelle Benutzernummer zugeordnet werden (siehe Abschnitt „Einer BS2000-Benutzerkennung eine individuelle Benutzernummer zuordnen“ auf Seite 186).

Eine Sonderrolle hat die Benutzernummer 0: Sie gibt ihrem Inhaber - zusammen mit der Gruppennummer 0 - die POSIX-Verwalter-Berechtigung, die im folgenden kurz Root-Berechtigung genannt wird. Die Systemkennung SYSROOT hat standardmäßig die Root-Berechtigung. Die Systemkennung TSOS erhält bei der Erstinstallation automatisch die Root-Berechtigung.

Gruppennummer

Sie legt die Zugehörigkeit zu einer POSIX-Gruppe fest. Diese POSIX-Gruppe erhält die Zugriffsrechte der Benutzerklasse „Gruppe“ für alle POSIX-Dateien, die dieser Benutzer erstellt.
Der Gruppennummer kann durch einen Eintrag im POSIX-Gruppenkatalog */etc/group* ein Gruppenname zugeordnet werden (siehe Seite 188).

Kommentar

An dieser Stelle kann ein Kommentar zum Eigentümer der BS2000-Benutzerkennung eintragen werden.

Login-Verzeichnis

Es bestimmt den absoluten Pfadnamen des Dateiverzeichnisses, in das der Benutzer automatisch gelangt, wenn er mit POSIX verbunden wird. Dies ist im Falle

– eines gemischten Programms, das aus dem BS2000 aufgerufen wird, der erste Aufruf einer POSIX-Schnittstelle (POSIX-SVCs).

– eines Benutzers der POSIX-Shell die Verarbeitung des Kommandos /START-POSIX-SHELL

– eines *rlogin*-Aufrufs die Verarbeitung des *rlogin*.

Programm

Dieses POSIX-Benutzerattribut bezeichnet den Namen des Programms, das gestartet wird, nachdem der Benutzer das Kommando /START-POSIX-SHELL (siehe Seite 263) aufgerufen hat.

8.3 Einer BS2000-Benutzerkennung eine individuelle Benutzernummer zuordnen

Eine BS2000-Benutzerkennung wird unter POSIX über die Benutzernummer identifiziert. Deshalb muss jeder BS2000-Benutzerkennung, von der aus POSIX nutzbar sein soll, eine Benutzernummer (siehe Seite 190) zugeordnet werden:

- Jeder existierenden BS2000-Benutzerkennung wird beim First-Start oder beim Versionsumstieg automatisch die Standard-Benutzernummer zugeordnet.
- Jede neu einzurichtende BS2000-Benutzerkennung erhält bereits bei ihrer Definition die Standard-Benutzernummer.

Dadurch gibt es eine Vielzahl von BS2000-Benutzerkennungen, die alle dieselbe Standard-Benutzernummer haben.

POSIX-Verwalter und BS2000-Systemverwalter können den Wert der Standard-Benutzernummer mit dem BS2000-Kommando /MODIFY-POSIX-USER-DEFAULTS festlegen.
Sie müssen jeder BS2000-Benutzerkennung anstelle der Standard-Benutzernummer eine individuelle Benutzernummer zuordnen, bevor POSIX unter dieser BS2000-Benutzerkennung genutzt werden kann. Dazu steht das BS2000-Kommando

/MODIFY-POSIX-USER-ATTRIBUTES zur Verfügung. Es wird eine Warnung ausgegeben, wenn eine Benutzernummer mehrfach vergeben wird, außer wenn es sich um die Standard-Benutzernummer handelt.

Die Benutzernummern von 0 bis 99 sind für privilegierte Benutzer (Systemkennungen) reserviert. Die Benutzernummern ab 100 sind für nichtprivilegierte Benutzer vorgesehen.

Verschiedene BS2000-Benutzerkennungen mit der gleichen Benutzernummer werden auf die gleiche POSIX-Benutzerkennung abgebildet. Zwischen der BS2000-Benutzerkennung und der Benutzernummer besteht aber keine technische Abhängigkeit.

Besonders in einem Rechnernetz mit UNIX-Systemen ist eine eindeutige Zuordnung von BS2000-Benutzerkennung und Benutzernummer wichtig, da hier über Rechner- und Systemgrenzen hinweg konsistente Benutzeridentifikationen auf Basis der Benutzernummer vorausgesetzt werden.

8.4 BS2000- und POSIX-Gruppen verwalten

Da sich die Gruppenverwaltung in POSIX und im BS2000 in wesentlichen Punkten unterscheidet (siehe Abschnitt „Gruppenverwaltung“ auf Seite 55), bestehen POSIX- und BS2000-Gruppen unabhängig nebeneinander und werden getrennt verwaltet.

Der POSIX-Gruppenkatalog ist kein Bestandteil von SRPM/SECOS. Deshalb muss der Root-Berechtigte die POSIX-Gruppen separat im POSIX-Gruppenkatalog */etc/group* definieren und verwalten. Es ist seine Aufgabe, alle Veränderungen einer BS2000-Benutzerkennung (Einrichten, Gruppenwechsel, Löschen) separat im POSIX-Gruppenkatalog */etc/group* (siehe Seite 188) nachzuvollziehen.

Die Gruppennummer wird beim Anschluss des Benutzers an POSIX ohne weitere Prüfung den POSIX-Benutzerattributen entnommen. Daher liegt es im Ermessen des POSIX-Verwalters und des Root-Berechtigten, das Attribut GROUP-NUMBER und den entsprechenden POSIX-Gruppeneintrag in einer separaten Aktion aufeinander abzustimmen.

Ein BS2000-Gruppenverwalter kann die Rolle des POSIX-Verwalters für die Mitglieder seiner Gruppe übernehmen. Damit er die BS2000-Gruppenstruktur auf die POSIX-Gruppenstruktur abbilden kann, wird folgende Konvention getroffen:

„Die Gruppennummer der zur BS2000-Gruppe korrespondierenden POSIX-Gruppe ist gleich der Gruppennummer des BS2000-Gruppenverwalters.“

Ein BS2000-Gruppenverwalter hat folgende Rechte:

- Er darf seine Gruppennummer an seine BS2000-Gruppenmitglieder weitergeben. Wenn ein übergeordneter Gruppenverwalter in Vertretung des Gruppenverwalters seiner Untergruppe handelt, kann er nur dessen Gruppennummer zuweisen.
- Er kann ein BS2000-Gruppenmitglied aus der POSIX-Gruppe ausschließen, indem er ihm die Standard-Gruppennummer zuweist.

Eine darüber hinausgehende Verwaltung der POSIX-Gruppen muss zentral durch einen POSIX-Verwalter erfolgen.

Beispiel

Die BS2000-Gruppe mit dem Gruppennamen `A5` enthält folgende Benutzer:
`POSIXTST`, `POSIX001` und `POSIX002`.
Die BS2000-Gruppe mit dem Gruppennamen `A7` enthält folgende Benutzer:
`MANUAL01` und `MANUAL02`.

Bei POSIX können dann ebenfalls Gruppen mit der Gruppennummer 5 (`POSIXTST`, `POSIX001` und `POSIX002`) und der Gruppennummer 7 (`MANUAL01` und `MANUAL02`) definiert werden.
Eine Doppelmitgliedschaft in beiden Gruppen - z.B. wenn `MANUAL01` zusätzlich Mitglied der Gruppe mit der Nummer 5 werden möchte - ist aber nur möglich, wenn von der BS2000-Gruppendefinition abgewichen wird.

POSIX-Gruppenkatalog */etc/group* der POSIX-Shell verwalten

Jeder Benutzer ist einer Benutzergruppe zugeordnet, nachdem ihm der BS2000-Systemverwalter eine numerische Gruppennummer zugewiesen hat. Dieser Gruppennummer kann der POSIX-Verwalter oder ein Root-Berechtigter im POSIX-Gruppenkatalog */etc/group* einen Gruppennamen zuordnen oder eine neue Benutzergruppe definieren.
Im BS2000 gibt es kein Äquivalent zum POSIX-Gruppenkatalog */etc/group*.

Der POSIX-Gruppenkatalog */etc/group* wird bei der Erstinstallation angelegt.
Er besteht aus Zeilen mit folgendem Aufbau:

gruppenname: : gruppennummer : benutzerkennung[,...]

gruppenname
Name, der für diese Gruppe vergeben werden soll.

gruppennummer
Numerische Gruppennummer, die im BS2000-Benutzerkatalog SYSSRPM festgelegt wurde. Dieser Gruppennummer kann mit <gruppenname> ein Gruppenname zugeordnet werden.

benutzerkennung
Eine oder mehrere Benutzerkennungen, die dieser Benutzergruppe angehören sollen. Wenn mehrere Benutzerkennungen angegeben werden, müssen Sie diese durch Kommata trennen.
Eine Benutzerkennung kann mehreren Benutzergruppen zugeordnet sein.

Die Einträge müssen voneinander durch einen Doppelpunkt getrennt sein. Wenn Sie den Eintrag für das Kennwort weglassen, müssen Sie den folgenden Doppelpunkt trotzdem angeben. Die Einträge für jede Benutzergruppe müssen jeweils in einer neuen Zeile beginnen.

Der POSIX-Gruppenkatalog */etc/group* enthält nach der Erstinstallation folgende Benutzergruppen:

```
SYSROOT       (Gruppennummer: 0, Member : SYSROOT)
OTHER         (Gruppennummer: 1)
SYSBIN        (Gruppennummer: 2)
SYSSYS        (Gruppennummer: 3, Members: SYSROOT, SYSBIN)
MAIL          (Gruppennummer: 6, Member : SYSROOT)
TTY           (Gruppennummer: 7)
LP            (Gruppennummer: 8)
USROTHER      (Gruppennummer: 100)
DFS_STARTGID  (Gruppennummer: 2000)
```

8.5 Neue POSIX-Benutzer eintragen

Nach dem Einrichten eines neuen BS2000-Benutzers mit dem Kommando /ADD-USER (siehe Seite 198) sind die POSIX-Benutzerattribute *Benutzernummer*, *Gruppennummer*,*Login-Verzeichnis* und *Programm* mit pubsetspezifischen Standardwerten belegt (siehe Abschnitt „Standardwerte für POSIX-Benutzerattribute festlegen“ auf Seite 190). Damit der neue BS2000-Benutzer POSIX nutzen kann, muss der POSIX-Verwalter oder der BS2000-Systemverwalter die Standardwerte ändern.

Hierzu wird eine Prozedur angeboten, die mit Hilfe des Kommandos /ADD-POSIX-USER (siehe Seite 195) aufgerufen wird. Diese Prozedur kann nur unter der Systemkennung TSOS ablaufen. Vor dem Start der Prozedur muss das Subsystem POSIX gestartet werden.

Die Prozedur legt ein Home-Verzeichnis für den Benutzer an und trägt den Namen in das entsprechende POSIX-Benutzerattribut ein.

8.6 Standardwerte für POSIX-Benutzerattribute festlegen

Die POSIX-Benutzerattribute werden beim Einrichten einer BS2000-Benutzerkennung mit den Standardwerten des angegebenen Pubsets initialisiert. Der Operandenwert *BY-POSIX-USER-DEFAULTS bezieht sich jeweils auf die Standardwerte des angegebenen Pubsets.

Die POSIX-Benutzerattribute werden automatisch beim First-Start oder beim Versionsumstieg eingerichtet; sie werden mit fest vorgegebenen Werten initialisiert:

USER-NUMBER	=	100
GROUP-NUMBER	=	1
COMMENT	=	*NONE
DIRECTORY	=	*ROOT
PROGRAM	=	*SHELL

Während der POSIX-Erstinstallation wird der Standardwert von DIRECTORY auf */home/gast* und der von GROUP-NUMBER auf 100 geändert.

Allen BS2000-Benutzerkennungen außer TSOS und SYSROOT wird zunächst die Standard-Benutzernummer und die Standard-Gruppennummer zugewiesen.

Die Systemkennungen TSOS und SYSROOT werden beim First-Start mit der fest vorgegebenen Benutzernummer 0 und der Gruppennummer 0 eingerichtet. Die Benutzernummer kann nicht geändert werden; für die Gruppennummer gibt es keine Einschränkungen.

Die Standardwerte für die Benutzerattribute können mit dem Kommando /MODIFY-POSIX-USER-DEFAULTS (siehe Seite 234) geändert und mit dem Kommando /SHOW-POSIX-USER-DEFAULTS (siehe Seite 256) angezeigt werden.

8.7 Zugangsberechtigung für Benutzer eines fernen Rechners erteilen

Wenn das Softwareprodukt SECOS eingesetzt wird, kann für bereits existierende BS2000-Benutzerkennungen festgelegt werden, ob dem Benutzer eines fernen Rechners mit dem Kommando *rlogin* Zugang zum System erlaubt wird (siehe Abschnitt „Zugang zur POSIX-Shell“ auf Seite 63). Dazu steht der Operand POSIX-RLOGIN-ACCESS bei den BS2000-Kommandos /SET-LOGON-PROTECTION (siehe Seite 240) und /MODIFY-LOGON-PROTECTION (siehe Seite 220) zur Verfügung.

Mit dem Kommando /SHOW-LOGON-PROTECTION (siehe Seite 245) können die Schutzattribute angezeigt werden.

Zugangsschutz klassifizieren mit SECOS

Wenn SECOS eingesetzt wird, dann können Sie die Zugriffe von einem fernen Rechner mit den Kommandos /SET-LOGON-PROTECTION und /MODIFY-LOGON-PROTECTION genauer klassifizieren.

Dabei können Sie für eine Benutzerkennnung einstellen:

- ob und von welchen Stationen aus der *rlogin*-Zugang erlaubt ist (Operand POSIX-RLOGIN-ACCESS).
- ob und von welchen Stationen aus der Zugang über Remote-Kommandos (*rsh*, *rcp*, ...) erlaubt ist (Operand POSIX-REMOTE-ACCESS).
- ob der Zugang über *rlogin* oder Remote-Kommandos mit einem Guard geschützt wird. Die Guards lassen sich für den *rlogin*-und Remote-Kommando-Zugang getrennt zuordnen.
- ob Tasks unter dieser Benutzerkennung mit Hilfe von *ufork* ihre Benutzerkennung ändern dürfen.

8.8 Abrechnungsnummer für Systemzugang über fernen Rechner eintragen

Mit den Kommandos /ADD-USER (siehe Seite 198), /MODIFY-USER-ATTRIBUTES (siehe Seite 237) und /SHOW-USER-ATTRIBUTES (siehe Seite 256) können Sie die Abrechnungsnummer beim Systemzugang über einen fernen Rechner verwalten.

8.9 POSIX-Benutzer löschen

Zum Löschen eines POSIX-Benutzers müssen folgende Aufgaben durchgeführt werden:

- POSIX-Verwalter:
  Individuelle Benutzernummer mit dem BS2000-Kommando /MODIFY-POSIX-USER-ATTRIBUTES (siehe Seite 228) auf den Standardwert zurücksetzen.
- Root-Berechtigter:
  Ggf. das Home-Verzeichnis mit dem POSIX-Kommando *rmdir* auf der POSIX-Shell löschen

  Eventuell müssen vorher die Dateien des Benutzers gelöscht oder anderen Benutzern zugewiesen werden.

  Das POSIX-Kommando *rmdir* ist im POSIX-Handbuch „Kommandos“ [1] beschrieben.

8.10 Benutzerinformationen per Programm lesen

Der BS2000-Systemverwalter legt für jede BS2000-Benutzerkennung einen Eintrag im Benutzerkatalog an. Der Eintrag enthält u.a.:

- BS2000-Benutzerkennung, Kennwortberechtigung
- Angaben zu Systemressourcen, die der Benutzer in Anspruch nehmen kann (CPU-Zeit, Speicherplatz, ...)
- besondere Rechte des Benutzers (privilegierter Zugriff, ...)
- Daten für die Abrechnung (Accounting)

Die Daten aus dem Benutzerkatalog können mit dem Makro SRMUINF gelesen und in einen vorher festgelegten Bereich übertragen werden. Je nach Angabe werden die Daten für die Abrechnung (Accounting), die benutzerspezifischen Daten oder der gesamte Eintrag einer BS2000-Benutzerkennung aus dem Benutzerkatalog ausgegeben.

An den Operanden und Operandenwerten des Makros SRMUINF ändert sich durch POSIX nichts. Zusätzlich wird aber die POSIX-Abrechnungsnummer gekennzeichnet. Der abrechnungsspezifische Teil der Ausgabe enthält zu jeder einzelnen Abrechnungsnummer einen Indikator, der sie für die Abrechnung des Remote-Login-Systemlaufs bestimmt.
Nähere Informationen zum Makro SRMUINF finden Sie im Handbuch „Makroaufrufe an den Ablaufteil“ [30].

Die Daten aus dem Benutzerkatalog können auch mit den CRTE-Makros *getlogin*, *getpwent*, *putpwent* etc. gelesen werden (siehe Handbuch „CRTE (BS2000/OSD)“ [7]).

8.11 POSIX-Standard-Jobklassen

Durch den *fork*-Mechanismus des POSIX-Subsystems können viele POSIX-Tasks entstehen. Ab BS2000/OSD V9.0 können solche POSIX-Tasks unabhängig von den sonstigen Batch- und Dialog-Aufträgen gesteuert werden.

Mit der ab BS2000/OSD V9.0 verfügbaren JMU-Anweisung SET-POSIX-JOB-CLASS-DEFAULT (siehe Handbuch „Dienstprogramme“ [17]) kann einzelnen oder allen Benutzerkennungen eine Standard-Jobklasse für POSIX-Tasks zugewiesen werden. Diese Jobklasse muss zuvor mit DEFINE-JOB-CLASS definiert worden sein. Ist keine POSIX-Standard-Jobklasse festgelegt, ist das Verhalten bei *fork* wie bisher.

Beispiel:

```
/START-JMU
//SET-MODIFICATION-MODE ...
//SET-POSIX-JOB-CLASS-DEFAULT -
//    NAME=JCBPSX1, ACTION=*ADD, USER=*ALL ——————————————————————  (1)
//SET-POSIX-JOB-CLASS-DEFAULT -
//    NAME=JCBPSX0, ACTION=*ADD, USER=(TSOS,SYSROOT) ———————————————  (2)
//END
```

(1) Es wird eine systemweit für alle Benutzerkennungen gültige POSIX-Standard-Jobklasse vereinbart.

(2) Für die Benutzerkennungen TSOS und SYSROOT wird eine abweichende POSIX-Standard-Jobklasse festgelegt.

Bei der herkömmlichen Festlegung von Standard-Jobklassen für BATCH oder DIALOG mit der JMU-Anweisung SET-JOB-CLASS-DEFAULT wird eine Jobklasse je nach ihrer Definition (JOB-TYPE=) zur Standard-Jobklasse entweder für Batch- oder für Dialog-Tasks. Bei POSIX-Standard-Jobklassen wird diese Unterscheidung nicht gemacht; der Typ der Jobklasse ist beliebig. Eine durch *fork* erzeugte POSIX-Task erhält den Task-Typ der Vater-Task und die Kategorie der Jobklasse, in der sie läuft, d.h. der POSIX-Standard-Jobklasse.

Wenn beim *fork* die Vater-Task nicht in einer Standard-Jobklasse (POSIX oder DIALOG bzw. BATCH) läuft und die Sohn-Task dieselbe Benutzerkennung wie die Vater-Task hat, wird der Sohn-Task nicht die POSIX-Standard-Jobklasse zugewiesen, sondern die Jobklasse der Vater-Task.

Die POSIX-Standard-Jobklassen gelten nur für durch *fork* erzeugte POSIX-Tasks. POSIX-Programme, die in Batch- oder Dialog-Tasks z.B. mit START-PROGRAM gestartet werden, laufen in der Jobklasse dieser Batch- oder Dialogtask. Das gilt insbesondere auch für den POSIX-Zugang mit dem Kommando START-POSIX-SHELL.

9 BS2000-Kommandos für POSIX

In diesem Kapitel sind alle speziellen BS2000-Kommandos für POSIX aufgeführt und die Jobvariable $.SYS.POSIXSTATUS beschrieben.

Die im Folgenden verwendete Kommandosyntax und die Beschreibung der Returncodes entspricht der üblichen Darstellung für SDF-Kommandos, siehe Handbuch „Kommandos“ [28].

ADD-POSIX-USER
POSIX-Attribute für Benutzerkennung festlegen

Anwendungsbereich: SYSTEM-MANAGEMENT

Privilegierung: TSOS,
USER-ADMINISTRATION

Dieses Kommando legt die Eigenschaften fest, die eine Benutzerkennung für die Arbeit mit POSIX haben soll.

Format

ADD-POSIX-USER
USER-NAME = <name 1..8>
,USER-NUMBER = ***DEFAULT** / <integer 0..60002>
,GROUP-**NUM**BER = ***DEFAULT** / <integer 0..60002>
,**PROG**RAM = ***DEFAULT** / <posix-pathname 1..1023 without-wild>
,**HOME-DIR**ECTORY = ***DEFAULT** / <posix-pathname 1..1023 without-wild>
,**RLOGIN-ACCOUNT** = ***NONE** / <alphanum-name 1..8>

Operandenbeschreibung

USER-NAME = <name 1..8>
BS2000-Benutzerkennung, deren POSIX-Benutzerattribute festgelegt werden sollen.

USER-NUMBER =
Benutzernummer, die für die Benutzerkennung festgelegt werden soll.

USER-NUMBER = *DEFAULT
Die Benutzernummer erhält den aktuell eingestellten Standardwert (siehe Kommando SHOW-POSIX-USER-DEFAULTS auf Seite 256).

USER-NUMBER = <integer 0..60002>
Die Benutzernummer erhält den angegebenen Wert.

GROUP-NUMBER =
Gruppennummer, die für die BS2000-Benutzerkennung festgelegt werden soll.

GROUP-NUMBER = *DEFAULT
Die Gruppennummer erhält den aktuell eingestellten Standardwert (siehe Kommando SHOW-POSIX-USER-DEFAULTS auf Seite 256).

GROUP-NUMBER = <integer 1..60002>
Die Gruppennummer erhält den angegebenen Wert.

PROGRAM =
Programm, das nach dem Kommando *rlogin* bzw. nach dem Aufruf des Kommandos START-POSIX-SHELL gestartet wird.

PROGRAM = *DEFAULT
Das zu startende Programm wird aus den aktuellen Standardattributen ermittelt (siehe Kommando SHOW-POSIX-USER-DEFAULTS auf Seite 256).

PROGRAM = <posix-pathname 1..1023 without-wild>
Das angegebene Programm wird gestartet.

HOME-DIRECTORY =
Bestimmt den absoluten Pfadnamen des Dateiverzeichnisses, in das der Benutzer automatisch gelangt, wenn er mit POSIX verbunden wird.

Falls das Verzeichnis noch nicht existiert, wird es erzeugt und der Eigentümer auf die Benutzer- und Gruppennummer der POSIX-Benutzerkennung gesetzt.

Wenn das Verzeichnis bereits existiert, bleiben dessen Attribute unverändert und es wird eine entsprechende Meldung ausgegeben (POS2907 THE HOME DIRECTORY EXISTS; ITS ATTRIBUTES ARE NOT CHANGED).

HOME-DIRECTORY = *DEFAULT
Das Verzeichnis wird aus den aktuellen Standardattributen ermittelt (siehe Kommando SHOW-POSIX-USER-DEFAULTS auf Seite 256).

HOME-DIRECTORY = <posix-pathname 1..1023 without-wild>
Das angegebene Verzeichnis wird festgelegt.

RLOGIN-ACCOUNT =
Abrechnungsnummer für die POSIX-Nutzung über Remote Login und über NFS.

RLOGIN-ACCOUNT = *NONE
Die mit ADD-USER (siehe Seite 198) bzw. MODIFY-USER-ATTRIBUTES (siehe Seite 237) festgelegte Abrechnungsnummer für den Zugang über Remote Login bleibt unverändert.

RLOGIN-ACCOUNT = <alphanum-name 1..8>
Die angegebene Abrechnungsnummer wird für den Zugang über Remote Login verwendet.

Die Abrechnungsnummer ist für Benutzerkennungen relevant, die einen Remote-Zugang zu POSIX wünschen (*rlogin*-, *ssh*- oder Telnet-Zugang, Kommandos *rsh* und *rcp*) oder die *at*, *crontab* bzw. *batch* nutzen wollen.

ADD-USER
Benutzereintrag im Benutzerkatalog erstellen

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: USER-ADMINISTRATION,
STD-PROCESSING

Dieses Kommando erstellt einen Eintrag im Benutzerkatalog eines Pubsets.

Die POSIX-Benutzerattribute werden implizit mit Standardwerten initialisiert. Darüber hinaus muss der BS2000-Systemverwalter für die Abrechnung eines Remote-Login-Systemlaufs eine Abrechnungsnummer bestimmen.

Folgende Benutzer dürfen dieses Kommando ausführen:

- Inhaber des Privilegs USER-ADMINISTRATION für alle BS2000-Benutzerkennungen.
- Gruppenverwalter, die mindestens das Attribut MANAGE-MEMBERS besitzen, für die BS2000-Benutzerkennungen, die ihren Gruppen zu- und untergeordnet sind.

Die folgende Syntaxdarstellung zeigt nur den POSIX-relevanten Kommandoteil . Das vollständige Kommando ist in den Handbüchern „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9] und „Kommandos“ [28] beschrieben.

Format

ADD-USER

```
...
,ACCOUNT-ATTRIBUTES = *PARAMETERS(...)
  *PARAMETERS(...)
    | ACCOUNT = <alphanum-name 1..8>
    | ...
    | ,POSIX-RLOGIN-DEFAULT = *NO / *YES
```

Operandenbeschreibung

ACCOUNT-ATTRIBUTES = *PARAMETERS(...)
Bestimmt die Abrechnungsdaten einer BS2000-Benutzerkennung.

. . .

POSIX-RLOGIN-DEFAULT =
Bestimmt, ob die angegebene Abrechnungsnummer zur Abrechnung eines Remote-Login-Systemlaufs herangezogen wird.

POSIX-RLOGIN-DEFAULT = *NO
Die Abrechnungsnummer wird nicht zur Abrechnung herangezogen.

POSIX-RLOGIN-DEFAULT = *YES
Die angegebene Abrechnungsnummer wird zur Abrechnung herangezogen.

Diese Angabe ist für Benutzerkennungen erforderlich, die einen Remote-Zugang zu POSIX wünschen (*rlogin*- oder Telnet-Zugang, Kommandos *rsh* und *rcp*) oder die *at*, *crontab* bzw. *batch* nutzen wollen.

Beispiel

Die Benutzerkennung PSXROOT wird mit der Abrechnungsnummer PSXACC eingerichtet.

```
/ADD-USER USER-ID=PSXROOT, -
/         ACCOUNT-ATTR=*PAR(ACCOUNT=PSXACC,POSIX-RLOGIN-DEFAULT=*YES)
```

COPY-POSIX-FILE
Dateien aus BS2000 ins POSIX-Dateisystem kopieren (und umgekehrt)

Anwendungsbereich: FILE

Privilegierung: alle Privilegien

Das BS2000-Kommando COPY-POSIX-FILE hat die Funktionalität des Shell-Kommandos *bs2cp* mit einigen Erweiterungen. Software-Voraussetzung für den Ablauf dieses Kommandos ist eine installierte Version von SDF-P-BASYS oder SDF-P größer oder gleich V2.2.

Hinweise zur technischen Realisierung: Die Parameter des Kommandos COPY-POSIX-FILE werden auf die Parameter des Shell-Kommandos *bs2cp* abgebildet. Anschließend wird nach Zugang in die POSIX-Shell (mit /START-SHELL) das Kommando *bs2cp* aufgerufen. Dies hat u.a. zur Folge, dass der aufrufende Benutzer für die Nutzung des Kommandos COPY-POSIX-FILE über ein HOME-Verzeichnis in POSIX verfügen muss, und dass sich die Einstellungen in der Datei *.profile* dieses HOME-Verzeichnisses auf den Kopiervorgang auswirken (siehe „Steuerung des Kopiervorgangs über die Datei .profile“ auf Seite 203).

Das Kommando ermöglicht es, Dateien oder PLAM-Bibliotheks-Elemente aus BS2000 in das POSIX-Dateisystem (als einfache Dateien, nicht als Elemente von ar-Bibliotheken) sowie POSIX-Dateien ins BS2000 zu kopieren. Die Ursprungsdateien können dabei explizit oder mit Wildcard-Syntax angegeben werden. Bei POSIX-Dateien wird die POSIX-Wildcard-Syntax unterstützt, bei BS2000-Dateien eine eingeschränkte BS2000-Wildcard-Syntax (nur „*“). Bei PLAM-Elementen wird die PLAM-Wildcard-Syntax unterstützt.

Wird nur eine einzelne Datei kopiert, so kann der Name der Zieldatei explizit angegeben werden. Werden dagegen mehrere Dateien kopiert, so werden die Namen der Zieldateien explizit aus den Namen der Ursprungsdateien abgeleitet (siehe Parameter *BY-SOURCE). Sind mehrere Ursprungsdateien angegeben, so heißen die Zieldateien wie die Ursprungsdateien, wobei die Zieldateien mit jeweils einem Suffix und Präfix versehen werden können. Hier ist Vorsicht beim Kopieren von POSIX nach BS2000 geboten, wenn die POSIX-Dateien exotische, aber in POSIX erlaubte Namen haben, die im BS2000 nicht unterstützt werden.

Das versehentliche Überschreiben existierender BS2000-Dateien oder PLAM-Bibliotheks-Elemente kann mit Hilfe des Parameters WRITE-MODE vermieden werden.

Die Konvertierung von Datei-Inhalten kann über den Parameter CHARACTER-CONVERSION gesteuert werden (analog zu den Optionen *-k* und *-t* des Shell-Kommandos *bs2cp*).

Die Steuerung der Meldungsausgabe ist ebenfalls möglich.

Mit dem Shell-Kommando *bs2cp* sind noch zwei weitere Kommandos eng gekoppelt: *bs2file* und *ftyp*.

Mit *bs2file* wird aus der Shell ein FILE-Makro im BS2000 abgesetzt, das eine Datei im BS2000 mit gewünschten Eigenschaften erzeugt. Hier ist der Parameter FILE-ATTRIBUTES eingeführt worden, dem die Parameter des Shell-Kommandos *bs2file* mitzugeben sind.

Das Kommando *ftyp* wird dazu benutzt, um beim Kopieren von Text- bzw. Binärdateien von BS2000 nach POSIX (und umgekehrt) mitzuteilen, ob Dateien als Text- oder als Binärdateien behandelt werden sollen. Hierzu wird der Parameter RECORD-CONVERSION eingeführt, der die entsprechenden Werte des Shell-Kommandos *ftyp* annehmen kann (siehe Beschreibung im POSIX-Handbuch „Kommandos“ [1]). Der Parameter ist nur für den aktuellen Aufruf des SDF-Kommandos gültig.

Aliasnamen sind BS2CP und CPXF.

Unterstützung des Kommandos bs2file durch den Parameter FILE-ATTRIBUTES

Vor dem Aufruf von *bs2cp* in der Shell kann das Kommando *bs2file* abgesetzt werden. Damit kann der Typ der BS2000-Datei bestimmt werden, indem mit dem in *bs2file* angegebenen String beim Aufruf von *bs2cp* implizit ein FILE-Kommando abgesetzt wird. Damit dies geschehen kann, werden die Parameterwerte in die Datei *.bs2cp* geschrieben. Ob und mit welchen Parametern das Kommando *bs2file* vor dem Kommando *bs2cp* abgesetzt wird, wird über den Parameter FILE-ATTRIBUTES gesteuert.

Dieser Parameter ist nur beim Kopieren nach BS2000 (*FROM-POSIX) anzugeben, und auch nur dann, wenn Plain Files, also keine PLAM-Bibliotheken, bearbeitet werden.

Behandlung von Umlenkungen

Wird vor dem Aufruf von COPY-POSIX-FILE eine Umlenkung von SYSOUT in eine Datei vorgenommen, so sind die folgenden Punkte zu beachten, um eine sinnvolle Abarbeitung zu gewährleisten:

- Läuft das Kommando im Dialog ab, so sollte die Umlenkung von SYSOUT in eine Datei mit den folgenden Parametern des ASSIGN-SYSOUT-Kommandos durchgeführt werden:

  ```
  /ASSIGN-SYSOUT TO = <datei>, TERMINAL-DISPLAY = *YES
  ```

  Dadurch werden alle in die Datei geschriebenen Meldungen zusätzlich am Terminal ausgegeben. Das ist insbesondere wichtig für Meldungen, die eine Antwort erwarten.

- Wird stattdessen der Standardwert

  ```
  TERMINAL-DISPLAY = *NO
  ```

  verwendet, so erscheint keine Frage am Bildschirm, wohl aber ein Stern (*), der eine Eingabe erwartet. Das sollte aus verständlichen Gründen vermieden werden.

  Der Parameter WRITE-MODE, der festlegt, ob die Datei neu geschrieben wird oder ob die neuen Daten am Ende der Datei angefügt werden, ist davon nicht betroffen. Um den beabsichtigten Effekt zu erreichen, muss er entsprechend gesetzt werden.

Bei einem Aufruf des Kommandos in einer Batch-Prozedur sind diese Vorkehrungen nicht notwendig.

EXIT-Wert bei nicht korrekter Abarbeitung von COPY-POSIX-FILE

Die Abarbeitung des Kommandos COPY-POSIX-FILE geschieht in drei Schritten:

1. Zuerst wird die Syntaxprüfung von SDF aktiviert. Diese erkennt Verstöße gegen die Syntaxregeln. Es erfolgt eine Fehlermeldung von SDF und die Kommandobearbeitung wird abgebrochen.
2. Danach wird eine SDF-Prozedur mit den syntaktisch korrekten Parametern aufgerufen. Hier findet dann eine semantische Prüfung statt, an deren Ende (im Erfolgsfall) die Generierung des Shell-Kommandos *bs2cp* mit den aktuell angegebenen Parametern steht. Im Fehlerfall (die Syntax erlaubt es explizit, zwei POSIX-Dateien anzugeben, die auf drei explizit angegebene BS2000-Dateien kopiert werden sollen) wird die Prozedur mit einem Fehlercode beendet. Dies gilt auch für den Fall, dass nicht die passende Version von SDF geladen ist, oder wenn *bs2cp* nicht gestartet werden kann (siehe Fehlermeldungen POS6010 bis POS6019).
3. Wenn das Shell-Kommando *bs2cp* ... ausgeführt wird, so kann während der Abarbeitung von COPY-POSIX-FILE trotzdem noch ein Fehler auftreten. In diesem Fall wird das Shell-Kommando *bs2cp* mit einem Exit-Wert ungleich Null beendet, der in der Meldung POS6020 ausgegeben wird. Wenn beim Aufruf von *bs2cp* mehrere Dateien kopiert werden und beim Kopieren einer Datei ein Fehler auftritt (Exit-Wert ungleich Null), so werden die noch verbleibenden Dateien nicht mehr kopiert.

Ein eventuell abgesetztes *bs2file*-Kommando wird wie folgt behandelt:

- Falls ein *bs2file*-Kommando abgesetzt wird, so werden vorher die angegebenen Parameter auf Korrektheit überprüft, indem ein FILE-Makro mit *DUMMY abgesetzt wird. Im Fehlerfall wird eine Meldung ausgegeben und die Verarbeitung beendet.
- Falls in gewissen Fällen vor dem Aufruf von *bs2cp* ein *bs2file*-Kommando abgesetzt wird, so wird dort ebenso dessen EXIT-Status gesichert und später ausgegeben.

Steuerung des Kopiervorgangs über die Datei .profile

Da bei der Verwendung des SDF-Kommandos COPY-POSIX-FILE das eigentliche Kopieren in der Shell mit *bs2cp* erfolgt, wirken sich die Einstellungen in der Datei *.profile* des aufrufenden Benutzers auf den Kopiervorgang aus. Beispiele für relevante Einstellungen sind:

- Wechsel des aktuellen Dateiverzeichnisses (Kommando *cd*). Standardmäßig ist das aktuelle Verzeichnis das Home-Verzeichnis des aufrufenden Benutzers.
- Definieren der *bs2cp*-relevanten Umgebungsvariablen *IO_CONVERSION* und *BS2CPTABS*.

Format

COPY-POSIX-FILE

COPY-DIRECTION = ***FROM-POSIX** / ***TO-POSIX**

,**POSIX-FILE** = ***BY-SOURCE**(...) / <list-poss(2000): posix-pathname 1..1023>

***BY-SOURCE**(...)
- **POSIX-DIR**ECTORY = **.** / <posix-pathname 1..1023 without-wild>
- ,**PREFIX** = ***NONE** / <c-string 0..80 with-low>
- ,**SUFFIX** = ***NONE** / <c-string 0..80 with-low>

,**BS2000-FILE** = ***BY-SOURCE**(...) / ***LIB**RARY-**ELEM**ENT(...) / list-poss(2000): <filename 1..80 with-wild>

***BY-SOURCE**(...)
- **PREFIX** = ***NONE** / <c-string 0..53>
- ,**SUFFIX** = ***NONE** / <c-string 0..40>

***LIB**RARY-**ELE**MENT(...)
- **LIB**RARY = <filename 1..54>
- ,**ELEM**ENT = ***BY-SOURCE**(…)/ list-poss(2000): <composed-name 1..64 with-under-wild>(...)
  - ***BY-SOURCE**(…)
    - **VERSION** = ***HIGH**EST-**EXIST**ING / ***UPPER-LIMIT** / <composed-name 1..24 with-under>
    - **,PREFIX** = ***NONE** / <c-string 0..63>
    - **,SUFFIX** = ***NONE** / <c-string 0..63>
  - <composed-name 1..64 with-under-wild>(...)
    - **VERSION** = ***HIGH**EST-**EXIST**ING / ***UPPER-LIMIT** / <composed-name 1..24 with-under>
- ,**TYPE** = ***S** / ***D** / ***J** / ***M** / ***P** / ***X** / ***L**

,**WRITE-MODE** = ***BY-DIAL**OG / ***REPLACE** / ***CREATE**

,**CHAR**ACTER-**CONV**ERSION = ***NO** / ***YES**(…)

***YES**(...)
- **TABLE** = ***STD** / <posix-pathname 1..1023 without-wild>

,**OUTPUT** = ***NONE** / ***SYSOUT**

,**RECORD-CONV**ERSION = ***TEXT**(...) / ***BINARY**

***TEXT**(...)
- **SUBST**ITUTE-**TAB**ULATOR = ***YES** / ***NO**

,**FILE-ATTR**IBUTES = ***STD** / ***PAR**AMETER(...)

***PAR**AMETER(...)
- **FILE-NAME** = ***ALL** / <filename 1..54>
- ,**ATTR**IBUTES = ***STD** / <c-string 0..1000>

Operandenbeschreibung

COPY-DIRECTION =
Richtung des Kopiervorgangs.

COPY-DIRECTION = *FROM-POSIX
POSIX-Dateien werden ins BS2000 kopiert.

COPY-DIRECTION = *TO-POSIX
BS2000-Dateien oder PLAM-Elemente werden nach POSIX kopiert.

POSIX-FILE =
Angabe der POSIX-Dateien, die beim Kopieren benutzt werden.

POSIX-FILE = *BY-SOURCE
Die Namen der POSIX-Dateien werden aus den BS2000-Namen abgeleitet.

Pflichtangabe bei der Kopierrichtung *TO-POSIX, wenn mehr als eine BS2000-Datei kopiert werden soll.

Alternativ mögliche Angabe zu *posix-pathname* bei Kopierrichtung *TO-POSIX, wenn nur eine BS2000-Datei kopiert wird und der Name der Zieldatei aus dem Namen der BS2000-Datei abgeleitet werden soll.

POSIX-DIRECTORY =
Angabe des Verzeichnisses, in das die BS2000-Dateien bzw. PLAM-Elemente kopiert werden.

POSIX-DIRECTORY = .
Es wird das aktuell eingestellte Verzeichnis benutzt.

Dies ist standardmäßig das Home-Verzeichnis des aufrufenden BS2000-Benutzers. Ein anderes aktuelles Verzeichnis lässt sich durch einen Verzeichnis-Wechsel (*cd*) in der Datei *.profile* einstellen.

POSIX-DIRECTORY = <posix-pathname 1..1023 without-wild>
Es wird das explizit angegebene Verzeichnis benutzt.

PREFIX =
Angabe des Präfixes, das den POSIX-Dateinamen vorangestellt wird.

PREFIX = *NONE
Es wird kein Präfix benutzt.

PREFIX = <c-string 0..80 with-low>
Der angegebene String wird als Präfix benutzt.

SUFFIX =
Angabe des Suffixes, das an die POSIX-Dateinamen angehängt wird.

SUFFIX = *NONE
Es wird kein Suffix benutzt.

SUFFIX = <c-string 0..80 with-low>
Der angegebene String wird als Suffix benutzt.

POSIX-FILE = list-poss(2000): <posix-pathname 1..1023>
Die Namen der POSIX-Dateien werden explizit angegeben. Folgendes ist zu beachten:

– Bei Kopierrichtung *FROM-POSIX: Einer oder mehrere absolute oder relative Pfadnamen von POSIX-Dateien. Für die Angabe des POSIX-Dateinamens wird die Wildcard-Syntax (Shell-Sonderzeichen für Dateinamensersatz) unterstützt.

– Bei Kopierrichtung *TO-POSIX, wenn nur eine BS2000-Datei kopiert wird und der Name der Zieldatei explizit vereinbart werden soll: Absoluter oder relativer Pfadname einer POSIX-Datei. Wildcard-Syntax ist nicht erlaubt.

– Relative Pfadnamen beziehen sich standardmäßig auf das Home-Verzeichnis des aufrufenden BS2000-Benutzers. Ein anderes Verzeichnis lässt sich durch einen Verzeichnis-Wechsel (*cd*) in der Datei *.profile* einstellen.

BS2000-FILE =
Angabe der BS2000-Dateien oder PLAM-Elemente, die beim Kopieren benutzt werden.

BS2000-FILE = *BY-SOURCE(...)
Die Namen der BS2000-Dateien werden aus den Namen der POSIX-Dateien abgeleitet.

Pflichtangabe bei der Kopierrichtung *FROM-POSIX, wenn mehr als eine POSIX-Datei kopiert werden soll.

Alternativ mögliche Angabe zu *filename* bei Kopierrichtung *FROM-POSIX, wenn nur eine POSIX-Datei kopiert wird und der Name der Zieldatei aus dem Namen der POSIX-Datei abgeleitet werden soll.

PREFIX =
Angabe des Präfixes, das den BS2000-Dateinamen vorangestellt wird.

PREFIX = *NONE
Es wird kein Präfix benutzt.

PREFIX = <c-string 0..53 with-low>
Der angegebene String wird als Präfix benutzt.

SUFFIX =
Angabe des Suffixes, das an die BS2000-Dateinamen angehängt wird.

SUFFIX = *NONE
Es wird kein Suffix benutzt.

SUFFIX = <c-string 0..40 with-low>
Der angegebene String wird als Suffix benutzt.

BS2000-FILE = *LIBRARY-ELEMENT(...)
Statt BS2000-Dateien werden PLAM-Elemente beim Kopieren benutzt.

LIBRARY = <filename 1..54>
Explizite Angabe der PLAM-Bibliothek, die beim Kopieren benutzt wird.

ELEMENT =
Angabe der PLAM-Elemente, die beim Kopieren benutzt werden.

ELEMENT = *BY-SOURCE(...)
Die Namen der Elemente werden aus den POSIX-Dateien abgeleitet.

Pflichtangabe bei Kopierrichtung *FROM-POSIX, wenn mehr als eine POSIX-Datei kopiert werden soll.

Alternativ mögliche Angabe zu *composed-name* bei Kopierrichtung *FROM-POSIX, wenn nur eine POSIX-Datei kopiert wird und der Name des Zielelements aus dem Namen der POSIX-Datei abgeleitet werden soll.

VERSION =
Angabe, welche Version eines Elements benutzt wird.

VERSION = *HIGHEST-EXISTING
Das Element mit der höchsten Version wird benutzt. Folgendes ist zu beachten:

– Wenn ein Element noch nicht existiert, erhält es die Version *001*.

– Wenn bereits existierende Elemente kopiert werden, wird das Element mit der höchsten Version überschrieben.

VERSION = *UPPER-LIMIT
Das kopierte Element soll die höchstmöglichste Version erhalten (X'FF', entspricht der Tilde im Kommando *bs2cp*).

VERSION = <composed-name 1..24 with-under>
Die Version wird explizit angegeben.

PREFIX =
Angabe des Präfixes, das den PLAM-Elementnamen vorangestellt wird.

PREFIX = *NONE
Es wird kein Präfix benutzt.

PREFIX = <c-string 0..63 with-low>
Der angegebene String wird als Präfix benutzt.

SUFFIX =
Angabe des Suffixes, das an die PLAM-Elementnamen angehängt wird.

SUFFIX = *NONE
Es wird kein Suffix benutzt.

SUFFIX = <c-string 0..63 with-low>
Der angegebene String wird als Suffix benutzt.

ELEMENT = list-poss (2000): <composed-name 1..64 with-under-wild>(...)
Die Namen der Elemente werden explizit angegeben. Folgendes ist zu beachten:

– Bei Kopierrichtung *TO-POSIX: Einer oder mehrere Elementnamen. Für die Angabe des Elementnamens wird die LMS-Wildcard-Syntax („*“, „<„, „:“, „>“) unterstützt. Die Angabe einer Liste von Elementnamen (list-poss) ist eine Erweiterung gegenüber dem Kommando *bs2cp*, mit dem die Angabe nur eines PLAM-Element-Operanden (**bs2:**) möglich ist. Bei Angabe einer Element-Liste wird pro Elementname (mit/ohne Wildcard) das Kommando *bs2cp* aufgerufen. Die Angaben in den restlichen SDF-Parametern gelten dann für sämtliche *bs2cp*-Aufrufe.

– Bei Kopierrichtung *FROM-POSIX, wenn nur eine POSIX-Datei kopiert wird und der Name der Zieldatei explizit vereinbart werden soll: Expliziter Name eines Elements. Wildcard-Syntax ist nicht erlaubt.

VERSION =
Angabe, welche Version eines Elements benutzt wird.

VERSION = *HIGHEST-EXISTING
Das Element mit der höchsten Version wird benutzt.

VERSION = *UPPER-LIMIT
Das kopierte Element soll die höchstmöglichste Version erhalten (X'FF').

VERSION = <composed-name 1..24 with-under>
Die Version wird explizit angegeben.

TYPE = *S / *D / *J / *M / *P / *X / *L
Typ-Angabe der zu behandelnden PLAM-Elemente. Standardmäßig wird der Typ *S* (Source) genommen.

BS2000-FILE = list-poss (2000): <filename 1..80 with-wild>
Die Namen der BS2000-Dateien werden explizit angegeben. Folgendes ist zu beachten:

– Bei Kopierrichtung *TO-POSIX: Einer oder mehrere DVS-Dateinamen. Für die Angabe des Dateinamens wird die Wildcard-Syntax für DVS-Dateien („*“) unterstützt. Die Angabe einer Liste von DVS-Dateinamen (list-poss) ist eine Erweiterung gegenüber dem Kommando *bs2cp*, mit dem die Angabe nur eines Dateinamen-Operanden (**bs2:**) möglich ist. Bei Angabe einer Dateinamen-Liste wird pro Dateiname (mit/ohne Wildcard) das Kommando *bs2cp* aufgerufen. Die Angaben in den restlichen SDF-Parametern gelten dann für sämtliche *bs2cp*-Aufrufe.

– Bei Kopierrichtung *FROM-POSIX, wenn nur eine POSIX-Datei kopiert wird und der Name der Zieldatei explizit vereinbart werden soll: Expliziter Name einer DVS-Datei. Wildcard-Syntax ist nicht erlaubt.

WRITE-MODE =
Angabe, ob BS2000-Dateien beim Kopiervorgang überschrieben werden. Die Angabe von WRITE-MODE = ist nur bei der Kopierrichtung *FROM-POSIX relevant und bestimmt, ob bereits existierende BS2000-Dateien (DVS-Dateien oder PLAM-Elemente) überschrieben werden (analog zur Option *-f* des Kommandos *bs2cp*).

WRITE-MODE = *BY-DIALOG
Es wird im Dialog abgefragt, ob eine bereits existierende Datei überschrieben werden soll.

WRITE-MODE = *REPLACE
Die Dialogabfrage wird unterdrückt und bereits existierende Dateien werden immer überschrieben.

WRITE-MODE = *CREATE
Die Dialogabfrage wird unterdrückt. Bereits existierende Dateien werden nicht überschrieben, aber es werden neue Dateien angelegt.

CHARACTER-CONVERSION =
Angabe, ob beim Kopiervorgang eine Konvertierung durchgeführt wird (analog zu den Optionen *-k* oder *-t* des Kommandos *bs2cp*, siehe POSIX-Handbuch „Kommandos“ [1]).

CHARACTER-CONVERSION = *NO
Es wird keine Konvertierung durchgeführt.

CHARACTER-CONVERSION = *YES(...)
Es wird eine Konvertierung durchgeführt.

TABLE =
Angabe der Konvertierungstabelle.

TABLE = *STD
Es werden POSIX-interne Standard-Tabellen benutzt (analog zur Option *-k* des Kommandos *bs2cp*, siehe POSIX-Handbuch „Kommandos“ [1]).

TABLE = <posix-pathname 1..1023 without-wild>
Die Konvertierungstabelle wird explizit angegeben (analog zur Option *-t* des Kommandos *bs2cp*, siehe POSIX-Handbuch „Kommandos“ [1]).

Hinweis

Die mögliche Versorgung der Shell-Variablen *BS2CPTABS* (siehe *bs2cp*) wird nicht über den SDF-Parameter gesteuert. Die Versorgung kann bei Bedarf in der Datei *.profile* erfolgen.

OUTPUT =
Angabe, ob die erweiterte Protokollierung von *bs2cp* ausgegeben wird (analog zur Option *-l* des Kommandos *bs2cp*, siehe POSIX-Handbuch „Kommandos“ [1]).

OUTPUT = *NONE
Die erweiterte Protokollierung wird nicht ausgegeben.

OUTPUT = *SYSOUT
Die erweiterte Protokollierung wird auf SYSOUT ausgegeben.

RECORD-CONVERSION =
Angabe, wie der Inhalt von BS2000-Dateien beim Kopieren zu behandeln ist.

Dieser Parameter generiert das Shell-Kommandos *ftyp* mit entsprechenden Parametern. Ohne explizite Angabe des Parameters RECORD-CONVERSION ist *ftyp text* voreingestellt.

RECORD-CONVERSION = *TEXT(...)
SAM-Dateien werden als Text-Dateien behandelt. „newline“ wird zu Satzwechsel.

SUBSTITUTE-TABULATOR =
Angabe, wie Tabulatorzeichen behandelt werden.

SUBSTITUTE-TABULATOR = *YES
Tabulatorzeichen werden entsprechend aufgefüllt (*ftyp text*).

SUBSTITUTE-TABULATOR = *NO
Tabulatorzeichen bleiben erhalten (*ftyp textbin*).

RECORD-CONVERSION = *BINARY
SAM-Dateien werden als Binär-Dateien behandelt.

FILE-ATTRIBUTES =
Beim Kopieren von POSIX-Dateien ins BS2000 als DVS-Dateien (nicht als PLAM-Elemente) werden die Dateiattribute analog zum Shell-Kommando *bs2file* angegeben. Je nach Parameter-Angabe wird dann in der Shell vor dem eigentlichen *bs2cp*-Kommando ein *bs2file*-Kommando abgesetzt.

FILE-ATTRIBUTES = *STD
Beim Kopieren wird kein Shell-Kommando *bs2file* abgesetzt.

Noch nicht existierende DVS-Dateien erhalten die Standardattribute FCBTYPE=SAM, RECFORM=V und BLKSIZE=STD. Werden bereits existierende DVS-Dateien überschrieben, so werden die Dateieigenschaften der überschriebenen Dateien beibehalten. Wenn zu einer DVS-Datei nur ein Katalogeintrag (ohne FCBTYPE, RECFORM, BLKSIZE) existiert, erhält die Datei die Standardattribute des Betriebssystems (FCBTYPE=ISAM).

FILE-ATTRIBUTES = *PARAMETER(...)
Beim Kopieren werden die Dateiattribute analog zum Shell-Kommando *bs2file* angegeben.

FILE-NAME =
Angabe der Dateien, für die die Attribute gesetzt werden.

FILE-NAME = *ALL
Die angegebenen Attribute sollen für alle zu kopierenden Dateien mit beliebigem Namen gelten (entspricht der Angabe „*“ im Kommando *bs2file*).

FILE-NAME = <filename 1..54>
Die angegebenen Attribute sollen nur für die Datei mit dem angegebenen Namen gelten.

ATTRIBUTES =
Angabe der Attribute, Format wie beim (ISP-)FILE-Kommando.

ATTRIBUTES = *STD
Standard-Attribut mit den Werten: FCBTYPE = SAM, RECFORM = V, BLKSIZE = STD.

ATTRIBUTES = <c-string 1..1000>
Explizite Angabe der Attribute.

Die unterstützten Dateiattribute sind beim Shell-Kommando *bs2cp* beschrieben (siehe POSIX-Handbuch „Kommandos“ [1], Abschnitt „Unterstützte Attribute von DVS-Dateien“).

Beispiel

```
ATTRIBUTES='FCBTYPE=SAM,RECFORM=F,BLKSIZE=80'
```

Kommando-Returncode

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Ohne Fehler
	64	POS6010	Ungültige Kombination der Angaben BS2000-FILE=*BY-SOURCE und COPY-DIRECTION=*TO-POSIX
	64	POS6011	Ungültige Kombination der Angaben BS2000-FILE=*LIBRARY-ELEMENT(...,ELEMENT=*BY-SOURCE,...) und COPY-DIRECTION=*TO-POSIX
	64	POS6012	Ungültige Kombination der Angaben POSIX-FILE=*BY-SOURCE und COPY-DIRECTION=*FROM-POSIX
	64	POS6013	Mehrere BS2000-Dateien als Ziel angegeben.
	64	POS6014	Mehrere PLAM-Bibliothekselemente als Ziel angegeben.
	64	POS6015	Mehrere POSIX-Dateien als Ziel angegeben.
	64	POS6016	Mehrere BS2000-Dateien oder PLAM-Bibliothekselemente als Quelle und eine POSIX-Datei als Ziel angegeben; diese ist aber kein Dateiverzeichnis.
	64	POS6017	Mehrere POSIX-Dateien als Quelle angegeben, aber nicht BS2000-FILE=*BY-SOURCE angegeben.
	64	POS6018	Es ist nicht die erforderliche Version von SDF bzw. SDF-P-BASYS installiert.
	64	POS6019	Fehler beim Starten der POSIX-Shell.
	64	POS6020	Fehler beim Ausführen des POSIX-Kommandos bs2cp.
	64	POS6021	Ungültige Angabe von FILE-ATTRIBUTES.
	64	POS6022	Fehler beim Ausführen des POSIX-Kommandos bs2file.
	64	POS6023	Fehlerhafte Angabe von Wildcards.
	64	POS6024	Ungültige Angaben im Operanden ATTRIBUTES.

EXECUTE-POSIX-CMD
POSIX-Kommandos aus BS2000 heraus aufrufen

Anwendungsbereich: PROCEDURE

Privilegierung: STD-PROCESSING

Mit dem BS2000-Kommando EXECUTE-POSIX-CMD wurde eine Möglichkeit geschaffen, Kommandos der POSIX-Shell aus dem BS2000 aufzurufen. Dies bedeutet, dass die Shell zur Kommando-Ausführung nicht mehr explizit aufgerufen werden muss, und man einzelne Kommandos, ganze Kommando-Sequenzen oder Shell-Skripts im BS2000 starten kann.

So kann man EXECUTE-POSIX-CMD beispielsweise dazu nutzen, vor einem Kopiervorgang mit COPY-POSIX-FILE in POSIX Verzeichnisse einzurichten oder nach einem Kopiervorgang die kopierten Dateien weiter zu bearbeiten.

Kenntnisse in der Syntax von Shell-Kommandos werden bei der Benutzung von EXECUTE-POSIX-CMD vorausgesetzt.

Die Kommandos bzw. Kommandosequenzen können entweder explizit eingegeben oder aus einer BS2000-Datei gelesen und ausgeführt werden. Bei der expliziten Eingabe mehrerer Kommandos/Kommandosequenzen werden diese als einzelne Listenelemente - getrennt durch Komma - eingegeben. Es wird allerdings keine separate Prüfung durchgeführt, jedes Listenelement wird so wie es ist an die Shell weitergereicht.
Bei der Eingabe über eine BS2000-Datei ergibt sich die Möglichkeit, Kommandos und Shell-Skripts nach vorherigem Kopieren in das BS2000 zu starten. Eine Parametrisierung ist dabei allerdings nicht möglich.

Die explizit oder implizit aufgerufenen Kommandos/Kommandosequenzen können bei der Ausführung von EXECUTE-POSIX-CMD in einer Log-Datei gespeichert werden. Da diese Datei eine BS2000-Datei ist, kann sie u. a. wiederum als Eingabedatei bei einem folgenden Aufruf von EXECUTE-POSIX-CMD genutzt werden. Soll z. B. eine umfangreiche Kommando-Sequenz mehrfach ausgeführt werden, kann man sich damit die erneute explizite Angabe ersparen.

Die Ausgaben des Kommandos werden entweder auf dem Bildschirm (SYSOUT) ausgegeben oder in eine BS2000-Datei geschrieben.

In der Shell, die vom Kommando EXECUTE-POSIX-CMD gestartet wird, ist folgende Umgebungsvariable gesetzt:

```
EXECUTE_POSIX_CMD="YES"
```

Durch Abfrage dieser Variablen, z.B. in */etc/profile* oder *.profile*, können Ausgaben unterdrückt werden, die beim Kommando EXECUTE-POSIX-CMD nicht erwünscht sind.

Format

EXECUTE-**POSIX-CMD**

CMD = <filename 1..54> / list-poss(15): <c-string 1..100 with-low>

,**INPUT-LOG-FILE** = ***NONE** / <filename 1..54 without-generation-version>(...)

<filename 1..54 without-gen-vers>(...)

WRITE-**MODE** = ***REPLACE** / ***EXTEND**

,**OUTPUT** = ***SYSOUT** / <filename 1..54>

Operandenbeschreibung

CMD=
Angabe der auszuführenden Kommandos oder Skripts.

CMD = <filename 1..54>
Die Kommandos/Kommandoseqenzen werden aus einer BS2000-Datei gelesen.

CMD = list-poss (15): <c-string 1..100 with-low>
Die Kommandos/Kommandoseqenzen werden explizit angegeben.

INPUT-LOG-FILE =
Angabe, ob eine Log-Datei geschrieben werden soll oder nicht.

INPUT-LOG-FILE =*NONE
Es wird keine Log-Datei geschrieben.

INPUT-LOG-FILE = <filename 1..54 without-generation-version>(...)
Angabe der BS2000-Datei, die als Log-Datei dienen soll.

WRITE-MODE = *REPLACE / *EXTEND
Angabe, ob die Log-Datei bei jedem Aufruf von EXECUTE-POSIX-CMD neu angelegt oder erweitert werden soll. Die Angabe von WRITE-MODE ist nur bei der Angabe einer BS2000-Datei relevant.

OUTPUT =
Angabe, wo die Ausgaben des Kommandos erfolgen sollen.

OUTPUT = *SYSOUT
Die Ausgaben des Kommandos werden auf dem Bildschirm angezeigt.

OUTPUT = <filename 1..54>
Die Ausgaben des Kommandos werden in eine BS2000-Datei geschrieben.

Einschränkungen

- Kommandos/Skripts, die mit EXECUTE-POSIX-CMD ausgeführt werden, können nicht von der Standardeingabe lesen, da diese vor der Ausführung des Kommandos/Skripts geschlossen wird. Daher erhalten solche Kommandos/Skripts EOF, wenn sie versuchen, von der Standardeingabe zu lesen.

  POSIX-Kommandos, die unter Umständen von der Standardeingabe lesen, sind:

  – rm

    Rückfrage beim Löschen schreibgeschützter Dateien, wenn die Option *-f* nicht angegeben wurde; EOF beim Lesen von *stdin* wird wie *ja* behandelt, d. h. die Datei wird gelöscht.

  – mv

    Rückfrage beim Überschreiben schreibgeschützter Dateien, wenn die Option *-f* nicht angegeben wurde; die Datei wird nicht überschrieben und ein Fehler generiert.

  – bs2cp

    Rückfrage beim Überschreiben von BS2000-Dateien, wenn die Option *-f* nicht angeben wurde; EOF beim Lesen von *stdin* wird wie *nein* behandelt, d. h. die BS2000-Datei wird nicht überschrieben.

  – mailx

    Eingaben an *mailx* sind nicht möglich, d. h. lediglich die Abfrage, ob Nachrichten vorhanden sind, ist sinnvol einsetzbar.

- Bei EXECUTE-POSIX-CMD sind *stdout* und *stderr* nicht mit einem Terminal verbunden, sondern mit einer Pipe. Kommandos/Skripts, die voraussetzen, dass *stdout* und *stderr* mit einem Terminal verbunden sind, arbeiten daher nicht oder nicht korrekt. Diese Kommandos/Skripts verwenden die CRTE-Funktionen *isatty()* bzw. *ttyname()*, um zu ermitteln, ob oder mit welchem Terminal *stdout* und *stderr* verbunden sind.

  POSIX-Kommandos, die deswegen unter Umständen nicht oder nicht korrekt funktionieren, sind:

  – tty

    Liefert *not a tty* mit Endestatus *1* anstatt */dev/term/nnnn*.

  – tabs

    Funktioniert an Blockterminals generell nicht.

– mesg, write, talk

  Diese Kommandos zum Austausch von Nachrichten zwischen Terminals funktionieren an Blockterminals nur rudimentär und unter EXECUTE-POSIX-CMD praktisch gar nicht.

– more

  Das Kommando *more* verhält sich unter EXECUTE-POSIX-CMD wie das Kommando *cat*.

– patch

  Bei Rückfragen wird eine leere Antwort generiert, was zu Endlosschleifen führen kann.

– pax

  Der interaktive Modus (Option *-i*) ist nicht möglich.

– nohup

  Das Kommando *nohup* funktioniert nicht, weil *stdout* kein Terminal ist.

– ls

  Das Kommando *ls* gibt die Dateien nur in mehreren Spalten aus, wenn dies explizit durch *ls -C* gefordert wird.

– fg

  Liefert *No Job Control*.

– bg

  Liefert *No Job Control*.

- Wird das Shell-Kommando *exec* mit EXECUTE-POSIX-CMD ausgeführt, so wird die aktuelle Shell entladen, und die Mechanismen zum Weiterleiten von Ausgaben und/oder des Exit-Wertes von geforkten Prozessen können möglicherweise außer Kraft gesetzt werden.

  Beispiel

```
/begin-block
%BEGIN-BLOCK/ecxcmd ('exec who am i')
%BEGIN-BLOCK/if-cmd-error
%IF-CMD-ERROR/wrtx 'cmd 1 failed'
%IF-CMD-ERROR/else
%ELSE/wrtx 'cmd 1 ok'
%ELSE/end-if
%BEGIN-BLOCK/ecxcmd ('exec who ar u')
%BEGIN-BLOCK/if-cmd-error
%IF-CMD-ERROR/wrtx 'cmd 2 failed'
%IF-CMD-ERROR/else
%ELSE/wrtx 'cmd 2 ok'
%ELSE/end-if
%BEGIN-BLOCK/end-block
FROEDE     sf/002       Nov 12 09:07
cmd 1 ok
who: Syntax:
    who [-mu] -s [-bHlprt] [datei]
    who [-mTu] [-abdHlprt] [datei]
    who -q [-n #] [datei]
    who [am i|am I]
cmd 2 ok
/
```

- Das Kommando *fc* wirkt nur auf Eingaben außerhalb von Skripts, es ist deshalb unter /EXECUTE-POSIX-CMD ungeeignet.
- Die mit EXECUTE-POSIX-CMD ausgeführten Shell-Kommandos werden nicht in das übliche Kommandogedächtnis (*$HOME/.sh_history*) protokolliert, sondern in ein separates relativ kurzes (*HISTSIZE=100*) Kommandogedächtnis unter */tmp/.ecxcmd_sh_history_<user-name>*.
- Kommandosubstitutionen durch *'kommando'* oder *$(kommando)* werden unter EXECUTE-POSIX-CMD generell in einer Sub-Shell ausgeführt. In der POSIX-Shell dagegen gibt es eine Reihe von Kommandos, die innerhalb der Shell selbst substituiert werden.

  Das hat zur Folge, dass einzelne Kommandos sich anders verhalten können als in der POSIX-Shell, sofern die Ergebnisse prozess-spezifisch sind. Bekannte Fälle dazu sind *ftyp* und *bs2file* sowie Zugriffe auf nicht exportierte Variablen oder Funktionen.

- Alias-Namen:
  Die von EXECUTE-POSIX-CMD auszuführende Kommandosequenz wird in der POSIX-Shell mit dem Punkt-Kommando ausgeführt. Deshalb steht das Kommando *alias* zum Definieren von Alias-Namen zwar zur Verfügung, hat aber innerhalb der Kommandosequenz keine Wirkung bei der Kommandoausführung. Sollen in einer Kommandosequenz Alias-Namen definiert und verwendet werden, muss die Kommandosequenz in eine (temporäre) POSIX-Datei kopiert werden. Diese muss das Execute-Recht bekommen und ausgeführt werden (nicht mit Punkt-Kommando).

  Folgende Wege sind sinnvoll:

  – Die Kommandosequenz wird als BS2000-Datei erstellt und mit COPY-POSIX-FILE in eine temporäre POSIX-Datei kopiert. Danach wird sie ausgeführt.

    Beispiel

    ```
    /EXEC-POSIX-CMD CMD=('chmod +x scriptfile', 'scriptfile', 'rm -f
    scriptfile')
    ```

  – Die Kommandosequenz generiert selbst eine temporäre Skript-Datei in POSIX aus einem so genannten Here-Dokument und führt sie aus.

    Beispiel

    ```
    cat <<***END_OF_SCRIPT >/tmp/my_scriptfile_$$
    kommando1
    kommando2
    ...
    ***END_OF_SCRIPT
    /tmp/my_scriptfile_$$
    rm -f /tmp/my_scriptfile_$$
    ```

- EXECUTE-POSIX-CMD beendet sich erst, wenn alle aus der Kommandosequenz gestarteten Hintergrundprozesse beendet sind. Auch mit dem Kommando *nohup* kann keine asynchrone Verarbeitung erzwungen werden.

- Die mit EXECUTE-POSIX-CMD aufgerufene Kommando-Sequenz wird in einer Sub-Shell ausgeführt, die intern per *fork* erzeugt wird. In dieser Sub-Shell steht die SYSFILE-Umgebung der aufrufenden Prozedur nicht zur Verfügung. Das kann Auswirkungen haben auf BS2000-Kommandos, die mit *bs2cmd* aufgerufen werden, sowie auf die POSIX-Kommandos *lp*, *lpstat* und *cancel*.

  Beispiel

```
/begin-block
/  ecxcmd 'bs2cmd sh-sys-file-ass \*syscmd'
/end-block
PROCEDURE LEVEL NUMBER 0
SYSCMD  : (PRIMARY)
/
```

  vgl. aber:

```
/begin-block
/  start-posix-shell
/end-block
POSIX Basisshell ...
*bs2cmd sh-sys-file-ass \*syscmd
PROCEDURE LEVEL NUMBER 1
SYSCMD  : *PRIMARY (DIALOG-BLOCK)
*
```

- Die Taste *K2* muss zweimal gedrückt werden, um EXECUTE-POSIX-CMD abzubrechen.

Kommando-Returncode

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Ohne Fehler
x	64	CCM0999	Das Shell-Kommando, die Kommandosequenz bzw. das Skript liefert einen Exit-Status mit dem Wert x (≠0), der dem SC2 entnommen werden kann.

MODIFY-LOGON-PROTECTION
Schutzattribute ändern

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: STD-PROCESSING, USER-ADMINISTRATION

Ändert bereits bestehende Schutzattribute für Benutzerkennungen.

Berechtigt zur Ausführung des Kommandos sind:

– Systemglobale Benutzerverwalter (Inhaber des Privilegs USER-ADMINISTRATION) für alle Benutzerkennungen

– Gruppenverwalter, die mindestens das Attribut MANAGE-MEMBERS besitzen, für die ihrer Benutzergruppe zu- und untergeordneten Benutzerkennungen.

Nicht angegebene Operanden bleiben unverändert (Standardwert *UNCHANGED oder *NONE).

Das Kommando MODIFY-LOGON-PROTECTION ist das gegebene Mittel, um Benutzerkennungen zu reaktivieren, die wegen Überschreitung ihres Verfallsdatums oder wegen zu lange nicht geändertem Kennwort vom System gesperrt wurden. Im ersten Fall muss ein neues, in der Zukunft liegendes Verfallsdatum (EXPIRATION-DATE) eingetragen, im zweiten ein neues Kennwort vereinbart werden.

Die folgende Syntaxdarstellung zeigt nur den POSIX-relevanten Teil des Kommandos. Zusätzlich kann noch der Operand BATCH-ACCESS von Bedeutung sein (z.B. für at, batch, crontab).

Das vollständige Kommando ist im Handbuch „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9] beschrieben.

Format

(Teil 1 von 2)

MODIFY-**LOGON-PROT**ECTION

```
. . .
,POSIX-RLOGIN-ACCESS = *UNCHANGED / *YES(...) / *NO
    *YES(...)
        PASSWORD-CHECK = *UNCHANGED / *YES / *NO
        ,TERMINAL-SET = *UNCHANGED / *NO-PROTECTION / *NONE /
                        *EXCEPTION-LIST(…) / *MODIFY-LIST(…) /
                        list-poss(48): <name 1..8>(…)
            *EXCEPTION-LIST (…)
                TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
                    <name 1..8>(…)
                        SCOPE = *STD / *USER / *GROUP / *SYSTEM
            *MODIFY-LIST(…)
                REMOVE-TERMINAL-SETS = *NONE / *ALL / list-poss(48): <name 1..8>(…)
                    <name 1..8>(…)
                        SCOPE = *STD / *USER / *GROUP / *SYSTEM
                ,ADD-TERMINAL-SETS = *NONE / *ALL / list-poss(48): <name 1..8>(…)
                    <name 1..8>(…)
                        SCOPE = *STD / *USER / *GROUP / *SYSTEM
            <name 1..8> (…)
                SCOPE = *STD / *USER / *GROUP / *SYSTEM
        ,GUARD-NAME = *UNCHANGED / *NONE / <filename 1..18 without-cat-gen-vers>
```

Fortsetzung ➠

(Teil 2 von 2)

```
,POSIX-REMOTE-ACCESS = *UNCHANGED / *YES(...) / *NO
    YES(...)
        TERMINAL-SET = *UNCHANGED / *NO-PROTECTION / *NONE /
                       *EXCEPTION-LIST(...) / *MODIFY-LIST(...)
                       list-poss(48): <name 1..8> (...)
          *EXCEPTION-LIST (...)
              TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(...)
                  <name 1..8>(...)
                      SCOPE = *STD / *USER / *GROUP / *SYSTEM
          *MODIFY-LIST(...)
              REMOVE-TERMINAL-SETS = *NONE / *ALL / list-poss(48): <name 1..8>(...)
                  <name 1..8>(...)
                      SCOPE = *STD / *USER / *GROUP / *SYSTEM
              ,ADD-TERMINAL-SETS = *NONE / *ALL / list-poss(48): <name 1..8>(...)
                  <name 1..8>(...)
                      SCOPE = *STD / *USER / *GROUP / *SYSTEM
          <name 1..8>(...)
              SCOPE = *STD / *USER / *GROUP / *SYSTEM
        ,GUARD-NAME = *UNCHANGED / *NONE / <filename 1..18 without-cat-gen-vers>
```

Operandenbeschreibung

POSIX-RLOGIN-ACCESS = *UNCHANGED / *YES(...) / *NO
Die Zugangsklassen-Attribute für POSIX-Remote-Login können festgelegt werden.

POSIX-RLOGIN-ACCESS = *YES(...)
Die BS2000-Benutzerkennung ist für den Systemzugang über POSIX-Remote-Login offen.

PASSWORD-CHECK = *UNCHANGED / *YES / *NO
Legt fest, ob beim Zugang über POSIX-Remote-Login eine Kennwortprüfung stattfindet.

TERMINAL-SET = *UNCHANGED / *NO-PROTECTION / *NONE / *EXCEPTION-LIST(...) / *MODIFY-LIST(...) / list-poss(48): <name 1..8>(...)
Angabe, ob die Kennung für den Zugang über POSIX-Remote-Login mit Terminal-Sets geschützt wird.

TERMINAL-SET = *NO-PROTECTION
Die Kennung wird nicht mit Terminal-Sets geschützt.

TERMINAL-SET = *NONE
Der Kennung wird für den Zugang über POSIX-Remote-Login eine leere Terminal-Set-Liste zugewiesen, d.h. es ist kein POSIX-Remote-Login erlaubt.

TERMINAL-SET = *EXCEPTION-LIST(...)
Es wird eine Negativliste von Terminal-Sets zugewiesen.

TERMINAL-SET = *NONE
Die Negativliste ist leer, d.h. POSIX-Remote-Login ist uneingeschränkt erlaubt.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über POSIX-Remote-Login verboten.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

TERMINAL-SET = *MODIFY-LIST(…)
Es werden Änderungen an einer bereits definierten Terminal-Set-Liste vorgenommen. Die Eigenschaft der Liste, ob sie eine Positiv- oder Negativliste ist, bleibt von der Modifikation unberührt.

REMOVE-TERMINAL-SETS =
Angabe von Terminal-Sets, die aus der Terminal-Set-Liste für den POSIX-Remote-Login-Zugang der Benutzerkennung entfernt werden sollen.

Falls für den POSIX-Remote-Login-Zugang der Benutzerkennung noch keine Terminal-Set-Liste definiert ist, wird eine Warnung ausgegeben und die Kommandobearbeitung fortgesetzt. Dasselbe gilt, wenn eines oder mehrere der zu entfernenden Terminal-Sets nicht in der Liste enthalten sind.

REMOVE-TERMINAL-SETS = *NONE
Es werden keine Terminal-Sets aus der Terminal-Set-Liste entfernt.

REMOVE-TERMINAL-SETS = *ALL
Alle Terminal-Sets werden aus der Terminal-Set-Liste entfernt.

REMOVE-TERMINAL-SETS = list-poss(48): <name 1..8>(…))
Die Terminal-Sets mit den angegebenen Namen werden aus der Terminal-Set-Liste entfernt.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

ADD-TERMINAL-SETS =
Angabe von Terminal-Sets, die in die definierte Terminal-Set-Liste für den POSIX-Remote-Login-Zugang der Benutzerkennung eingefügt werden sollen.

Falls für den POSIX-Remote-Login-Zugang der Benutzerkennung noch keine Terminal-Set-Liste definiert ist, wird implizit eine Positivliste angelegt. Wenn eines oder mehrere der einzufügenden Terminal-Sets bereits in der Liste enthalten sind, wird eine Warnung ausgegeben.

ADD-TERMINAL-SETS = *NONE
Es werden keine Terminal-Sets in die definierte Terminal-Set-Liste eingefügt.

ADD-TERMINAL-SETS = list-poss(48): <name 1..8>(…))
Die Terminal-Sets mit den angegebenen Namen werden in die definierte Terminal-Set-Liste eingefügt.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Es wird eine Positivliste von Terminal-Sets zugewiesen. Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über POSIX-Remote-Login erlaubt.

SCOPE =
Klasse des Terminal-Set Namens.

SCOPE = *STD
Ein systemglobaler Benutzerverwalter weist standardmäßig globale, ein Gruppenverwalter lokale Terminal-Sets zu.

SCOPE = *USER
Es wird ein Terminal-Set aus dem Eigentum der Benutzerkennung zugewiesen.

SCOPE = *GROUP
Es wird ein Terminal-Set aus dem Eigentum der Gruppe der Benutzerkennung zugewiesen.

SCOPE = *SYSTEM
Es wird ein Terminal-Set aus gemeinschaftlichem Eigentum zugewiesen.

GUARD-NAME = *UNCHANGED / *NONE / <filename 1..18 without-cat-gen-vers>
Gibt an, ob der Zugang über POSIX-Remote-Login mit einem Guard geschützt wird.

GUARD-NAME = *NONE
Der Zugang über POSIX-Remote-Login wird nicht mit einem Guard geschützt.

GUARD-NAME = <filename 1..18 without-cat-gen-vers>
Der Zugang über POSIX-Remote-Login ist nur erlaubt, wenn die Zugriffsbedingungen im angegebenen Guard erfüllt sind. Die geschützte Benutzerkennung muss berechtigter Benutzer des angegebenen Guards sein. Bei der Auswertung des Guards werden nur die Zeitbedingungen Date, Time und Weekday berücksichtigt. Subjekt der Zugriffsbedingung ist die geschützte Benutzerkennung.

POSIX-RLOGIN-ACCESS = NO
Die BS2000-Benutzerkennung ist für den Systemzugang über POSIX-Remote-Login gesperrt.

POSIX-REMOTE-ACCESS = *UNCHANGED / *YES(...) / *NO
Die BS2000-Benutzerkennung wird für den Systemzugang über ein POSIX-Remote-Kommando (z.B. rsh, rcp) geöffnet oder gesperrt.

TERMINAL-SET = *UNCHANGED / *NO-PROTECTION / *NONE / *EXCEPTION-LIST(...) / *MODIFY-LIST(…) / list-poss(48): <name 1..8>(…)
Angabe, ob die Kennung für den Zugang über ein POSIX-Remote-Kommando mit Terminal-Sets geschützt wird.

TERMINAL-SET = *NO-PROTECTION
Die Kennung wird nicht mit Terminal-Sets geschützt.

TERMINAL-SET = *NONE
Der Kennung wird für den Zugang über ein POSIX-Remote-Kommando eine leere Terminal-Set-Liste zugewiesen, d.h. es ist kein Zugang über ein POSIX-Remote-Kommando erlaubt.

TERMINAL-SET = *EXCEPTION-LIST(...)
Es wird eine Negativliste von Terminal-Sets zugewiesen.

TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
Die Negativliste ist leer, d.h. der Zugang über ein POSIX-Remote-Kommando ist uneingeschränkt erlaubt.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über ein POSIX-Remote-Kommando verboten.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

TERMINAL-SET = *MODIFY-LIST(…)
Es werden Änderungen an einer bereits definierten Terminal-Set-Liste vorgenommen. Die Eigenschaft der Liste, ob sie eine Positiv- oder Negativliste ist, bleibt von der Modifikation unberührt.

REMOVE-TERMINAL-SETS =
Angabe von Terminal-Sets, die aus der Terminal-Set-Liste für den POSIX-Remote-Kommando-Zugang der Benutzerkennung entfernt werden sollen.

Falls für den POSIX-Remote-Kommando-Zugang der Benutzerkennung noch keine Terminal-Set-Liste definiert ist, wird eine Warnung ausgegeben und die Kommandobearbeitung fortgesetzt. Dasselbe gilt, wenn eines oder mehrere der zu entfernenden Terminal-Sets nicht in der Liste enthalten sind.

REMOVE-TERMINAL-SETS = *NONE
Es werden keine Terminal-Sets aus der Terminal-Set-Liste entfernt.

REMOVE-TERMINAL-SETS = *ALL
Alle Terminal-Sets werden aus der Terminal-Set-Liste entfernt.

REMOVE-TERMINAL-SETS = list-poss(48): <name 1..8>(…))
Die Terminal-Sets mit den angegebenen Namen werden aus der Terminal-Set-Liste entfernt.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

ADD-TERMINAL-SETS =
Angabe von Terminal-Sets, die in die definierte Terminal-Set-Liste für den POSIX-Remote-Kommando-Zugang der Benutzerkennung eingefügt werden sollen.

Falls für den POSIX-Remote-Kommando-Zugang der Benutzerkennung noch keine Terminal-Set-Liste definiert ist, wird implizit eine Positivliste angelegt. Wenn eines oder mehrere der einzufügenden Terminal-Sets bereits in der Liste enthalten sind, wird eine Warnung ausgegeben.

ADD-TERMINAL-SETS = *NONE
Es werden keine Terminal-Sets in die definierte Terminal-Set-Liste eingefügt.

ADD-TERMINAL-SETS = list-poss(48): <name 1..8>(…))
Die Terminal-Sets mit den angegebenen Namen werden in die definierte Terminal-Set-Liste eingefügt.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET=list-poss(48): <name 1..8>(...).

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Es wird eine Positivliste von Terminal-Sets zugewiesen. Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über ein POSIX-Remote-Kommando erlaubt.

SCOPE =
Klasse des Terminal-Set Namens.

SCOPE = *STD
Ein systemglobaler Benutzerverwalter weist standardmäßig globale, ein Gruppenverwalter lokale Terminal-Sets zu.

SCOPE = *USER
Es wird ein Terminal-Set aus dem Eigentum der Benutzerkennung zugewiesen.

SCOPE = *GROUP
Es wird ein Terminal-Set aus dem Eigentum der Gruppe der Benutzerkennung zugewiesen.

SCOPE = *SYSTEM
Es wird ein Terminal-Set aus gemeinschaftlichem Eigentum zugewiesen.

GUARD-NAME = *UNCHANGED / *NONE / <filename 1..18 without-cat-gen-vers>
Gibt an, ob der Zugang über ein POSIX-Remote-Kommando mit einem Guard geschützt wird.

GUARD-NAME = *NONE
Der Zugang über ein POSIX-Remote-Kommando wird nicht mit einem Guard geschützt.

GUARD-NAME = <filename 1..18 without-cat-gen-vers>
Der Zugang über ein POSIX-Remote-Kommando ist nur erlaubt, wenn die Zugriffsbedingungen im angegebenen Guard erfüllt sind. Die geschützte Benutzerkennung muss berechtigter Benutzer des angegebenen Guards sein. Bei der Auswertung des Guards werden nur die Zeitbedingungen Date, Time und Weekday berücksichtigt. Subjekt der Zugriffsbedingung ist die POSIX-Benutzerkennung, unter der das Kommando *rsh* bzw. *rcp* eingegeben wurde. Es ist nicht notwendig, dass diese Kennung im BS2000 existiert.

POSIX-REMOTE-ACCESS = *NO
Die BS2000-Benutzerkennung ist für den Systemzugang über ein POSIX-Remote-Kommando gesperrt.

MODIFY-POSIX-USER-ATTRIBUTES
POSIX-Benutzerattribute ändern

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: POSIX-ADMINISTRATION,
USER-ADMINISTRATION,
STD-PROCESSING

Dieses Kommando ändert die POSIX-Benutzerattribute einer BS2000-Benutzerkennung im Benutzerkatalog des angegebenen Pubsets.

Für jede neue BS2000-Benutzerkennung werden bei ihrem Einrichten automatisch die POSIX-Benutzerattribute angelegt, die mit Standardwerten versehen sind (siehe Seite 190). Diese POSIX-Benutzerattribute können bei Bedarf geändert werden. Dazu sind folgende Benutzer berechtigt:

- Inhaber des Privilegs POSIX-ADMINISTRATION oder USER-ADMINISTRATION für alle BS2000-Benutzerkennungen auf allen Pubsets.
- Gruppenverwalter für die ihnen unterstellten Gruppen- und Untergruppenmitglieder auf dem von ihnen verwalteten Pubset. Für einen Gruppenverwalter gelten aber folgende Einschränkungen:
  - Seine Autorisierung ADM-AUTHORITY bestimmt die POSIX-Benutzerattribute, zu deren Verwaltung er berechtigt ist.
  - Der Wertebereich der POSIX-Benutzerattribute ist für ihn eingeschränkt.

  Näheres dazu steht beim entsprechenden Operanden in der Operandenbeschreibung.

Ab BS2000/OSD V9.0 steht zusätzlich das Kommando EDIT-POSIX-USER-ATTRIBUTES zur Verfügung, mit dem die aktuellen Einstellungen angezeigt und modifiziert werden können.

Format

MODIFY-**POSIX-USER-ATTR**IBUTES

USER-IDENTIFICATION = <name 1..8>

,**PUBSET** = ***HOME** / <catid 1..4>

,**USER-NUM**BER = ***UNCHA**NGED / ***BY-POSIX-USER-DEF**AULTS / ***HOME** / <integer 0..60002>

,**GR**OUP-**NUM**BER = ***UNCHA**NGED / ***BY-POSIX-USER-DEF**AULTS / ***GR**OUP-**ADM**INISTRATOR / <integer 0..60002>

,**COMMENT** = ***UNCHA**NGED / ***BY-POSIX-USER-DEF**AULTS / ***NONE** / <c-string 1..255 with-low>

,**DIR**ECTORY = ***UNCHA**NGED / ***BY-POSIX-USER-DEF**AULTS / ***ROOT** / <posix-pathname 1..1023 without-wild>

,**PROG**RAM = ***UNCHA**NGED / ***BY-POSIX-USER-DEF**AULTS / ***SHELL** / <posix-pathname 1..1023 without-wild>

Operandenbeschreibung

USER-IDENTIFICATION = <name 1..8>
BS2000-Benutzerkennung, deren POSIX-Benutzerattribute geändert werden sollen.

PUBSET =
Pubset, in dessen Benutzerkatalog die POSIX-Benutzerattribute geändert werden sollen.

PUBSET = *HOME
Die Änderung erfolgt auf dem Home-Pubset.

PUBSET = <catid 1..4>
Die Änderung erfolgt auf dem Pubset mit der angegebenen Katalogkennung.

USER-NUMBER =
Die Benutzernummer, die beim Einrichten einer BS2000-Benutzerkennung automatisch vergeben wird, kann geändert werden.

Das Attribut USER-NUMBER ist sicherheitsrelevant, da die Benutzernummer die Privilegierung ausdrückt und den Eigentümer einer Datei bestimmt.

Der Gruppenverwalter kann die Benutzernummer nur ändern, wenn er mindestens das Gruppenverwalterrecht MANAGE-MEMBERS besitzt. Für ihn ist aber der Wertebereich eingeschränkt:
– Er kann nicht die Benutzernummer 0 vergeben, d.h. die Root-Berechtigung.
– Er kann nur die Standard-Benutzernummer ändern.
– Er kann nur Benutzernummern vergeben, die größer als die Standard-Benutzernummer sind.
– Er kann Benutzernummern nicht mehrfach vergeben.
– Er kann auf einem Daten-Pubset nur die Benutzernummer der gleichnamigen BS2000-Benutzerkennung auf dem Home-Pubset zuweisen.

USER-NUMBER = *UNCHANGED
Die Benutzernummer wird nicht geändert.

USER-NUMBER = *BY-POSIX-USER-DEFAULTS
Die Benutzernummer erhält den entsprechenden Standardwert, der im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

USER-NUMBER = *HOME
Die Benutzernummer der gleichnamigen BS2000-Benutzerkennung auf dem Home-Pubset wird übernommen.
Dieser Wert ist nur von Bedeutung, wenn die Benutzernummer auf einem Daten-Pubset geändert wird. Auf dem Home-Pubset ist diese Angabe redundant.

USER-NUMBER = <integer 0..60002>
Die Benutzernummer erhält den angegebenen Wert.

GROUP-NUMBER =
Die Gruppennummer, die beim Einrichten einer BS2000-Benutzerkennung automatisch vergeben wird, kann geändert werden.

Das Attribut GROUP-NUMBER ist sicherheitsrelevant, da POSIX beim Login nicht die Zulässigkeit der Kombination BS2000-Benutzerkennung und -Gruppe gegen den POSIX-Gruppenkatalog prüft.

Der Gruppenverwalter kann die Gruppennummer nur ändern, wenn er das Gruppenverwalterrecht MANAGE-MEMBERS besitzt. Für ihn ist aber der Wertebereich eingeschränkt:
– Er kann nur die Gruppennummer vergeben, die der Gruppenverwalter der BS2000-Benutzergruppe besitzt, deren Mitglied die BS2000-Benutzerkennung ist, oder die Standard-Gruppennummer.
– Er kann für seine eigene BS2000-Benutzerkennung keine andere Gruppennummer vergeben.

GROUP-NUMBER = *UNCHANGED
Die Gruppennummer wird nicht geändert.

GROUP-NUMBER = *BY-POSIX-USER-DEFAULTS
Die Gruppennummer erhält den entsprechenden Standardwert, der im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

GROUP-NUMBER = *GROUP-ADMINISTRATOR
Es wird die Gruppennummer vergeben, die der Gruppenverwalter der BS2000-Benutzergruppe besitzt, deren Mitglied die BS2000-Benutzerkennung ist.

GROUP-NUMBER = <integer 0..60002>
Die Gruppennummer erhält den angegebenen Wert.

COMMENT =
Der Kommentar kann geändert werden. Nach eigenem Ermessen können nähere Angaben zum Eigentümer der BS2000-Benutzerkennung gemacht werden.

Hinweis
Dieser Kommentar wird z.B. von Mail-Programmen zur Beschreibung des Absenders verwendet.

COMMENT = *UNCHANGED
Der Kommentar wird nicht geändert.

COMMENT = *BY-POSIX-USER-DEFAULTS
Der Wert des entsprechenden POSIX-Standardattributs wird übernommen, das im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

COMMENT = *NONE
Es wird kein Kommentar eingetragen.

COMMENT = <c-string 1..255 with-low>
Der angegebene Kommentar wird eingetragen.

DIRECTORY =
Der absolute Pfadname zum Login-Verzeichnis des Benutzers kann geändert werden.

Dieses Attribut ist nicht sicherheitsrelevant, da es nur den Inhalt der Shell-Variablen *HOME* und den Anfangswert des Arbeitsverzeichnisses bestimmt. Die Schutzattribute von Dateien und Dateiverzeichnissen können damit nicht umgangen werden.

DIRECTORY = *UNCHANGED
Der absolute Pfadname wird nicht geändert.

DIRECTORY = *BY-POSIX-USER-DEFAULTS
Der Wert des entsprechenden POSIX-Standardattributs wird übernommen, das im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

DIRECTORY = *ROOT
Das Root-Verzeichnis "/" wird zugeordnet.

DIRECTORY = <posix-pathname 1..1023 without-wild>
Der angegebene Pfadname wird übernommen.

PROGRAM =
Das Programm kann geändert werden, das nach dem Kommando *rlogin* bzw. nach dem Aufruf des Kommandos START-POSIX-SHELL gestartet wird.

Dieses Attribut ist nicht sicherheitsrelevant, da nur solche Programme gestartet werden können, die der Benutzer ausführen darf.

PROGRAM = *UNCHANGED
Das Programm wird nicht geändert.

PROGRAM = *BY-POSIX-USER-DEFAULTS
Der Wert des entsprechenden POSIX-Standardattributs wird übernommen, das im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

PROGRAM = *SHELL
Die POSIX-Shell wird gestartet.

PROGRAM = <posix-pathname 1..1023 without-wild>
Das angegebene Programm wird gestartet.

Kommando-Returncodes

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Kommando fehlerfrei ausgeführt
2	0	SRM6001	Kommando mit Warnung ausgeführt
	32	SRM6020	Kommando wegen eines Systemfehlers abgewiesen
	130	SRM6030	Kommando wegen Ressourcenmangel abgewiesen
	64	SRM6040	Kommando mit Fehlermeldung abgewiesen

Beispiele

Der Benutzerkennung POSIXTST soll die Benutzernummer 107 und die Gruppennummer 66 zugeordnet werden. Das Login-Verzeichnis (Home-Verzeichnis) soll */home/posixtst* heißen. Nach dem POSIX-Login soll die Bourne-Shell aufgerufen werden. Der Kommentar soll lauten: „posix-user@posix-server.com“.

```
/MODIFY-POSIX-USER-ATTRIBUTES USER-ID=POSIXTST, -
/                             USER-NUMBER=107, -
/                             GROUP-NUMBER=66, -
/                             DIRECTORY=/home/posixtst, -
/                            COMMENT='posix-user@posix-server.com'
```

Die Benutzerkennung PSXROOT soll die Root-Berechtigung erhalten. Als Home-Verzeichnis soll */home/psxroot* eingetragen werden.

```
/MODIFY-POSIX-USER-ATTRIBUTES USER-ID=PSXROOT, -
/                             USER-NUMBER=0, -
/                             GROUP-NUMBER=0, -
/                            DIRECTORY=/home/psxroot
```

MODIFY-POSIX-USER-DEFAULTS
Standardwerte für POSIX-Benutzerattribute ändern

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: POSIX-ADMINISTRATION,
USER-ADMINISTRATION,
STD-PROCESSING

Dieses Kommando ändert die POSIX-Standardattribute im Benutzerkatalog des angegebenen Pubsets. Folgende Benutzer dürfen es ausführen:

- Inhaber des Privilegs POSIX-ADMINISTRATION oder USER-ADMINISTRATION für alle Pubsets.
- Gruppenverwalter der Gruppe *UNIVERSAL auf dem von ihnen verwalteten Pubset.

Die POSIX-Standardattribute werden beim Anlegen eines neuen Benutzers (mit ADD-USER) verwendet.

Ab BS2000/OSD V9.0 steht zusätzlich das Kommando EDIT-POSIX-USER-DEFAULTS zur Verfügung, mit dem die aktuellen Einstellungen angezeigt und modifiziert werden können.

Format

MODIFY-**POSIX-USER-DEF**AULTS

PUBSET = ***HOME** / <catid 1..4>

,USER-NUMBER = ***UNCHA**NGED / <integer 0..60002>

,GROUP-**NUM**BER = ***UNCHA**NGED / <integer 0..60002>

,COMMENT = ***UNCHA**NGED / ***NONE** / <c-string 1..255 with-low>

,DIRECTORY = ***UNCHA**NGED / ***ROOT** / <posix-pathname 1..1023 without-wild>

,**PROG**RAM = ***UNCHA**NGED / ***SHELL** / <posix-pathname 1..1023 without-wild>

Operandenbeschreibung

PUBSET =
Pubset, in dessen Benutzerkatalog die POSIX-Standardattribute geändert werden sollen.

PUBSET = *HOME
Die POSIX-Standardattribute werden im Benutzerkatalog des Home-Pubsets geändert.

PUBSET = <catid 1..4>
Die POSIX-Standardattribute werden im Benutzerkatalog des angegebenen Pubsets geändert.

USER-NUMBER =
Die Benutzernummer kann geändert werden.

USER-NUMBER = *UNCHANGED
Die Benutzernummer wird nicht geändert.

USER-NUMBER = <integer 0..60002>
Die Benutzernummer erhält den angegebenen Wert.

GROUP-NUMBER =
Die Gruppennummer kann geändert werden.

GROUP-NUMBER = *UNCHANGED
Die Gruppennummer wird nicht geändert.

GROUP-NUMBER = <integer 0..60002>
Die Gruppennummer erhält den angegebenen Wert.

COMMENT =
Der Kommentar kann geändert werden.

Hinweis
Dieser Kommentar wird z.B. von Mail-Programmen zur Beschreibung des Absenders verwendet.

COMMENT = *UNCHANGED
Der Kommentar wird nicht geändert.

COMMENT = *NONE
Es wird kein Kommentar eingetragen.

COMMENT = <c-string 1..255 with-low>
Der angegebene Kommentar wird eingetragen.

DIRECTORY =
Der absolute Pfadname zum Login-Verzeichnis des Benutzers kann geändert werden.

DIRECTORY = *UNCHANGED
Der absolute Pfadname wird nicht geändert.

DIRECTORY = *ROOT
Es wird ins Root-Verzeichnis gewechselt.

DIRECTORY = <posix-pathname 1..1023 without-wild>
Es wird zum angegebenen Pfadnamen gewechselt.

PROGRAM =
Das Programm, das nach der Anmeldung des Benutzers gestartet wird, kann geändert werden.

PROGRAM = *UNCHANGED
Das Programm wird nicht geändert.

PROGRAM = *SHELL
Die standardmäßige POSIX-Shell wird gestartet.

PROGRAM = <posix-pathname 1..1023 without-wild>
Das angegebene Programm wird gestartet.

Kommando-Returncodes

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Kommando fehlerfrei ausgeführt
2	0	SRM6001	Kommando mit Warnung ausgeführt
	32	SRM6020	Kommando wegen eines Systemfehlers abgewiesen
	64	SRM6040	Kommando mit Fehlermeldung abgewiesen
	130	SRM6030	Kommando wegen Ressourcenmangel abgewiesen

MODIFY-USER-ATTRIBUTES
Katalogeintrag eines Benutzers ändern

Anwendungsbereich: ACCOUNTING,
USER-ADMINISTRATION

Privilegierung: USER-ADMINISTRATION,
STD-PROCESSING

Dieses Kommando ändert die Attribute einer BS2000-Benutzerkennung im Benutzerkatalog. Folgende Benutzer dürfen es ausführen:

- Inhaber des Privilegs USER-ADMINISTRATION für alle BS2000-Benutzerkennungen.
- Gruppenverwalter für die BS2000-Benutzerkennungen, die ihren Gruppen zu- und untergeordnet sind.

Die folgende Syntaxdarstellung zeigt nur den POSIX-relevanten Teil des Kommandos. Das vollständige Kommando ist in den Handbüchern „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9] und „Kommandos“ [28] beschrieben.

Ab BS2000/OSD V9.0 steht zusätzlich das Kommando EDIT-USER-ATTRIBUTES zur Verfügung, mit dem die aktuellen Einstellungen angezeigt und modifiziert werden können.

Format

MODIFY-**USER**-ATTRIBUTES

. . .
,**ACCOUNT-ATTR**IBUTES = ***UNCHA**NGED / ***ADD**(...) / ***MOD**IFY(...) / ***REMOVE**(...)

***ADD**(...)
ACCOUNT = <alphanum-name 1..8>
. . .
,**POSIX-RLOGIN-DEFAULT** = ***NO** / ***Y**ES

***MOD**IFY(...)
ACCOUNT = <alphanum-name 1..8>
. . .
,**POSIX-RLOGIN-DEFAULT** = ***UNCHANGED** / ***NO** / ***Y**ES

***REMOVE**(...)

Operandenbeschreibung

ACCOUNT-ATTRIBUTES = ... / *ADD(...) / *MODIFY(...) / ...
Legt die Abrechnungsdaten einer BS2000-Benutzerkennung fest.

ACCOUNT-ATTRIBUTES = *ADD(...)
Eine neue Abrechnungsnummer und spezifische Attribute werden für die BS2000-Benutzerkennung eingetragen.

ACCOUNT = <alphanum-name 1..8>
Abrechnungsnummer der BS2000-Benutzerkennung, die in den Benutzerkatalog aufgenommen wird und auf die sich die folgenden Angaben beziehen.

. . .

POSIX-RLOGIN-DEFAULT =
Legt fest, ob die angegebene Abrechnungsnummer zur Abrechnung des Remote-Login-Systemlaufs herangezogen wird.

POSIX-RLOGIN-DEFAULT = *NO
Die angegebene Abrechnungsnummer wird nicht zur Abrechnung herangezogen.

POSIX-RLOGIN-DEFAULT = *YES
Die angegebene Abrechnungsnummer wird zur Abrechnung herangezogen.

ACCOUNT-ATTRIBUTES = *MODIFY(...)
Die Attribute einer eingetragenen Abrechnungsnummer der BS2000-Benutzerkennung werden geändert.

ACCOUNT = <alphanum-name 1..8>
Abrechnungsnummer der BS2000-Benutzerkennung, für die die nachfolgenden Werte im Benutzerkatalog geändert werden.

. . .

POSIX-RLOGIN-DEFAULT =
Legt fest, ob die zu ändernde Abrechnungsnummer zur Abrechnung des Remote-Login-Systemlaufs herangezogen wird.

POSIX-RLOGIN-DEFAULT = *UNCHANGED
Der bisher eingestellte Wert bleibt erhalten.

POSIX-RLOGIN-DEFAULT = *NO
Die Abrechnungsnummer wird nicht zur Abrechnung herangezogen.

POSIX-RLOGIN-DEFAULT = *YES
Die Abrechnungsnummer wird zur Abrechnung herangezogen.

Innerhalb einer BS2000-Benutzerkennung ist die Abrechnungsnummer für Remote Login eindeutig. Die Benutzerverwaltung führt automatisch einen Abgleich mit den vorhandenen Abrechnungsnummern durch.

Die Abrechnungsnummer für Remote Login kann auch für die Abrechnung eines BS2000-Systemlaufs angegeben werden. Dadurch kann eine BS2000-Benutzerkennung mit einer einzigen Abrechnungsnummer auskommen.

POSIX-RLOGIN-DEFAULT=*YES ist für Benutzerkennungen erforderlich, die einen Remote-Zugang zu POSIX wünschen (*rlogin*- oder Telnet-Zugang, Kommandos *rsh* und *rcp*) oder die *at*, *crontab* bzw. *batch* nutzen wollen.

Wenn keine Abrechnungsnummer für Remote Login angegeben ist, wird der Systemzugang über Remote Login abgewiesen, außer wenn es sich um die Benutzerkennung TSOS handelt.

SET-LOGON-PROTECTION
Schutzattribute vereinbaren

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: STD-PROCESSING, USER-ADMINISTRATION

Vereinbart Schutzattribute für existierende Benutzerkennungen.

Berechtigt zur Ausführung des Kommandos sind:

– Systemglobale Benutzerverwalter (Inhaber des Privilegs USER-ADMINISTRATION) für alle Benutzerkennungen

– Gruppenverwalter, die mindestens das Attribut MANAGE-MEMBERS besitzen, für die ihrer Benutzergruppe zu- und untergeordneten Benutzerkennungen.

An der Benutzeroberfläche des Kommandos SHOW-LOGON-PROTECTION ändert sich durch POSIX nichts. Die folgende Syntaxdarstellung zeigt nur den POSIX-relevanten Teil des Kommandos. Zusätzlich kann noch der Operand BATCH-ACCESS von Bedeutung sein (z.B. für at, batch, crontab).

Das vollständige Kommando finden Sie im Handbuch „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9].

Format

SET-LOGON-PROTECTION

```
. . .
,POSIX-RLOGIN-ACCESS = *YES (...) / *NO
    *YES(...)
        PASSWORD-CHECK = *YES / *NO
        ,TERMINAL-SET = *NO-PROTECTION / *NONE / *EXCEPTION-LIST(…) /
                        list-poss(48): <name 1..8>(…)
            *EXCEPTION-LIST (…)
                TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
                    <name 1..8>(…)
                        SCOPE = *STD / *USER / *GROUP / *SYSTEM
            <name 1..8>(…)
                SCOPE = *STD / *USER / *GROUP / *SYSTEM
        ,GUARD-NAME = *NONE / <filename 1..18 without-cat-gen-vers>
,POSIX-REMOTE-ACCESS = *YES (...) / *NO
    *YES(...)
        ,TERMINAL-SET = *NO-PROTECTION / *NONE / *EXCEPTION-LIST(…) /
                        list-poss(48): <name 1..8>(…)
            *EXCEPTION-LIST (…)
                (TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
                    <name 1..8>(…)
                        SCOPE = *STD / *USER / *GROUP / *SYSTEM
            <name 1..8>(…)
                SCOPE = *STD / *USER / *GROUP / *SYSTEM
        ,GUARD-NAME = *NONE / <filename 1..18 without-cat-gen-vers>
```

Operandenbeschreibung

POSIX-RLOGIN-ACCESS =
Die Zugangsklassen-Attribute für POSIX-Remote-Login können festgelegt werden.

POSIX-RLOGIN-ACCESS = *YES(...)
Die BS2000-Benutzerkennung ist für den Systemzugang über POSIX-Remote-Login offen.

PASSWORD-CHECK = *YES / *NO
Legt fest, ob beim Zugang über POSIX-Remote-Login eine Kennwortprüfung stattfindet.

TERMINAL-SET =
Angabe, ob die Kennung für den Zugang über POSIX-Remote-Login mit Terminal-Sets geschützt wird.

TERMINAL-SET = *NO-PROTECTION
Die Kennung wird nicht mit Terminal-Sets geschützt.

TERMINAL-SET = *NONE
Der Kennung wird eine leere Terminal-Set-Liste zugewiesen, d.h. es ist kein POSIX-Remote-Login erlaubt.

TERMINAL-SET = *EXCEPTION-LIST(...)
Es wird eine Negativliste von Terminal-Sets zugewiesen.

TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
Die Negativliste ist leer, d.h. POSIX-Remote-Login ist uneingeschränkt erlaubt.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über POSIX-Remote-Login verboten.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Es wird eine Positivliste von Terminal-Sets zugewiesen. Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über POSIX-Remote-Login erlaubt.

SCOPE =
Klasse des Terminal-Set Namens.

SCOPE = *STD
Ein systemglobaler Benutzerverwalter weist standardmäßig globale, ein Gruppenverwalter lokale Terminal-Sets zu.

SCOPE = *USER
Es wird ein Terminal-Set aus dem Eigentum der Benutzerkennung zugewiesen.

SCOPE = *GROUP
Es wird ein Terminal-Set aus dem Eigentum der Gruppe der Benutzerkennung zugewiesen.

SCOPE = *SYSTEM
Es wird ein Terminal-Set aus gemeinschaftlichem Eigentum zugewiesen.

GUARD-NAME =
Gibt an, ob der Zugang über POSIX-Remote-Login mit einem Guard geschützt wird.

GUARD-NAME = *NONE
Der Zugang über POSIX-Remote-Login wird nicht mit einem Guard geschützt.

GUARD-NAME = <filename 1..18 without-cat-gen-vers>
Der Zugang über POSIX-Remote-Login ist nur erlaubt, wenn die Zugriffsbedingungen im angegebenen Guard erfüllt sind. Die geschützte Benutzerkennung muss berechtigter Benutzer des angegebenen Guards sein. Bei der Auswertung des Guards werden nur die Zeitbedingungen Date, Time und Weekday berücksichtigt. Subjekt der Zugriffsbedingung ist die geschützte Benutzerkennung.

POSIX-RLOGIN-ACCESS = *NO
Die BS2000-Benutzerkennung ist für den Systemzugang über POSIX-Remote-Login gesperrt.

POSIX-REMOTE-ACCESS = *YES(...) / *NO
Die BS2000-Benutzerkennung wird für den Systemzugang über ein POSIX-Remote-Kommando geöffnet oder gesperrt.

TERMINAL-SET =
Angabe, ob die Kennung für den Zugang über ein POSIX-Remote-Kommando mit Terminal-Sets geschützt wird.

TERMINAL-SET = *NO-PROTECTION
Die Kennung wird nicht mit Terminal-Sets geschützt.

TERMINAL-SET = *NONE
Der Kennung wird eine leere Terminal-Set-Liste zugewiesen, d.h. es ist kein Zugang über ein POSIX-Remote-Kommando erlaubt.

TERMINAL-SET = *EXCEPTION-LIST(...)
Es wird eine Negativliste von Terminal-Sets zugewiesen.

TERMINAL-SET = *NONE / list-poss(48): <name 1..8>(…)
Die Negativliste ist leer, d.h. der Zugang über ein POSIX-Remote-Kommando ist uneingeschränkt erlaubt.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über ein POSIX-Remote-Kommando verboten.

Die Bedeutung der untergeordneten Operanden ist wie beim folgenden Operanden TERMINAL-SET.

TERMINAL-SET = list-poss(48): <name 1..8>(…)
Es wird eine Positivliste von Terminal-Sets zugewiesen. Den Datensichtstationen mit den Namen, die auf die Datensichtstationsnamen in den angegebenen Terminal-Sets passen, wird der Zugang über ein POSIX-Remote-Kommando erlaubt.

SCOPE =
Klasse des Terminal-Set Namens.

SCOPE = *STD
Ein systemglobaler Benutzerverwalter weist standardmäßig globale, ein Gruppenverwalter lokale Terminal-Sets zu.

SCOPE = *USER
Es wird ein Terminal-Set aus dem Eigentum der Benutzerkennung zugewiesen.

SCOPE = *GROUP
Es wird ein Terminal-Set aus dem Eigentum der Gruppe der Benutzerkennung zugewiesen.

SCOPE = *SYSTEM
Es wird ein Terminal-Set aus gemeinschaftlichem Eigentum zugewiesen.

GUARD-NAME =
Gibt an, ob der Zugang über ein POSIX-Remote-Kommando mit einem Guard geschützt wird.

GUARD-NAME = *NONE
Der Zugang über ein POSIX-Remote-Kommando wird nicht mit einem Guard geschützt.

GUARD-NAME = <filename 1..18 without-cat-gen-vers>
Der Zugang über ein POSIX-Remote-Kommando ist nur erlaubt, wenn die Zugriffsbedingungen im angegebenen Guard erfüllt sind. Die geschützte Benutzerkennung muss berechtigter Benutzer des angegebenen Guards sein. Bei der Auswertung des Guards werden nur die Zeitbedingungen Date, Time und Weekday berücksichtigt. Subjekt der Zugriffsbedingung ist die POSIX-Benutzerkennung, unter der das Kommando *rsh* bzw. *rcp* eingegeben wurde. Es ist nicht notwendig, dass diese Kennung im BS2000 existiert.

POSIX-REMOTE-ACCESS = *NO
Die BS2000-Benutzerkennung ist für den Systemzugang über ein POSIX-Remote-Kommando gesperrt.

Beispiel

Die Benutzerkennung PSXROOT wird für den Systemzugang über einen fernen Rechner zugelassen:

```
/SET-LOGON-PROTECTION USER-ID=PSXROOT,POSIX-RLOGIN-ACCESS=*YES
```

SHOW-LOGON-PROTECTION
Schutzattribute anzeigen

An der Benutzeroberfläche des Kommandos SHOW-LOGON-PROTECTION ändert sich durch POSIX nichts. Das vollständige Kommando finden Sie im Handbuch „SECOS (BS2000/OSD) Security Control System - Zugangs- und Zugriffskontrolle“ [9].

Folgende Tabelle zeigt die möglichen POSIX-spezifischen Inhalte im Feld Type der Zugangshistorie und deren Bedeutung:

Type	**Bedeutung**
POS-BATCH	POSIX-Batch-Kommandos *at*, *cron* oder *batch*
POS-REMOTE	POSIX-Remote-Kommandos *rcp* oder *rsh*
RLOGIN	POSIX Remote-Login

POSIX-spezifische S-Variablen

Ausgabe-Information	Name der S-Variablen	T	Inhalt	Bedingung
Zugangskontrolle beim POSIX-Remote-Zugang aktiv	Var(*LIST).POSIX-REM.ACCESS	S	*YES *NO	1
Name des Guards, mit dem der POSIX-Remote-Zugang geschützt wird	Var(*LIST).POSIX-REM.GUARD	S	*NONE <filename 1..18>	1
POSIX-Remote: Password-Check	Var(*LIST).POSIX-REM.PASS-CHECK	S	*YES *NO	1
Terminal-Sets der Klasse GROUP	Var(*LIST).POSIX-REM.TER-SET.GROUP(*LIST)	S	<name 1..8>	1
Gruppenname	Var(*LIST).POSIX-REM.TER-SET.GROUP-ID	S	<name 1..8> *UNIV	1
Terminal-Sets der Klasse SYSTEM	Var(*LIST).POSIX-REM.TER-SET.SYSTEM(*LIST)	S	<name 1..8>	1
Terminal-Sets der Klasse USER	Var(*LIST).POSIX-REM.TER-SET.USER(*LIST)	S	<name 1..8>	1
Benutzerkennung	Var(*LIST).POSIX-REM.TER-SET.USER-ID	S	<name 1..8>	1
POSIX-Remote-Zugang mit Terminal-Sets geschützt	Var(*LIST).POSIX-REM.TER-SET-DEFI	S	*NO-PROT *LIST *EXCEPT	1
Zugangskontrolle beim POSIX-Zugang über rlogin aktiv?	var(*LIST).POSIX-RLOG.ACCESS	S	*NO *YES	1
Name des Guards, mit dem der POSIX-Rlogin-Zugang geschützt wird	Var(*LIST).POSIX-RLOG.GUARD	S	*NONE <filename 1..18>	1
Kennwortprüfung beim POSIX-Zugang über rlogin aktiv?	var(*LIST).POSIX-RLOG.PASS-CHECK	S	*NO *YES	1
Terminal-Sets der Klasse GROUP	Var(*LIST).POSIX-RLOG.TER-SET.GROUP(*LIST)	S	<name 1..8>	1
Gruppenname	Var(*LIST).POSIX-RLOG.TER-SET.GROUP-ID	S	<name 1..8> *UNIV	1
Terminal-Sets der Klasse SYSTEM	Var(*LIST).POSIX-RLOG.TER-SET.SYSTEM(*LIST)	S	<name 1..8>	1
Terminal-Sets der Klasse USER	Var(*LIST).POSIX-RLOG.TER-SET.USER(*LIST)	S	<name 1..8>	1
Benutzerkennung	Var(*LIST).POSIX-RLOG.TER-SET.USER-ID	S	<name 1..8>	1
POSIX-Rlogin-Zugang mit Terminal-Sets geschützt	Var(*LIST).POSIX-RLOG.TER-SET-DEFI	S	*NO-PROT *LIST *EXCEPT	1
POSIX-Server: Access	Var(*LIST).POSIX-SERVER.ACCESS	S	*YES *NO	1

SHOW-POSIX-STATUS
POSIX-Status anzeigen

Anwendungsbereich: SYSTEM-MANAGEMENT

Privilegierung: SUBSYSTEM-MANAGEMENT

Dieses Kommando gibt den Zustand von POSIX aus.

Format

SHOW-POSIX-STATUS

Das Kommando gibt den aktuellen Zustand des Subsystems POSIX in folgender Form nach SYSOUT aus:

```
%POSSTAT POSIX-STATUS=<status>
```

<status> kann folgende Werte annehmen:

POSIX-STATUS	**Bedeutung**
*AVAILABLE	POSIX ist für Anwendungen freigegeben.
*IN-CREATE	Das Subsystem POSIX wird gestartet.
*IN-DELETE	Das Subsystem POSIX wird beendet.
*NOT-ACCESSIBLE	Das Subsystem POSIX ist gestartet, aber nicht für Anwendungen freigegeben.
*NOT-AVAILABLE	Das Subsystem POSIX ist nicht geladen.
*UNKNOWN	Der Zustand konnte nicht ermittelt werden.

Kommando-Returncodes

(SC2)	**SC1**	**Maincode**	**Bedeutung**
	0	CMD0001	Kommando fehlerfrei ausgeführt

SHOW-POSIX-USER-ATTRIBUTES
POSIX-Benutzerattribute anzeigen

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: POSIX-ADMINISTRATION, USER-ADMINISTRATION, STD-PROCESSING

Dieses Kommando zeigt die POSIX-Benutzerattribute einer BS2000-Benutzerkennung an, die im Benutzerkatalog des angegebenen Pubsets eingetragen sind. Folgende Benutzer dürfen es ausführen:

- Inhaber des Privilegs POSIX-ADMINISTRATION oder USER-ADMINISTRATION für alle BS2000-Benutzerkennungen auf allen Pubsets.
- Gruppenverwalter für die Gruppen- und Untergruppenmitglieder, die ihnen unterstellt sind, auf dem von ihnen verwalteten Pubset.
- jeder Benutzer für seine eigene BS2000-Benutzerkennung.

Format

SHOW-POSIX-USER-ATTRIBUTES

USER-IDENTIFICATION = ***OWN** / ***ALL** / list-poss(20): <name 1..8>

,**PUBSET** = ***HOME** / ***ALL** / list-poss(20): <catid 1..4>

,**SEL**ECT = ***ALL** / ***BY-ATTR**IBUTES(...)

***BY-ATTR**IBUTES(...)

USER-NUMBER = ***ANY** / ***BY-POSIX-USER-DEF**AULTS / ***OWN** / <integer 0..60002>

,**GR**OUP-**NUM**BER = ***ANY** / ***BY-POSIX-USER-DEF**AULTS / ***OWN** / <integer 0..60002>

,**COMMENT** = ***ANY** / ***BY-POSIX-USER-DEF**AULTS / ***NONE** / <c-string 1..255 with-low>

,**DIR**ECTORY = ***ANY** / ***BY-POSIX-USER-DEF**AULTS / ***ROOT** / <posix-pathname 1..1023 without-wild>

,**PROG**RAM = ***ANY** / ***BY-POSIX-USER-DEF**AULTS / ***SHELL** / <posix-pathname 1..1023 without-wild>

,**INF**ORMATION = ***ALL** / ***USER-LIST**

,**OUTPUT** = list-poss(2): ***SYSOUT** / ***SYSLST**(...)

***SYSLST**(...)

SYSLST-NUMBER = ***STD** / <integer 1..99>

,**LINE**S-PER-**PAGE** = **64** / <integer 20..255>

Operandenbeschreibung

USER-IDENTIFICATION =
BS2000-Benutzerkennungen, deren POSIX-Benutzerattribute angezeigt werden sollen.

USER-IDENTIFICATION = *OWN
Die POSIX-Benutzerattribute der eigenen BS2000-Benutzerkennung werden angezeigt, die im Benutzerkatalog des angegebenen Pubsets eingetragen sind.

USER-IDENTIFICATION = *ALL
Die POSIX-Benutzerattribute aller BS2000-Benutzerkennungen, zu deren Kenntnisnahme der Aufrufer berechtigt ist, werden angezeigt.

USER-IDENTIFICATION = list-poss(20): <name 1..8>
Die POSIX-Benutzerattribute der angegebenen Kennung werden angezeigt.

PUBSET =
Bestimmt das Pubset, aus dessen Benutzerkatalog die POSIX-Benutzerattribute angezeigt werden sollen.

PUBSET = *HOME
Die POSIX-Benutzerattribute des Home-Pubsets werden angezeigt.

PUBSET = *ALL
Die POSIX-Benutzerattribute aller Pubsets, die zum Zeitpunkt der Kommandoeingabe verfügbar sind, werden angezeigt.

PUBSET = list-poss(20): <catid 1..4>
Die POSIX-Benutzerattribute des angegebenen Pubsets werden angezeigt.

SELECT =
Die BS2000-Benutzerkennungen werden anhand ihrer POSIX-Benutzerattribute ausgewählt.

SELECT = *ALL
Die BS2000-Benutzerkennungen werden unabhängig von ihren POSIX-Benutzerattributen ausgewählt.

SELECT = *BY-ATTRIBUTES(...)
Die BS2000-Benutzerkennungen werden abhängig von ihren POSIX-Benutzerattributen ausgewählt. Wenn mehr als ein POSIX-Benutzerattribut angegeben ist, erfolgt die Auswahl durch „und“-Verknüpfung.

USER-NUMBER =
Die BS2000-Benutzerkennungen werden anhand ihrer Benutzernummer ausgewählt.

USER-NUMBER = *ANY
Die BS2000-Benutzerkennungen werden unabhängig von ihrer Benutzernummer ausgewählt.

USER-NUMBER = *BY-POSIX-USER-DEFAULTS
Nur die BS2000-Benutzerkennungen werden ausgewählt, für die als Benutzernummer der entsprechende Standardwert im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

USER-NUMBER = *OWN
Nur die BS2000-Benutzerkennungen werden ausgewählt, die dieselbe Benutzernummer wie der Aufrufer im Benutzerkatalog des angegebenen Pubsets eingetragen haben.

USER-NUMBER = <integer 0..60002>
Nur die BS2000-Benutzerkennungen werden ausgewählt, die die angegebene Benutzernummer im Benutzerkatalog des angegebenen Pubsets eingetragen haben.

GROUP-NUMBER =
Die BS2000-Benutzerkennungen werden anhand ihrer Gruppennummer ausgewählt.

GROUP-NUMBER = *ANY
Die BS2000-Benutzerkennungen werden unabhängig von ihrer Gruppennummer ausgewählt.

GROUP-NUMBER = *BY-POSIX-USER-DEFAULTS
Nur die BS2000-Benutzerkennungen werden ausgewählt, für die als Gruppennummer der entsprechende Standardwert im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

GROUP-NUMBER = *OWN
Nur die BS2000-Benutzerkennungen werden ausgewählt, die dieselbe Gruppennummer wie der Aufrufer im Benutzerkatalog des angegebenen Pubsets eingetragen haben.

GROUP-NUMBER = <integer 0..60002>
Nur die BS2000-Benutzerkennungen werden ausgewählt, die die angegebene Gruppennummer im Benutzerkatalog des angegebenen Pubsets eingetragen haben.

COMMENT =
Die BS2000-Benutzerkennungen werden anhand ihres Kommentars ausgewählt.

COMMENT = *ANY
Die BS2000-Benutzerkennungen werden unabhängig von ihrem Kommentar ausgewählt.

COMMENT = *BY-POSIX-USER-DEFAULTS
Nur die BS2000-Benutzerkennungen werden ausgewählt, für die als Kommentar der entsprechende Standardwert im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

COMMENT = *NONE
Nur die BS2000-Benutzerkennungen mit einem leeren Kommentar werden ausgewählt.

COMMENT = <c-string 1..255 with-low>
Nur die BS2000-Benutzerkennungen mit dem angegebenen Kommentar werden ausgewählt.

DIRECTORY =
Die BS2000-Benutzerkennungen werden anhand ihres Login-Verzeichnisses ausgewählt.

DIRECTORY = *ANY
Die BS2000-Benutzerkennungen werden unabhängig von ihrem Login-Verzeichnis ausgewählt.

DIRECTORY= *BY-POSIX-USER-DEFAULTS
Nur die BS2000-Benutzerkennungen werden ausgewählt, für die als Login-Verzeichnis der entsprechende Standardwert im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

DIRECTORY = *ROOT
Nur die BS2000-Benutzerkennungen werden ausgewählt, die als Login-Verzeichnis das Root-Verzeichnis eingetragen haben.

DIRECTORY = <posix-pathname 1..1023 without-wild>
Nur die BS2000-Benutzerkennungen mit dem angegebenen Login-Verzeichnis werden ausgewählt.

PROGRAM =
Die BS2000-Benutzerkennungen werden anhand ihres Programmnamens ausgewählt.

PROGRAM = *ANY
Die BS2000-Benutzerkennungen werden unabhängig vom Programmnamen ausgewählt.

PROGRAM = *BY-POSIX-USER-DEFAULTS
Nur die BS2000-Benutzerkennungen werden ausgewählt, für die als Programmname der entsprechende Standardwert im Benutzerkatalog des angegebenen Pubsets eingetragen ist.

PROGRAM = *SHELL
Nur die BS2000-Benutzerkennungen werden ausgewählt, die als Programmnamen *SHELL eingetragen haben.

PROGRAM = <posix-pathname 1..1023 without-wild>
Nur die BS2000-Benutzerkennungen mit dem angegebenen Programmnamen werden ausgewählt.

INFORMATION =
Umfang der Informationsausgabe.

INFORMATION = *ALL
Alle POSIX-Benutzerattribute einer BS2000-Benutzerkennung werden angezeigt (siehe Beispiel 1 auf Seite 253).

INFORMATION = *USER-LIST
Eine Liste der BS2000-Benutzerkennungen ohne POSIX-Benutzerattribute wird angezeigt (siehe Beispiel 2 auf Seite 254).

OUTPUT =
Systemdatei für die Ausgabe der Information.

OUTPUT = *SYSOUT
Die Information wird in die Systemdatei SYSOUT ausgegeben.

OUTPUT = *SYSLST(...)
Die Information wird in die Systemdatei SYSLST ausgegeben.

SYSLST-NUMBER =
Bestimmt die SYSLST-Nummer.

SYSLST-NUMBER = *STD
Bestimmt die Standard-SYSLST-Ausgabe.

SYSLST-NUMBER = <integer 1..99>
Bestimmt die angegebene SYSLST-Nummer.

LINES-PER-PAGE =
Gibt die Zeilenzahl pro Seite an.

LINES-PER-PAGE = 64
Standardmäßig werden 64 Zeilen pro Seite gedruckt.

LINES-PER-PAGE = <integer 20..255>
Die angegebene Zeilenzahl wird pro Seite gedruckt.

Ein Benutzer ohne Verwaltertätigkeit erhält bis auf zwei Ausnahmen nur Informationen über seine eigene BS2000-Benutzerkennung:

`INFORMATION=*USER-LIST,SELECT=*BY-ATTRIBUTES(USER-NUMBER=*OWN)`
Bei dieser Angabe erfährt der Benutzer auch die Identität der Benutzer, die dieselbe Benutzernummer wie er haben, wenn diese Benutzernummer ungleich der Standard-Benutzernummer ist.

`INFORMATION=*USER-LIST,SELECT=*BY-ATTRIBUTES(GROUP-NUMBER=*OWN)`
Bei dieser Angabe erfährt der Benutzer auch die Identität der Mitglieder seiner POSIX-Gruppe, wenn diese POSIX-Gruppe ungleich der Standardgruppe ist.

Bei INFORMATION=*ALL werden die Benutzernummer und die Gruppennummer gekennzeichnet, wenn der entsprechende Standardwert, der im Benutzerkatalog des angegebenen Pubsets eingetragen ist, zugewiesen ist (SHOW-Ausgabe mit „(DEFAULT)“ bzw. S-Variablen mit dem Suffix „-DEF“).

Kommando-Returncodes

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Kommando fehlerfrei ausgeführt
2	0	SRM6001	Kommando mit Warnung ausgeführt
	32	CMD2009	Fehler beim Erzeugen der Ausgabe-Variablen
	32	SRM6020	Kommando wegen eines Systemfehlers abgewiesen
	64	OPS0002	K2-Unterbrechung bei Ausgabe in S-Variable
	64	SRM6040	Kommando mit Fehlermeldung abgewiesen
	130	OPS0001	Ressourcenmangel bei Ausgabe in S-Variable
	130	SRM6030	Kommando wegen Ressourcenmangel abgewiesen

Beispiel 1

```
/SHOW-POSIX-USER-ATTRIBUTES USER-IDENTIFICATION=EXAMPLES,PUBSET=A

POSIX-USER-ATTRIBUTES --- PUBSET A                          2009-03-09 17:19:48
------------------------------------------------------------------------------
USER-ID          EXAMPLES                   PUBSET    A
USER-NUMBER           632
GROUP-NUMBER          632
COMMENT      S. Nobody, Mch-P, Tel.: 12345
DIRECTORY    /home/examples
PROGRAM      *SHELL
------------------------------------------------------------------------------
POSIX-USER-ATTRIBUTES                                              END OF DISPLAY
```

Beispiel 2

```
/SHOW-POSIX-USER-ATTRIBUTES USER-ID=*ALL,PUBSET=A,INFORMATION=*USER-LIST

POSIX-USER-LIST       --- PUBSET A                          2009-03-09 17:23:12
------------------------------------------------------------------------------
EXAMPLES  SERVICE   SYSAUDIT  SYSDUMP   SYSGEN    SYSHSMS   SYSNAC   SYSPRIV
SYSROOT   SYSSNAP   SYSSPOOL  SYSUSER   TSOS
------------------------------------------------------------------------------
POSIX-USER-LIST                                                 END OF DISPLAY
```

S-Variablen

Mit dem Operanden INFORMATION des Kommandos wird festgelegt, welche S-Variablen mit Werten versorgt werden. Folgende Angaben sind für INFORMATION möglich:

Schreibweise im Kommando	gekürzte Schreibweise in Tabelle
INFORMATION = *ALL	INF=ALL
INFORMATION = *USER-LIST	INF=U-LIST

Bitte beachten Sie, dass S-Variablen nur erzeugt werden, wenn die entsprechenden Bedingungen (siehe Spalte „Bedingung“) gültig sind.

Ausgabe-Information	**Name der S-Variablen**	**T**	**Inhalt**	**Bedingung**
Kommentar, anhand dessen die BS2000-Benutzerkennung ausgewählt wird	var(*LIST).COMMENT	S	*NONE <c-string 1..255>	INF=*ALL
Login-Verzeichnis	var(*LIST).DIR	S	<posix-pathname 1..1023>	INF=*ALL
POSIX-Gruppennummer	var(*LIST).GROUP-NUM	I	<integer 0..60002>	INF=*ALL
Default-POSIX-Gruppennummer	var(*LIST).GROUP-NUM-DEF	B	FALSE TRUE	INF=*ALL
Name des Programms, anhand dessen die BS2000-Benutzerkennung ausgewählt wird	var(*LIST).PROG	S	*SHELL <posix-pathname 1..1023>	INF=*ALL
Katalogkennung des Pubsets	var(*LIST).PUBSET	S	<catid 1..4>	INF=*ALL/ *USER-LIST
BS2000-Benutzerkennung, deren POSIX-Benutzerattribute angezeigt werden	var(*LIST).USER-ID	S	<name 1..8>	INF=*ALL
	var(*LIST).USER-ID(*LIST)	S	<name 1..8>	INF= *USER-LIST
POSIX-Benutzernummer	var(*LIST).USER-NUM	I	<integer 0..60002>	INF=*ALL
Default-POSIX-Benutzernummer	var(*LIST).USER-NUM-DEF	B	FALSE TRUE	INF=*ALL

Nähere Informationen zu S-Variablen finden Sie im Handbuch „Kommandos“ [28].

Beispiel 1

```
/declare-var var-name=pos-user-att(type=*struct),multi-elem=*list
/exec-cmd (show-posix-user-attr inf=*all),text-output=*none,structure-output=pos-user-att
/show-var pos-user-att,inf=*par(value=*c-literal)
```

```
POS-USER-ATT(*LIST).PUBSET = '1SBZ'
POS-USER-ATT(*LIST).USER-ID = 'TSOS'
POS-USER-ATT(*LIST).USER-NUM = 0
POS-USER-ATT(*LIST).USER-NUM-DEF = FALSE
POS-USER-ATT(*LIST).GROUP-NUM = 0
POS-USER-ATT(*LIST).GROUP-NUM-DEF = FALSE
POS-USER-ATT(*LIST).COMMENT = '*NONE'
POS-USER-ATT(*LIST).DIR = '/'
POS-USER-ATT(*LIST).PROG = '*SHELL'
*END-OF-VAR
/
```

Beispiel 2

```
/exec-cmd (show-posix-user-attr inf=*user-list),text-outp=*none,struct-outp=pos-user-att
/show-var pos-user-att,inf=*par(value=*c-literal)
```

```
POS-USER-ATT(*LIST).PUBSET = '1SBZ'
POS-USER-ATT(*LIST).USER-ID(*LIST) = 'TSOS'
*END-OF-VAR
/
```

SHOW-POSIX-USER-DEFAULTS
Standardwerte für POSIX-Benutzerattribute anzeigen

Anwendungsbereich: USER-ADMINISTRATION

Privilegierung: POSIX-ADMINISTRATION,
USER-ADMINISTRATION,
STD-PROCESSING

Dieses Kommando zeigt die POSIX-Standardattribute im Benutzerkatalog des angegebenen Pubsets an. Folgende Benutzer dürfen es ausführen:

- Inhaber des Privilegs POSIX-ADMINISTRATION oder USER-ADMINISTRATION für alle Pubsets.
- Gruppenverwalter der Gruppe *UNIVERSAL auf dem von ihnen verwalteten Pubset.

Format

SHOW-POSIX-USER-DEFAULTS

PUBSET = ***HOME** / ***ALL** / list-poss(20): <catid 1..4>

,**OUTPUT** = list-poss(2): ***SYSOUT** / ***SYSLST**(...)

***SYSLST**(...)

SYSLST-NUMBER = ***STD** / <integer 1..99>

,**LINE**S-PER-**PAGE** =**64** / <integer 20..255>

Operandenbeschreibung

PUBSET =
Pubset, aus dessen Benutzerkatalog die POSIX-Standardattribute angezeigt werden sollen.

PUBSET = *HOME
Die POSIX-Standardattribute werden aus dem Benutzerkatalog des Home-Pubsets angezeigt.

PUBSET = *ALL
Die POSIX-Standardattribute werden aus den Benutzerkatalogen aller Pubsets, die zum Zeitpunkt der Kommandoeingabe verfügbar sind, angezeigt.

PUBSET = list-poss(20): <catid 1..4>
Die POSIX-Standardattribute werden aus dem Benutzerkatalog des angegebenen Pubsets angezeigt.

OUTPUT =
Systemdatei für die Ausgabe der Information.

OUTPUT = *SYSOUT
Die Information wird in die Systemdatei SYSOUT ausgegeben.

OUTPUT = *SYSLST(...)
Die Information wird in die Systemdatei SYSLST ausgegeben.

SYSLST-NUMBER =
Bestimmt die SYSLST-Nummer.

SYSLST-NUMBER = *STD
Bestimmt die Standard-SYSLST-Ausgabe.

SYSLST-NUMBER = <integer 1..99>
Bestimmt die angegebene SYSLST-Nummer.

LINES-PER-PAGE =
Gibt die Zeilenzahl pro Seite an.

LINES-PER-PAGE = 64
Standardmäßig werden 64 Zeilen pro Seite gedruckt.

LINES-PER-PAGE = <integer 20..255>
Die angegebene Zeilenzahl wird pro Seite gedruckt.

Kommando-Returncodes

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Kommando fehlerfrei ausgeführt
2	0	SRM6001	Kommando mit Warnung ausgeführt
	32	CMD2009	Fehler beim Erzeugen der Ausgabe-Variablen
	32	SRM6020	Kommando wegen eines Systemfehlers abgewiesen
	64	OPS0002	K2-Unterbrechung bei Ausgabe in S-Variable
	64	SRM6040	Kommando mit Fehlermeldung abgewiesen
	130	OPS0001	Ressourcenmangel bei Ausgabe in S-Variable
	130	SRM6030	Kommando wegen Ressourcenmangel abgewiesen

Beispiel

```
/SHOW-POSIX-USER-DEFAULTS PUBSET=A
```

Ausgabe:

```
/SHOW-POSIX-USER-DEFAULTS PUBSET=A
POSIX-USER-DEFAULTS   --- PUBSET A                          2009-03-10 14:14:05
------------------------------------------------------------------------------
USER-NUMBER           200
GROUP-NUMBER            8
COMMENT      POSIX public userID
DIRECTORY    /home/usr0/gast
PROGRAM      *SHELL
------------------------------------------------------------------------------
POSIX-USER-DEFAULTS                                          END OF DISPLAY
```

S-Variablen

Ausgabe-Information	Name der S-Variablen	T	Inhalt	Bedingung
Kommentar	var(*LIST).COMMENT	S	*NONE <c-string 1..255>	
Login-Verzeichnis	var(*LIST).DIR	S	<posix-pathname 1..1023>	
POSIX-Gruppennummer	var(*LIST).GROUP-NUM	I	<integer 0..60002>	
Name des Programms	var(*LIST).PROG	S	*SHELL <posix-pathname 1..1023>	
Katalogkennung des Pubsets	var(*LIST).PUBSET	S	<catid 1..4>	
POSIX-Benutzernummer	var(*LIST).USER-NUM	I	<integer 0..60002>	

Nähere Informationen zu S-Variablen finden Sie im Handbuch „Kommandos“ [28].

Beispiel

```
/declare-var var-name=pos-user-def(type=*struct),multi-elem=*list
/exec-cmd cmd=(show-posix-user-defaults pubset=a), -
/         structure-output=pos-user-def,text-output=*none
/show-var var-name=pos-user-def
POS-USER-DEF(*LIST).PUBSET = A
POS-USER-DEF(*LIST).USER-NUM = 100
POS-USER-DEF(*LIST).GROUP-NUM = 1
POS-USER-DEF(*LIST).COMMENT = ´Systemadministration´
POS-USER-DEF(*LIST).DIR = /home/bs2000
POS-USER-DEF(*LIST).PROG = *SHELL
```

SHOW-USER-ATTRIBUTES
Informationen über die Einträge im Benutzerkatalog ausgeben

Anwendungsbereich: ACCOUNTING,
USER-ADMINISTRATION

Privilegierung: USER-ADMINISTRATION**,** SECURITY-ADMINISTRATION,
SAT-FILE-MANAGEMENT, SAT-FILE-EVALUATION,
STD-PROCESSING, HARDWARE-MAINTENANCE

Dieses Kommando zeigt die Attribute einer BS2000-Benutzerkennung an, die mit den Kommandos ADD-USER und MODIFY-USER-ATTRIBUTES vereinbart wurden.

Die POSIX-Benutzerattribute können gesondert durch das Kommando SHOW-POSIX-USER-ATTRIBUTES angezeigt werden (siehe Seite 248).

Folgende Benutzer dürfen das Kommando SHOW-USER-ATTRIBUTES ausführen:

- Inhaber des Privilegs USER-ADMINISTRATION für alle BS2000-Benutzerkennungen.
- Gruppenverwalter für die BS2000-Benutzerkennungen, die ihren Gruppen zu- und untergeordnet sind.
- jeder Benutzer für seine eigene BS2000-Benutzerkennung.

An der Benutzeroberfläche des Kommandos SHOW-USER-ATTRIBUTES ändert sich durch POSIX nichts. Es ist im Handbuch „Kommandos“ [28] beschrieben.

Das Kommando SHOW-USER-ATTRIBUTES zeigt die POSIX-Abrechnungsnummer an, falls sie festgelegt wurde. Andernfalls wird *NONE ausgegeben.

Die POSIX-Abrechnungsnummer wird für Gruppen in den Gruppenkommandos nicht berücksichtigt.

S-Variablen

Die Information zur Abrechnungsnummer für *rlogin* wird in folgender S-Variablen abgelegt:

Ausgabe-Information	Name der S-Variablen	T	Inhalt	Bedingung
Abrechnungsnummer beim POSIX-Zugang über rlogin	var(*LIST).ACCOUNT(*LIST).POSIX-RLOG-DEF	S	*NO *YES	INF=*ATTR

Nähere Informationen zu S-Variablen finden Sie im Handbuch „Kommandos“ [28].

START-POSIX-INSTALLATION
POSIX-Installationsprogramm starten

Anwendungsbereich: SYSTEM-MANAGEMENT

Privilegierung: TSOS,
POSIX-ADMINISTRATION

Dieses Kommando startet das POSIX-Installationsprogramm.

Format

START-POSIX-INSTALLATION

INPUT-INTERFACE = ***STD** / ***FHS** / ***FILE**(...)

***FILE**(...)
FILE-NAME = <filename 1..54>
,**ERROR-HANDLING** = ***PAR**AMETERS(...)
***PAR**AMETERS(...)
RETURNCODE = ***NO** / ***Y**ES
,**ABORT-ON-WARNING** = ***NO** / ***Y**ES

Operandenbeschreibung

INPUT-INTERFACE = *STD / *FHS / *FILE(...)
Angabe, ob die Installation im Dialog oder automatisiert ablaufen soll.

INPUT-INTERFACE = *STD / *FHS
Die Installation soll im Dialog (über FHS-Masken) ablaufen. Zum Ablauf der Installation siehe Abschnitt „POSIX-Installationsprogramm im Dialog“ auf Seite 126.

INPUT-INTERFACE= *FILE(...)
Die Installation soll automatisiert ablaufen unter Verwendung der angegebenen Parameterdatei. Zum Aufbau der Parameterdatei siehe Abschnitt „Automatisierter Ablauf des POSIX-Installationsprogramms“ auf Seite 136.

FILE-NAME= <filename 1..54>
Name der Parameterdatei.

ERROR-HANDLING=*PARAMETERS(...)
Legt die Reaktion in Fehlerfällen fest .

RETURN-CODE=*NO / *YES
Legt fest, ob das Kommando in Fehlerfällen einen Returncode liefern und den Spin-Off-Mechanismus auslösen soll.

RETURNCODE=*NO
In Fehlerfällen wird der Spin-Off-Mechanismus nicht ausgelöst und Kommando-Returncodes werden nicht geliefert.

RETURNCODE=*YES
In Fehlerfällen wird innerhalb von Prozeduren der Spin-Off-Mechanismus ausgelöst und das Kommando liefert Returncodes (MAINCODE = POS295x).

ABORT-ON-WARNING=*NO / *YES
Steuert das Verhalten, wenn in der Parameterdatei Fehler der Klasse 'warning' auftreten (MAINCODE = POS2956).

Bei Fehlern der Klasse 'error' wird die Verabeitung immer abgebrochen, bei Fehlern der Klasse 'note' wird sie immer mit der nächsten Zeile fortgesetzt.

ABORT-ON-WARNING=*NO
Die Verarbeitung der Parameterdatei wird bei Fehlern der Klasse 'warning' mit der nächsten Zeile fortgesetzt.

ABORT-ON-WARNING=*YES
Die Verarbeitung der Parameterdatei wird bei Fehlern der Klasse 'warning' abgebrochen.

Kommando-Returncodes

Die Ausgabe von Returncodes erfolgt nur bei Angabe von
ERROR-HANDLING=*PARAMETERS(RETURNCODE=*YES)

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Ohne Fehler
	64	POS2950	Ungültiger Prozedurparameter.
	64	POS2951	Parameterdatei wurde nicht gefunden oder ist nicht zugreifbar.
	64	POS2952	Die Benutzerkennung ist nicht zur Ausführung des POSIX-Installationsprogramms berechtigt.
	64	POS2953	Eine andere Instanz des POSIX-Installationsprogramms wird gerade ausgeführt.
	64	POS2954	Installationsprogramm kann nicht geladen werden.
	64	POS2955	Schwerer Fehler im Installationsprogramm.
znr	64	POS2956	Fehler in der Parameterdatei. Die Nummer der Zeile (znr), bei oder nach der der Fehler aufgetreten ist, kann dem SC2 entnommen werden. Bei Batch-Installation im Online-Modus sind detailliertere Informationen in der POSIX-Datei */var/sadm/pkg/insterr* zu finden.
	64	POS2957	Zeitablauf beim Warten auf POSIX-Neustart. Datei ist gesperrt.

Protokollierung von Fehlern in der Parameterdatei (MAINCODE = POS2956):

Bei Batch-Installation im Online-Modus, d.h. nicht bei einer Erstinstallation oder bei einer Dateisystemerweiterung im Offline-Modus, werden folgende Informationen in die POSIX-Datei `/var/sadm/pkg/insterr` geschrieben:

– Name der Parameterdatei
– Datum und Uhrzeit der Installation

Bei Hinweisen / Warnungen / Fehlern zusätzlich:

– fehlerhafte Zeile in der Parameterdatei
– Fehlerklasse (note, warning, error) und Fehlertext

Beispiele:

```
input file : :FR01:$TSOS.POSIX-INSTALL.FS.TMP.TEST
time       : Wed Jan 28 13:17:40 2009
- line   3 : a;$SYSROOT.FS.TMP.TEST;8192;Y;Y;/tmp/test;N;;y;N
  note     : Line 3: BS2000 file already existing, file size may not be changed
  note     : Line 3: file system size of existing filesystem will be used
input file : :V70A:$TSOS.INSTALL.POSIX-SOCKENS
time       : Wed Jan 28 13:27:25 2009
- line   4 : POSIX-SOCKENS;Y
  warning  : IMON-GPN: installation unit not found in SCI
input file : :FR01:$TSOS.POSIX-INSTALL.FIRST
time       : Wed Jan 28 13:38:51 2009
- line 1   : [FirstInstallation]
  error    : POSIX subsystem is available
```

START-POSIX-SHELL
POSIX-Shell zur Verfügung stellen

Anwendungsbereich: PROCEDURE, UTILITIES

Privilegierung: STD-PROCESSING

Dieses Kommando startet die POSIX-Shell. Nach erfolgreichem Zugang zur POSIX-Shell kann der Benutzer POSIX-Kommandos eingeben (siehe Abschnitt „Kommandoumfang der POSIX-Shell“ auf Seite 270) und POSIX-Handbuch „Kommandos“ [1]).

Beendet wird die POSIX-Shell mit dem POSIX-Kommando *exit*.

Format

START-POSIX-SHELL	Kurzname: **SHELL, START-SHELL, POSIX-SHELL**
VERSION = ***STD** / <product-version without-man-corr>	

Oprandenbeschreibung

VERSION = *STD / <product-version without-man-corr>
Versionsnummer des aufzurufenden Programms (hier der POSIX-Shell).
Voreingestellt ist *STD, d.h. es wird die aktuell verfügbare Version aufgerufen.

Kommando-Returncodes

(SC2)	SC1	Maincode	Bedeutung
	0	CMD0001	Ohne Fehler

Beispiel

Siehe Abschnitt „Beispielsitzung“ auf Seite 74.

10 Anhang

Im Anhang finden Sie:

- die Privilegien bei POSIX
- den Kommandoumfang der POSIX-Shell
- die Dämonen von POSIX
- die Dateiverzeichnisse, die bei einer Erstinstallation angelegt werden
- die Gerätedateien, die bei einer Erstinstallation angelegt werden
- die Verwaltungsdateien, die bei einer Erstinstallation angelegt werden
- Tuning-Maßnahmen
- POSIX-Protokolldateien
- Lokale Uhrzeit in POSIX

10.1 Privilegien bei POSIX

Die folgende Tabelle zeigt, wer welche Aufgaben in POSIX ausführen darf:

Privileg	**Berechtigung für**
Operator (Privileg **Operating**)	POSIX starten: /START-SUBSYSTEM SUBSYSTEM-NAME=POSIX
	POSIX beenden: /STOP-SUBSYSTEM SUBSYSTEM-NAME=POSIX
Subsystem-Verwalter (Privileg **SUBSYSTEM-MANAGEMENT**)	POSIX-Status anzeigen: /SHOW-POSIX-STATUS
	POSIX starten: /START-SUBSYSTEM SUBSYSTEM-NAME=POSIX
	POSIX beenden: /STOP-SUBSYSTEM SUBSYSTEM-NAME=POSIX
Sicherheitsbeauftragter (Privileg **SECURITY-ADMINISTRATION**)	BS2000-Kennungen (außer SYSROOT) das Privileg POSIX-ADMINISTRATION verleihen oder entziehen: /SET-PRIVILEGE /RESET-PRIVILEGE

Privileg	**Berechtigung für**
BS2000-Systemverwalter (Privileg **USER-ADMINISTRATION**, aber **keine** Root-Berechtigung)	Werte der Tuningparameter in der POSIX-Informationsdatei SYSSSI.POSIX-BC.*version* ändern
	Neue POSIX-Benutzer verwalten: /ADD-USER und weitere Maßnahmen
	BS2000-Benutzerkennungen individuelle POSIX-Benutzer-attribute zuordnen: /MODIFY-POSIX-USER-ATTRIBUTES
	POSIX-Benutzerattribute verwalten: /MODIFY-POSIX-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES
	Standardwerte für POSIX-Benutzerattribute verwalten: /MODIFY-POSIX-USER-DEFAULTS /SHOW-POSIX-USER-DEFAULTS
	Zugangsberechtigung für den Benutzer eines fernen Rechners verwalten: /SET-LOGON-PROTECTION /MODIFY-LOGON-PROTECTION /SHOW-LOGON-PROTECTION
	Abrechnungsnummer für den Systemzugang über einen fernen Rechner verwalten: /ADD-USER /MODIFY-USER-ATTRIBUTES /SHOW-USER-ATTRIBUTES
	POSIX-Gruppen/Benutzer verwalten: Benutzerattribut GROUP-NUMBER
BS2000-Systemverwalter (Privileg **USER-ADMINISTRATION** und **zusätzlich** Root-Berechtigung)	Alle Aufgaben, die ein BS2000-Systemverwalter ohne Root-Berechtigung durchführen darf, und zusätzlich:
	POSIX installieren: POSIX-Installationsprogramm
	POSIX-Dateisysteme einrichten, ändern und löschen: POSIX-Installationsprogramm

Privileg	**Berechtigung für**
POSIX-Verwalter (Privileg **POSIX-ADMINISTRATION** und zusätzlich Root-Berechtigung)	Neue POSIX-Benutzer verwalten: /ADD-USER und weitere Maßnahmen
	POSIX-Benutzer löschen: /MODIFY-POSIX-USER-ATTRIBUTES
	POSIX-Dateisysteme einrichten, ändern, löschen: POSIX-Installationsprogramm
	POSIX-Dateisysteme ein- und aushängen: *mount* und *mountall*; *umount* und *umountall*
	POSIX-Benutzerattribute verwalten: /MODIFY-POSIX-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES
	Standardwerte für POSIX-Benutzerattribute verwalten: /MODIFY-POSIX-USER-DEFAULTS /SHOW-POSIX-USER-DEFAULTS
	BS2000- und POSIX-Gruppen verwalten: Benutzerattribut GROUP-NUMBER, Datei */etc/group*
Root-Berechtigung (**Benutzernummer 0, Gruppennummer 0**)	POSIX-Gruppen in POSIX verwalten: Datei */etc/group*
	POSIX-Benutzer löschen: *rmdir*
BS2000-Gruppenverwalter (Gruppe ***UNIVERSAL**)	POSIX-Benutzerattribute verwalten (mit Einschränkungen): /MODIFY-POSIX-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES
	Standardwerte für POSIX-Benutzerattribute verwalten (mit Einschränkungen): /MODIFY-POSIX-USER-DEFAULTS /SHOW-POSIX-USER-DEFAULTS
	Zugangsberechtigung für den Benutzer eines fernen Rechners verwalten (mit Einschränkungen): /SET-LOGON-PROTECTION /MODIFY-LOGON-PROTECTION /SHOW-LOGON-PROTECTION
	Abrechnungsnummer für den Systemzugang über einen fernen Rechner verwalten (mit Einschränkungen): /ADD-USER /MODIFY-USER-ATTRIBUTES /SHOW-USER-ATTRIBUTES
	POSIX-Gruppen/Benutzer verwalten: Benutzerattribut GROUP-NUMBER

Privileg	**Berechtigung für**
Nichtprivilegierter POSIX-Benutzer (Privileg **STD-PROCESSING**)	Informationen über die Einträge im Benutzerkatalog für die eigenen Benutzerkennungen ausgeben: /SHOW-USER-ATTRIBUTES /SHOW-POSIX-USER-ATTRIBUTES

10.2 Kommandoumfang der POSIX-Shell

Die POSIX-Shell setzt sich zusammen aus der Basis-Shell (POSIX-BC) und der erweiterten Shell (POSIX-SH). Sie beinhaltet die in folgender Tabelle aufgeführten POSIX-Kommandos.

Die Einträge in der Spalte *Typ* beschreiben den Typ des Kommandos:

bin eigener Modul

blt Builtin in Shell

scr Skript

Die Spalte *LFS* beschreibt, ob die Kommandos große POSIX-Dateien bearbeiten können:

A (large file **a**ware): arbeitet korrekt mit großen Dateien

S (large file **s**afe): erkennt große Dateien, weist die Bearbeitung jedoch definiert zurück

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
alias	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Alias-Namen definieren oder anzeigen	
ar	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Bibliotheken verwalten	S
asa	/usr/bin	bin	SINLIB.POSIX-SH	Steuerzeichen für die Positionierung umsetzen	
at	/usr/bin	bin	SINLIB.POSIX-SH	Kommandos zu einem späteren Zeitpunkt ausführen	
awk	/usr/bin	bin	SINLIB.POSIX-SH	Programmierbare Bearbeitung von Textdateien	A
basename	/usr/bin	scr	SINLIB.POSIX-BC.SHELL	Dateinamen vom Pfad trennen	
batch	/usr/bin	scr	SINLIB.POSIX-BC.SHELL	Kommandos zu einer späteren Zeit ausführen	
bc	/usr/bin	bin	SINLIB.POSIX-SH	Arithmetische Sprache	
bg	-	blt	SINLIB.POSIX-BC.SHELL	Jobs in den Hintergrund schicken	
bs2cmd	-	blt	SINLIB.POSIX-BC.SHELL	BS2000-Kommando ausführen	
bs2cp	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	BS2000-Dateien kopieren (BS2000)	A
bs2do	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	BS2000-Prozeduren aus POSIX aufrufen	
bs2file	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateiattribute für BS2000-Dateien festlegen (BS2000)	
bs2lp	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Dateien ausdrucken (BS2000)	
bs2pkey	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	P-Tasten belegen (BS2000)	
cal	/usr/bin	bin	SINLIB.POSIX-SH	Kalender ausgeben	
cancel	/usr/bin	bin	SINLIB.POSIX-SH	Druckaufträge löschen	
cat	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateien aneinanderfügen und ausgeben	A
cd	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Aktuelles Dateiverzeichnis wechseln	A
chgrp	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Gruppennummer einer Datei ändern	A
chmod	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Zugriffsrechte ändern	A
chown	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Eigentümer einer Datei ändern	A
cksum	/usr/bin	bin	SINLIB.POSIX-SH	Prüfsummen und Größen von Dateien schreiben	A
cmp	/usr/bin	bin	SINLIB.POSIX-SH	Dateien zeichenweise vergleichen	A
comm	/usr/bin	bin	SINLIB.POSIX-SH	Gleiche Zeilen in zwei sortierten Dateien suchen	S
command	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	einfaches Kommando ausführen	
compress	/usr/bin	bin	SINLIB.POSIX-SH	Dateien komprimieren	A
cp	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateien kopieren	A
cp	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateien kopieren	A
cpio	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Dateien und Dateiverzeichnisse ein- und auslagern	A
crontab	/usr/bin	bin	SINLIB.POSIX-SH	Kommandos regelmäßig zu bestimmten Zeitpunkten ausführen	
csplit	/usr/bin	bin	SINLIB.POSIX-SH	Datei nach bestimmten Kriterien unterteilen	S
cut	/usr/bin	bin	SINLIB.POSIX-SH	Bytes, Zeichen oder Felder aus den Zeilen einer Datei ausschneiden	S
date	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Datum und Uhrzeit ausgeben	

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
dd	/sbin	bin	SINLIB.POSIX-SH	Dateien kopieren und konvertieren	S
debug	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Testen von POSIX-Programmen	
df	/sbin	bin	SINLIB.POSIX-BC.SHELL	Anzahl der freien und belegten Plattenblöcke ausgeben	A
diff	/usr/bin	bin	SINLIB.POSIX-SH	Dateien zeilenweise vergleichen	A
dirname	/usr/bin	scr	SINLIB.POSIX-BC.SHELL	Pfad-Präfix vom Dateinamen trennen	
du	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Belegten Speicherplatz ausgeben	A
dumpfs	/sbin	bin	SINLIB.POSIX-BC.ROOT	interne Dateisystem-Information ausgeben	
echo	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Aufruf-Argumente ausgeben	
ed	/sbin	bin	SINLIB.POSIX-SH	Zeilenorientierter Editor im Dialogbetrieb	
edt	-	blt	SINLIB.POSIX-BC.SHELL	BS2000-Dateibearbeiter EDT aufrufen	S
edtu	-	blt	SINLIB.POSIX-BC.SHELL	BS2000-Dateibearbeiter EDT aufrufen	S
egrep	/usr/bin	bin	SINLIB.POSIX-SH	Muster suchen	S
env	/usr/bin	bin	SINLIB.POSIX-SH	Umgebung bei Ausführung von Kommandos ändern	
eval	-	blt	SINLIB.POSIX-BC.SHELL	Aufrufargumente bearbeiten und als Kommando ausführen	
ex	/usr/bin	bin	SINLIB.POSIX-SH	Zeilenorientierter Editor	
exec	-	blt	SINLIB.POSIX-BC.SHELL	Die aktuelle Shell überlagern	
exit	-	blt	SINLIB.POSIX-BC.SHELL	Shell-Prozedur beenden	
expand	/usr/bin	bin	SINLIB.POSIX-SH	Tabulatorzeichen in Leerzeichen umwandeln	S
export	-	blt	SINLIB.POSIX-BC.SHELL	Shell-Variablen exportieren	
expr	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Ausdrücke auswerten	
expr	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Ausdrücke auswerten	
false	/usr/bin	alias+scr	SINLIB.POSIX-BC.SHELL	Endestatus ungleich 0 zurückgeben	
fc	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Zugriff auf die History-Datei	
fg	-	blt	SINLIB.POSIX-BC.SHELL	Jobs in den Vordergrund bringen	
fgrep	/usr/bin	bin	SINLIB.POSIX-SH	Zeichenketten suchen	S
file	/usr/bin	bin	SINLIB.POSIX-SH	Art einer Datei bestimmen	A
find	/usr/bin	bin	SINLIB.POSIX-SH	Dateiverzeichnisse durchsuchen	A
fold	/usr/bin	bin	SINLIB.POSIX-SH	Lange Zeilen zerlegen	S
fsck	/sbin	bin	SINLIB.POSIX-BC.ROOT	Konsistenzprüfung des Dateisystems und Korrektur im Benutzer-Dialog	
fsexpand	/sbin	bin	SINLIB.POSIX-BC.ROOT	Existierende Dateisysteme vergrößern	A
ftyp	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Bearbeitungsart für Dateien festlegen (BS2000)	
fuser	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Dateinutzer anzeigen	
gencat	/usr/bin	bin	SINLIB.POSIX-SH	Binär codierten Meldungskatalog erzeugen	
genso	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Shared Object erzeugen	

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
getconf	/usr/bin	bin	SINLIB.POSIX-SH	Konfigurationswerte abrufen	A
getopts	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Argumente einer Prozedur nach Optionen durchsuchen	
grep	/sbin	bin	SINLIB.POSIX-BC.SHELL	Muster suchen	A
hash	/usr/bin	alias+scr	SINLIB.POSIX-BC.SHELL	Hash-Tabelle der Shell bearbeiten	
hd	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Dateiinhalt hexadezimal ausgeben	A
head	/usr/bin	bin	SINLIB.POSIX-SH	Anfang einer Datei ausgeben	A
iconv	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Code konvertieren	A
id	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Benutzer-Identifikation ausgeben	
inetd	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Dämon für Internet-Dienste	
info	/sbin	bin	SINLIB.POSIX-BC.ROOT	Online-Diagnosetool	
ipcrm	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Einrichtungen zur Interprozess-Kommunikation löschen	
ipcs	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Zustand von Interprozess-Kommunikationseinrichtungen anzeigen	
jobs	-	blt	SINLIB.POSIX-BC.SHELL	Auftragsinformationen ausgeben	
join	/usr/bin	bin	SINLIB.POSIX-SH	Zwei Dateien nach Vergleichsfeldern verbinden	A
kill	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Signale an Prozesse senden	
last	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Zuletzt angemeldete Benutzer anzeigen	
let	-	blt	SINLIB.POSIX-BC.SHELL	Arithmetische Berechnungen	
lex	/usr/bin	bin	SINLIB.POSIX-SH	Scanner erstellen	
ln	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Verweis auf eine Datei eintragen	A
locale	/usr/bin	bin	SINLIB.POSIX-SH	Informationen über die internationale Umgebung abrufen	
localedef	/usr/bin	bin	SINLIB.POSIX-SH	Internationale Umgebung definieren	
logger	/usr/bin	bin	SINLIB.POSIX-SH	Meldungen protokollieren	
logname	/usr/bin	bin	SINLIB.POSIX-SH	Login-Kennung abfragen	
logrotate	/usr/sbin	scr	SINLIB.POSIX-BC.ROOT	Wechsel der Protokolldateien des syslog-Dämonen	S
lp	/usr/bin	bin	SINLIB.POSIX-SH	Dateien ausdrucken	
lpstat	/usr/bin	bin	SINLIB.POSIX-SH	Informationen über Druckaufträge ausgeben	
ls	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Informationen über Dateiverzeichnisse und Dateien ausgeben	A
mailx	/usr/bin	bin	SINLIB.POSIX-SH	Nachrichten interaktiv bearbeiten	
make	/usr/bin	bin	SINLIB.POSIX-SH	Gruppen von Dateien verwalten	
man	/usr/bin	scr	SINLIB.POSIX-SH	Online-Dokumentation nutzen	
mesg	/usr/bin	bin	SINLIB.POSIX-SH	Nachrichtenempfang verbieten oder erlauben	
mkdir	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateiverzeichnis erzeugen	

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
mkfifo	/usr/bin	bin	SINLIB.POSIX-SH	FIFO erstellen	A
mkfs	/sbin	bin	SINLIB.POSIX-BC.ROOT	Dateisystem erstellen	
mknod	/sbin	bin	SINLIB.POSIX-BC.ROOT	Gerätedatei anlegen	A
more	/usr/bin	bin	SINLIB.POSIX-SH	Bildschirmausgabe steuern	A
mount	/sbin	bin	SINLIB.POSIX-BC.ROOT	Dateisysteme und ferne Ressourcen einhängen	
mountall	/sbin	scr	SINLIB.POSIX-BC.ROOT	Mehrere Dateisysteme einhängen	
mv	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateien versetzen oder umbenennen	A
newgrp	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Gruppenzugehörigkeit ändern	
nice	/usr/bin	bin	SINLIB.POSIX-SH	Priorität von Kommandos ändern	
nl	/usr/bin	bin	SINLIB.POSIX-SH	Textzeilen nummerieren	S
nm	/usr/bin	bin	SINLIB.POSIX-SH	Symboltabelle einer Objektdatei ausgeben	
nohup	/usr/bin	bin	SINLIB.POSIX-SH	Kommando ausführen und dabei Signale ignorieren	
od	/usr/bin	bin	SINLIB.POSIX-SH	Inhalt einer Datei oktal ausgeben	S
paste	/usr/bin	bin	SINLIB.POSIX-SH	Zeilen zusammenfügen	S
patch	/usr/bin	bin	SINLIB.POSIX-SH	Differenzliste anwenden	
pathchk	/usr/bin	bin	SINLIB.POSIX-SH	Pfadnamen überprüfen	
pax	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Bearbeitung portierbarer Archive	A
pdbl	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Privaten POSIX-Lader verwalten	
ping	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Senden von Echo-Request-Paketen an Netzwerkkomponenten	
pkginfo	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Informationen über Software-Pakete anzeigen	
posdbl	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Verwalten des POSIX-Laders	
pr	/usr/bin	bin	SINLIB.POSIX-SH	Dateien formatieren und auf Standard-Ausgabe ausgeben	
print	-	blt	SINLIB.POSIX-BC.SHELL	Ausgabemechanismus ähnlich echo	
printf	/usr/bin	blt+bin	SINLIB.POSIX-SH	Formatierte Ausgabe	
ps	/sbin	bin	SINLIB.POSIX-BC.SHELL	Prozessdaten abfragen	
pwd	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Pfadnamen des aktuellen Dateiverzeichnisses ausgeben	
rcp	/usr/bin	bin	SINLIB.POSIX-BC.INET	Datei von oder zu einem fernen Rechner kopieren	A
read	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Argumente von der Standard-Eingabe lesen, Shell-Variablen zuweisen	
readonly	-	blt	SINLIB.POSIX-BC.SHELL	Shell-Variablen schützen	
renice	/usr/bin	bin	SINLIB.POSIX-SH	Priorität laufender Prozesse ändern	
rm	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateien löschen	
rmdir	/sbin	blt+bin	SINLIB.POSIX-BC.SHELL	Dateiverzeichnisse löschen	A
rmpart	/sbin	bin	SINLIB.POSIX-BC.ROOT	Partition entfernen	

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
rsh	/usr/bin	bin	SINLIB.POSIX-BC.INET	Shell-Kommando am fernen Rechner ausführen	
sed	/usr/bin	bin	SINLIB.POSIX-SH	Editor im Prozedurbetrieb	
set	-	blt	SINLIB.POSIX-BC.SHELL	Parameter oder Optionen setzen, Variablen ausgeben	
sh	/sbin	bin	SINLIB.POSIX-BC.SHELL	Kommandointerpreter und Programmiersprache der POSIX-Shell	A
shift	-	blt	SINLIB.POSIX-BC.SHELL	Werte der Stellungsparameter nach links verschieben	
show_pubset_ export	/sbin	scr	SINLIB.POSIX-BC.ROOT	vom EXPORT-PUBSET betroffene Dateisysteme anzeigen	
sleep	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Prozesse zeitweise stillegen	
sort	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Dateien sortieren und/oder mischen	
split	/usr/bin	bin	SINLIB.POSIX-SH	Datei auf mehrere Dateien verteilen	A
start_bs2fsd	/sbin	scr	SINLIB.POSIX-BC.ROOT	Kopierdämönen starten	
strings	/usr/bin	bin	SINLIB.POSIX-SH	Druckbare Zeichenketten in Objekt- oder Binärdateien suchen	S
stty	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Eigenschaften einer Datensichtstation ausgeben oder ändern	
su	/sbin	bin	SINLIB.POSIX-BC.ROOT	Benutzerkennung wechseln	
sum	/usr/bin	bin	SINLIB.POSIX-SH	Prüfsumme einer Datei berechnen	A
sync	/sbin	bin	SINLIB.POSIX-BC.ROOT	Systempuffer zurückschreiben	
tabs	/usr/bin	bin	SINLIB.POSIX-SH	Tabulatorstops setzen	
tail	/usr/bin	bin	SINLIB.POSIX-SH	Den letzten Teil einer Datei ausgeben	A
talk	/usr/bin	bin	SINLIB.POSIX-SH	Dialog mit anderem Benutzer führen	
tar	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Archivieren von Dateien	A
tee	/usr/bin	bin	SINLIB.POSIX-SH	Pipes zusammenfügen und Eingabe kopieren	
test	/usr/bin	blt-scr	SINLIB.POSIX-BC.SHELL	Bedingungen prüfen	
time	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Laufzeit eines Kommandos messen	
times	-	blt	SINLIB.POSIX-BC.SHELL	Gesamtlaufzeit der bisher gestarteten Prozesse ausgeben	
touch	/usr/bin	blt+bin	SINLIB.POSIX-BC.SHELL	Änderungs- und Zugriffszeiten aktualisieren	A
tput	/usr/bin	bin	SINLIB.POSIX-SH	Datensichtstation initialisieren oder Datenbank terminfo abfragen	
tr	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Zeichen ersetzen oder löschen	A
trap	-	blt	SINLIB.POSIX-BC.SHELL	Signalbehandlung ändern	
true	/usr/bin	alias+scr	SINLIB.POSIX-BC.SHELL	Endestatus 0 zurückgeben	
tsort	/usr/bin	bin	SINLIB.POSIX-SH	Topologisch sortieren	
tty	/usr/bin	bin	SINLIB.POSIX-SH	Pfadnamen der aktuellen Datensichtstation ausgeben	
type	/usr/bin	alias+scr	SINLIB.POSIX-BC.SHELL	Typ eines Kommandos abfragen	

Name	Ort	Typ	Auslieferung	Beschreibung	LFS
typeset	-	blt	SINLIB.POSIX-BC.SHELL	Attribute für Shell-Variablen setzen	
ulimit	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Datei-Größe für das Schreiben begrenzen oder Grenzwert abfragen	A
umask	/usr/bin	blt-scr	SINLIB.POSIX-BC.SHELL	Standard-Vergabe der Zugriffsrechte ausgeben oder ändern	
umount	/sbin	bin	SINLIB.POSIX-BC.ROOT	Dateisysteme und ferne Ressourcen aushängen	
umountall	/sbin	scr	SINLIB.POSIX-BC.ROOT	Aushängen mehrerer Dateisysteme	
unalias	/usr/bin	blt-scr	SINLIB.POSIX-BC.SHELL	Variablen aus der alias-Tabelle löschen	
uname	/sbin	bin	SINLIB.POSIX-BC.SHELL	Basisdaten über das aktuelle Betriebssystem ausgeben	
uncompress	/usr/bin	bin	SINLIB.POSIX-SH	Komprimierte Dateien dekomprimieren	A
unexpand	/usr/bin	bin	SINLIB.POSIX-SH	Leerzeichen in Tabulatorzeichen umwandeln	S
uniq	/usr/bin	bin	SINLIB.POSIX-BC.SHELL	Mehrfache Zeilen suchen	
uudecode	/usr/bin	bin	SINLIB.POSIX-SH	Datei nach der Übertragung per mailx decodieren	
uuencode	/usr/bin	bin	SINLIB.POSIX-SH	Datei für die Übertragung per mailx codieren	
uuname	/usr/bin	bin	SINLIB.POSIX-SH	Namen des Systems auflisten	
usp	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Dynamisches Setzen von POSIX-Steuerparametern	
vi	/usr/bin	bin	SINLIB.POSIX-SH	Bildschirmorientierter Editor	
wait	/usr/bin	blt+scr	SINLIB.POSIX-BC.SHELL	Auf die Beendigung von Hintergrundprozessen warten	
wc	/usr/bin	bin	SINLIB.POSIX-SH	Wörter, Zeichen und Zeilen zählen	A
whence	-	blt	SINLIB.POSIX-BC.SHELL	Kommando-Lokalisation	
who	/sbin	bin	SINLIB.POSIX-SH	Aktive Benutzerkennungen anzeigen	
write	/usr/bin	bin	SINLIB.POSIX-SH	Nachricht an einen Benutzer senden	
xargs	/usr/bin	bin	SINLIB.POSIX-SH	Argumentliste(n) aufbauen und Kommando ausführen	
yacc	/usr/bin	bin	SINLIB.POSIX-SH	Parser erstellen	
zcat	/usr/bin	bin	SINLIB.POSIX-SH	Komprimierte Dateien ausgeben	A

10.3 Dämonen von POSIX

Die folgende Tabelle enthält eine Liste aller Dämonen, die mit POSIX ausgeliefert werden.

Name	Ort	Typ	Auslieferung	Beschreibung
bs2fsd	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Kopierdämon für bs2fs-Dateisysteme
cron	/usr/sbin	bin	SINLIB.POSIX-SH	Dämon zum Ausführen von Kommandos zu bestimmten Zeiten
fsmond	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Dämon zur Dateisystemüberwachung
in.rlogind	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Server für rlogin (Remote Login), unterstützt IPv4 und IPv6
in.rshd	/usr/sbin	bin	SINLIB.POSIX-BC.INET	Server für rsh (Remote Shell), unterstützt IPv4 und IPv6
in.talkd	/usr/sbin	bin	SINLIB.POSIX-SH	Server für talk (Dialog mit einem anderen Benutzer)
in.telnetd	/usr/sbin	bin	SINLIB.POSIX-BC.INET	Server für telnet (Pseudo-Terminalprotokoll), unterstützt IPv4 und IPv6
inetd	/usr/bin	bin	SINLIB.POSIX-BC.ROOT	Dämon für Internet-Dienste, unterstützt IPv4 und IPv6
rpcbind	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Dämon für RPC-Protokoll
shmd	/usr/sbin	bin	SINLIB.POSIX-BC.ROOT	Shared Memory Dämon
syslogd	/usr/sbin	bin	SINLIB.POSIX.BC.ROOT	syslog-Dämon

10.4 Dateiverzeichnisse, die bei einer Erstinstallation angelegt werden

Die folgenden Dateiverzeichnisse werden bei einer Erstinstallation angelegt:

/dev mit den Unterverzeichnissen:
/dsk, /fd, /pts, /rdsk, /sad, /sf und /term

/etc mit den Unterverzeichnissen:
/default, /dfs, /fs, /inet, /init.d, /net, /net/ticlts, /net/ticots, /net/ticotsord, /sm, /sm.d, /.products und /.products/.legit

/home

/lost+found

/proc

/sbin

/tmp

/usr mit den Unterverzeichnissen:
/bin, /lib, /lib/iconv, /lib/lex, /lib/nfs, /sbin und /ucb

/var mit den Unterverzeichnissen:
/adm, /lp, /mail, /preserve, /spool, /spool/lp/temp, /sadm/pkg und /tmp

Diese Dateiverzeichnisse, die das POSIX-Installationsprogramm einrichtet, darf der POSIX-Verwalter weder editieren noch verändern!

10.5 Gerätedateien, die bei einer Erstinstallation angelegt werden

Die folgenden Gerätedateien werden bei einer Erstinstallation angelegt:

/dev/console

/dev/kmem

/dev/log

/dev/loop

/dev/null

/dev/dsk/0s0

/dev/rdsk/0s0

/dev/root

/dev/rroot

/dev/ptmx

/dev/sf/mmm 000 <= mmm < 1024

/dev/ticlts

/dev/ticots

/dev/ticotsord

/dev/tty

/dev/zero

/dev/sad/admin

/dev/sad/user

/dev/ptmx

/dev/pts/mmm 000 <= mmm < 1024

/dev/term/mmm 000 <= mmm < 1024

Diese Gerätedateien, die das POSIX-Installationsprogramm einrichtet, darf der POSIX-Verwalter weder editieren noch verändern!

10.6 Verwaltungsdateien, die bei einer Erstinstallation angelegt werden

Die folgenden Verwaltungsdateien werden bei einer Erstinstallation angelegt:

/etc/dfs/dfstab

/etc/dfs/fstypes

/etc/dfs/sharetab

/etc/group

/etc/inetd.conf

/etc/mnttab

/etc/net/ticlts/hosts

/etc/net/ticlts/services

/etc/net/ticots/hosts

/etc/net/ticots/services

/etc/net/ticotsord/hosts

/etc/net/ticotsord/services

/etc/netconfig

/etc/partitions

/etc/print

/etc/protocols

/etc/profile

/etc/services

/etc/syslog.conf

/etc/termcap

/etc/TIMEZONE

/etc/vfstab

Die Tabellen, die in */etc* eingerichtet werden, stimmen in Bezug auf Inhalt und Bedeutung mit den entsprechenden Tabellen von Reliant UNIX V5.45 überein.

10.7 Tuning-Maßnahmen

Die folgende Tabelle enthält Maßnahmen, die zur Steigerung der POSIX-Performance empfohlen werden.

Maßnahme	**Bemerkungen**
Verwendung des POSIX-Laders *posdbl*	Empfohlen wird ein Memory-Pool mit 30 MB (Parameter DBLPOOL in der Informationsdatei) Hinweis: Die PAGING AREA muss diesen zusätzlichen Adressraum aufnehmen können. Für die Programmproduktion: Explizites Einbinden des Compiler-Aufrufs in posdbl mittels posdbl -b /usr/bin/c89 oder posdbl -b /usr/bin/cc
Einsatz aktueller Subsysteme	Für die Shell-Kommandos und die Ausführung eigener Programme sollten folgende Subsysteme gestartet sein: Für CRTE: CRTEC, CRTEPART Für CRTE-BASYS: CRTEBASY Für die Nutzung von EDT: EDT, EDTU und EDTCON Für die Programmproduktion: BINDER, PMLOG, CPP
Nutzung von DAB	Folgende Dateien sollten im Cache gehalten werden: SYS.AIDIT0 (ca. 30 PAM-Seiten groß): vollständig cachen
Verwendung des privaten POSIX-Laders *pdbl*	siehe POSIX-Handbuch „Kommandos“ [1]
Erhöhung von Steuerparametern in der POSIX-Informationsdatei (siehe Seite 145)	Bei hoher I/O-Last: HDSTNI: Anzahl der Servertasks zur Durchführung asynchroner I/Os NPBUF: Maximale Anzahl der I/O-Puffer für physikalische I/Os. Dieser Wert sollte mindestens 4 * HDSTNI sein.

10.8 POSIX-Protokolldateien

Die Protokollierung von Meldungen der POSIX-Programme und -Dämonen erfolgt durch Aufruf des syslog-Dämonen (*syslogd*) über die CRTE-Schnittstelle *syslog()*. Der syslog-Dämon protokolliert die Meldungen in eine oder mehrere Protokolldateien. Das Verhalten des syslog-Dämonen kann über die Datei */etc/syslog.conf* gesteuert werden. Standardmäßig protokolliert der syslog-Dämon alle Meldungen in die Datei */var/adm/syslog*.

Der syslog-Dämon entspricht der Version 1.167 der FreeBSD-Implementierung (siehe *http://www.freebsd.org*). Neben Anpassungen an die Randbedingungen von POSIX/BS2000 wurden folgende Funktionen entfernt:

– Empfangen von Meldungen von fernen Rechnern

– Senden von Meldungen zu fernen Rechnern

– Schreiben von Meldungen auf Terminals

– Schreiben von Meldungen auf Kommando-Pipes

Die lizenzrechtlichen Bedingungen finden Sie im Abschnitt „Lizenzrechtliche Bedingungen für die FreeBSD-Implementierung“ auf Seite 283.

Falls auf dem System auch POSIX-Programme mit veraltetem Laufzeitsystem (CRTE) zum Ablauf kommen, werden deren Meldungen in die Datei */var/adm/messages* protokolliert. Es sollte in diesem Fall vermieden werden, die Datei */var/adm/messages* dem syslog-Dämon als Protokolldatei zuzuweisen, weil das Kommando *logrotate* dann nicht korrekt funktioniert

Bei jedem Neustart des POSIX-Subsystems werden neue Protokolldateien angelegt, nachdem die vorhergehenden Dateien umbenannt wurden (Suffix ".0" bis ".3"). Der Wechsel der Protokolldateien kann auch im laufenden Betrieb mit dem Kommando *logrotate* (siehe POSIX-Handbuch „Kommandos“ [1]) erfolgen.

10.8.1 Lizenzrechtliche Bedingungen für die FreeBSD-Implementierung

syslogd – syslog-Dämon zur Protokollierung von Systemnachrichten

Der *syslog*-Dämon liest und protokolliert Nachrichten in Protokolldateien gemäß den Definitionen in seiner Konfigurationsdatei (*/etc/syslog.conf*)

Syntax **syslogd** [**-Cduv**] [**-f** config_file] [**-l** [mode:]path] [**-m** mark_interval]

-C Protokolldateien, die noch nicht existieren, werden erzeugt (mit den Rechten 0600).

Ist diese Option nicht angegeben, müssen die Protokolldateien angelegt werden, bevor der *syslog*-Dämon gestartet wird, da der syslog-Dämon sonst keine Nachrichten protokolliert.

-d Startet den syslog-Dämon im Debug-Modus, z.B. zur Problemdiagnose

-f Angabe einer alternativen Konfigurationsdatei; Standard-Konfigurationsdatei ist */etc/syslog.conf*.

-m Intervall (in Minuten), in dem der syslog-Dämon so genannte *MARK*-Meldungen in die Protokolldatei schreibt. Sie dienen dazu, in der Protokolldatei zu dokumentieren, dass der syslog-Dämon aktiv ist.

Standardmäßig sind 20 Minuten eingestellt, jedoch muss die Ausgabe von *MARK*-Meldungen zusätzlich in der Konfigurationsdatei aktiviert werden.

-l Pfadname, unter dem der syslog-Dämon einen zusätzlichen Socket einrichten soll. Der Pfad muss absolut angegeben werden.

Hauptaufgabe dieser Option ist es, für verschiedene *chroot*-Umgebungen zusätzliche Log-Sockets neben */var/run/log* zu erstellen. Zugriffsrechte der Sockets können als Oktalzahlen durch Doppelpunkt getrennt vor dem Socket-Namen angegeben werden (z.B.: `0640:/rootjail/var/run/log`).

-u Unique-Priority-Logging

Diese Option ändert den Defaultwert für den Vergleichsoperator der *priority*-Angabe in der Konfigurationsdatei von **=>** nach **=** (siehe Seite 288).

-v Verbose-Logging

Diese Option steuert die Protokollierung von Herkunft (*facility*) und/oder Gewicht (*priority*) der Nachricht. Ohne Angabe dieser Option werden beide Attribute nicht protokolliert.

Ist die Option **einmal** angegeben, werden die Zahlenwerte beider Attribute der Nachricht vorangestellt (z.B.: <1.4>).

Ist die Option **zweimal** angegeben, werden die symbolischen Werte beider Attribute der Nachricht vorangestellt (z.B.: <user.warn>).

Ist die Option **dreimal** angegeben, wird der symbolische Wert der Priorität in kompatibler Form der Nachricht vorangestellt (z.B.: LOG_WARNING).

Hinweis

Der *syslog*-Dämon wird durch das Skript */etc/rc2.d/S001syslog* beim POSIX-Start automatisch mit den Optionen **-Cvv** gestartet und protokolliert standardmäßig alle Meldungen in die Datei */var/adm/syslog*. Dies entspricht dem Eintrag `*.* /var/adm/syslog` in der Konfigurationsdatei */etc/syslog.conf*. In dieser Datei können zusätzliche oder alternative Protokolldateien festgelegt werden.

Damit Änderungen der syslog-Konfigurationsdatei sofort wirksam werden, muss dem syslog-Dämon ein Hangup-Signal gesendet werden (z.B. durch den Aufruf von `/etc/rc2.d/S001syslog restart`). Daraufhin schließt er alle Protokolldateien, liest die Konfigurationsdatei neu ein und öffnet die darin angegebenen Protokolldateien.

Der *syslog*-Dämon liest Nachrichten vom UNIX-Domain-Socket */var/run/log* und ggf. von den mit der Option *–l* angegebenen alternativen Sockets.

Der *syslog*-Dämon erstellt eine Datei */var/run/syslog.pid*, die seine Prozess-ID enthält.

Der aktuelle Zustand des syslog-Dämonen, seine Prozess-ID und die verwendeten Protokolldateien können durch den Aufruf von `/etc/rc2.d/S001syslog status` ermittelt werden.

Datei

/etc/syslog.conf
Standardkonfigurationsdatei

/var/run/syslog.pid
Datei für die Prozess-ID

/var/run/log
Name des UNIX-Domain-Sockets

/var/run/logpriv
UNIX-Domain-Socket für privilegierte Applikationen

/var/adm/syslog
Standard-Protokolldatei

/etc/rc2.d/S001syslog
Start-Skript

/etc/rc0.d/K999syslog
Stop-Skript

BS2000

Kernel-Logging wird im syslog-Dämon nicht unterstützt. Dieses erfolgt direkt auf die BS2000-Konsole.

Folgende Features in der Konfigurationsdatei werden nicht unterstützt und durch eine Konsolmeldung abgewiesen:

– Logging von und zu Remote-Hosts (fernen syslog-Dämonen)

– Logging auf Kommando-Pipes oder Benutzer-Terminals

Das Logging auf die Systemkonsole durch Angabe der Option LOG_CONS beim Aufruf der Funktion *syslog()* wird nicht unterstützt. Diese Option wird ignoriert.

Nur der Systemverwalter kann die Protokollierung von Meldungen auf die BS2000-Konsole oder die BS2000-CONSLOG-Datei bewirken, indem er in der Konfigurationsdatei für die betreffenden Meldungen die Ausgabedatei */dev/console* oder */dev/conslog* angibt.

syslog.conf – syslogd Konfigurationsdatei

syslog.conf ist die Konfigurationsdatei für den syslog-Dämon *syslogd*. Sie steuert, welche Nachrichten der syslog-Dämon in welche Protokolldateien schreibt.

Die Datei besteht aus Blöcken, die durch *program*-Spezifikationen getrennt sind. Jede in einem Block definierte Zeile enthält zwei Angaben:

– Die *selector*-Angabe wählt Nachrichten anhand von Typ und Priorität aus.

– Die *action*-Angabe definiert die Aktion, die der syslog-Dämon für die mit der *selector*-Angabe ausgewählten Nachrichten ausführen soll.

Die *selector*-Angabe wird von der *action*-Angabe durch mindestens einen Tabulator oder ein Leerzeichen getrennt.

Bei der Benutzung von Leerzeichen ist die Kompatibilität Ihrer Konfigurationsdatei mit anderen Unix-artigen Systemen nicht gewährleistet, wenn diese nur Tabulatoren erlauben.

program-Spezifikation

Eine *program*-Spezifikation leitet einen Block ein, der die Protokollierung von Nachrichten nur für die Programme steuert, die in dieser *program*-Spezifikation festgelegt sind. Ein Block am Anfang der Datei *syslog.conf* ohne vorangegangene *program*-Spezifikation steuert die Protokollierung für beliebige Programme.

Syntax

```
[#]![+|-]program[,program] ...
```

! Kennzeichnet die *program*-Spezifikation. Die optionale Angabe von **#** wird aus Kompatibiltätsgründen unterstützt.

+|- Positiv- oder Negativ-Auswahl (Standard: Positiv)

Bei Positiv-Auswahl (+) werden die im aktuellen Block enthaltenen Angaben nur für Nachrichten von Programmen ausgewertet, die in der *program*-Spezifikation angegeben sind, bei Negativ-Auswahl (-) nur für Nachrichten von Programmen, die nicht in dieser Spezifikation angegeben sind.

program

Programm-Name oder ***** (Stern, für alle Programme).

Beim *syslog()*-Aufruf wird geprüft, ob der angegebene Name (oder einer der angegebenen Namen) mit der *ident*-Angabe der Funktion *openlog()* des Programms übereinstimmt, aus dem der *syslog()*-Aufruf erfolgt.

selector-Angabe

Mit der *selector*-Angabe werden die Nachrichten ausgewählt, für die die in der *action*-Angabe festgelegten Aktionen ausgeführt werden sollen. Die *selector*-Angabe ist folgendermaßen aufgebaut (ohne Leerzeichen):

– *facility*-Angabe
– Punkt (.)
– optionale Negation
– optionale Vergleichsoperatoren
– *priority*-Angabe

Syntax

```
facility.[[!] [=>|>=|>|=|<|=<|<=]]priority
```

facility Es werden nur Nachrichten ausgewählt, die vom angegebenen Teil des Systems gesendet wurden.

Folgende Angaben sind möglich:

auth, **cron**, **daemon**, **lpr**, **mail**, **mark**, **news**, **syslog**, **user**, **uucp**, **local0** bis **local7** und *****.

Diese Schlüsselworte (mit Ausnahme von **mark**) stimmen mit den LOG_-Werten aus */usr/include/sys/syslog.h* überein, die in der Funktion *openlog()* angegeben werden können.

Die spezielle *facility*-Angabe **mark** bewirkt, dass periodisch (Standard: alle 20 Minuten, siehe Option *-m* des syslog-Dämons auf Seite 284) so genannte *MARK*-Meldungen in die Protokolldatei geschrieben.werden.

Die Angabe * (Stern) steht für beliebige *facility*-Angaben außer **mark**.

Bei der *facility*-Angabe wird zwischen Groß- und Kleinschreibung unterschieden.

! Negation. Die folgende Angabe (Vergleichsoperator und *priority*) wird logisch umgekehrt.

=>|>=|>|=|<|=<|<=
Die Vergleichsoperatoren wählen zusammen mit der nachfolgenden *priority*-Angabe die Meldungen anhand ihrer Priorität aus.
Wenn kein Vergleichsoperator angegeben ist, hängt das Verhalten von der Option *-u* beim Starten des syslog-Dämons (siehe Seite 284) ab. Standardmäßig (ohne Option *-u*) werden Nachrichten ausgewählt, deren Priorität größer oder gleich der angegebenen ist (entspricht **>=** bzw **=>**). Bei angegebener Option -u werden nur die Nachrichten mit der angegeben Priorität ausgewählt (entspricht **=**).
Vergleichsoperatoren mit einem vorangesetztem "!" werden logisch umgekehrt. Beispielsweise bedeutet "!=info", dass alle Nachrichten mit einer Priorität ungleich "info" ausgewählt werden.

priority Es werden nur Nachrichten ausgewählt, deren Priorität gleich der angegebenen (oder abhängig vom Vergleichsoperator höher bzw. niedriger) ist.
Folgende Angaben sind möglich (beginnend mit dem höchsten bis zum niedrigsten Fehlergewicht):
alert, **crit**, **err**, **warning**, **notice**, **info** und **debug**.
Diese Schlüsselworte stimmen mit den LOG_-Werten aus */usr/include/sys/syslog.h* überein, die in der Funktion *syslog()* angegeben werden können.

Die spezielle *priority*-Angabe **none** bewirkt, dass die betreffende *facility*-Angabe aus der beschriebenen Aktion ausgenommen wird.

Mit der *priority*-Angabe ***** (Stern) werden alle Prioritäten ausgewählt.

Bei der *priority*-Angabe wird zwischen Groß- und Kleinschreibung unterschieden.

Innerhalb einer *selector*-Angabe können mehrere *facility*-Angaben und auch mehere *priority*-Angaben stehen. Diese werden jeweils durch ein Komma (",") getrennt.

Es können auch mehrere *selector*-Angaben mit einer *action*-Angabe kombiniert werden. Diese werden jeweils durch ein Semikolon (";") getrennt. Wichtig dabei ist zu beachten, dass jede *selector*-Angabe die zuvor angegebene Definition wieder verändern kann.

action-Angabe

Die *action*-Angabe legt fest, welche Aktionen für die Nachrichten ausgeführt werden, die mit der *selector*-Angabe ausgewählt wurden und mit der *program*-Spezifikation des aktuellen Blockes übereinstimmen. Die *action*-Angabe ist folgendermaßen aufgebaut:

Syntax

```
[-]pfadname
```

- Nach jeder geschriebenen Nachricht wird *fsync()* aufgerufen, d. h. Nachrichten werden explizit synchronisiert. Standardmäßig werden Nachrichten nicht explizit synchronisiert.

pfadname
Die Nachrichten werden in die Datei mit dem angegebenen Pfadname protokolliert (angehängt). Der Pfadname muss beginnend mit **/** (Schrägstrich) angegeben werden.

Die Ausgabe von Nachrichten auf andere Ziele wie Kommando-Pipes, Terminals oder ferne Rechner wird im Gegensatz zu anderen syslogd-Implementierungen nicht unterstützt.

Kommentare

Kommentare werden durch Nummernzeichen # eingeleitet. Die restlichen Zeichen der Zeile einschließlich des # werden ignoriert, außer bei *program*-Spezifikationen (siehe Seite 287). Leere Zeilen und Zeilen, die vor dem ersten Nummernzeichen ausschließlich Leerzeichen oder Tabulatoren enthalten, werden vollständig ignoriert. Die Kommentarfunktion des Zeichens **#** kann durch Entwerten mit **** aufgehoben werden.

Datei *`/etc/syslog.conf`*
Konfigurationsdatei für syslogd

BS2000 Folgende Features werden nicht unterstützt und durch eine Konsolmeldung abgewiesen:

- Logging von und zu Remote-Hosts (fernen syslog-Dämonen)
- Logging auf Kommando-Pipes oder Benutzer-Terminals

Bei der Protokollierung in eine BS2000-Datei (bs2fs-Datei) ist zu beachten, dass das bs2fs-Dateisystem, in dem diese Datei liegt, bereits beim Start des syslog-Dämons eingehängt sein muss. Dazu kann der automatische *mount* durch einen Eintrag in */etc/vfstab* genutzt werden, z.B.:

```
/etc/vfstab
:FR07:$SYSROOT.POSIX.SYSLOG*  -  /var/adm/SYSROOT  bs2fs  1  yes  ftyp=text

/etc/syslog.conf
*.*    /var/adm/SYSROOT/POSIX.SYSLOG
```

Dabei müssen folgende Besonderheiten beachtet werden:

- Der syslog-Dämon wird unter der BS2000-Benutzerkennung SYSROOT gestartet. Die bs2fs-Dateien müssen daher in dieser Kennung liegen.
- Die aktuelle Protokolldatei bleibt aus BS2000-Sicht bis zur Beendung des syslog-Dämonen oder bis zum *logrotate* gesperrt. Sie kann so lange nur von POSIX aus gelesen werden.

Beispiel 1 Alle Meldungen werden in die Datei */var /adm/syslog* ausgegeben. Mit Ausnahme von Mail-Meldungen werden Meldungen ab dem Gewicht *err* und alle Authentisierungsmeldungen ab dem Gewicht *notice* zusätzlich in die BS2000-CONSLOG-Datei ausgegeben.

```
*.*                                 /var/adm/syslog
*.err;auth.notice;mail.none         /dev/conslog
```

Beispiel 2 Alle Meldungen ab dem Gewicht *info* mit Ausnahme der Mail-Meldungen und der Authentisierungsmeldungen werden in die Datei ausgegeben.

```
*.info;mail.none;auth.none   /var/adm/syslog
```

Beispiel 3 Dämon-Meldungen (ausschließlich) mit dem Gewicht *debug* werden in die Datei ausgegeben.

```
daemon.=debug                 /var/adm/daemon.debug
```

Beispiel 4 Mail-Meldungen und Authentisierungsmeldungen werden in spezifische Dateien ausgegeben. Alle anderen Meldungen werden in die syslog-Datei ausgegeben:

```
*.*;auth.none;mail.none       /var/adm/syslog
mail.*                        /var/adm/maillog
auth.*                        /var/adm/authlog
```

Beispiel 5 Mail- und News-Meldungen ab dem Gewicht *err* werden in die Datei ausgegeben.

```
mail,news.err                 /var/adm/mailerr
```

Beispiel 6 Meldungen des Programms *postfix* werden in die Mail-Protokolldatei ausgegeben.

```
!postfix
*.*                           /var/adm/maillog
```

Beispiel 7 Meldungen der Programme *named* und *dnsd* werden in eine eigene Datei ausgegeben und diese wird nach jeder einzelnen Meldung mit *fcync()* synchronisiert.

```
!named,dnsd
*.*                           -/var/adm/dnslog
```

Beispiel 8 *MARK*-Meldungen werden protokolliert.

```
mark.*                        /var/adm/syslog
```

10.9 Lokale Uhrzeit in POSIX

Die von POSIX-Schnittstellen gelieferte Information über die **lokale** Uhrzeit kann unter bestimmten Umständen von der von den entsprechenden BS2000-Schnittstellen gelieferten Information abweichen. Das betrifft jedoch nicht die Information über die **universelle** Uhrzeit (UTC).

Der Grund für diese Abweichung ist, dass BS2000-Schnittstellen und POSIX-Schnittstellen separate Mechanismen zur "Lokalisierung" der Uhrzeit verwenden.

- Im BS2000 gelten für die lokale Uhrzeit die Einstellungen, die in der Datei PARAMS.GTIME beim Systemstart festgelegt sind.

  Beispiel:

```
/BS2000 PARAMS
/BEGIN GTIME
ZONE=+01:00
DIFF=1:00
SEASON=W
CHDATE=1980-04-06/02:00
CHDATE=1980-09-28/03:00
...
CHDATE=2012-03-25/02:00
CHDATE=2012-10-28/03:00
CHDATE=2013-03-31/02:00
CHDATE=2013-10-27/03:00
...
/EOF
/END-PARAMS
```

- In POSIX gelten für die lokale Uhrzeit die Einstellungen, die beim Aufruf der betreffenden CRTE-Schnittstellen in der Umgebungsvariablen *TZ* festgelegt sind.

  Beispiel (Standard-Einstellung = Mitteleuropa):

```
$ echo $TZ
MEZ-1MSZ-2,M3.5.0/02:00:00,M10.5.0/03:00:00
$
```

  Der Inhalt der Variablen *TZ* ist in diesem Beispiel folgendermaßen zu interpretieren:

```
MEZ-1
    Standard-Zeitzone
        Name        MEZ
        Differenz   MEZ - 01:00:00 = UTC
```

`MSZ-2`
alternative Zeitzone
Name MSZ
Differenz MSZ - 02:00:00 = UTC

`M3.5.0/02:00:00`
Umschaltzeitpunkt zur alternativen Zeitzone
Monat 3 = März
Woche 5 (oder 4, falls der Monat nicht 5 Wochen hat)
Wochentag
0 = Sonntag
Uhrzeit 02:00:00

`M10.5.0/03:00:00`
Rückschaltzeitpunkt zur Standard-Zeitzone
Monat 10 = Oktober
Woche 5 (oder 4, falls der Monat nicht 5 Wochen hat)
Wochentag
0 = Sonntag
Uhrzeit 03:00:00

Versorgung der Variablen *TZ* in POSIX

Bei der Installation von POSIX-BC wird die Datei */etc/TIMEZONE* mit folgendem Inhalt installiert:

```
TZ=MEZ-1MSZ-2,M3.5.0/02:00:00,M10.5.0/03:00:00
```

Hierbei handelt es sich um ein Shell-Skript, das die Variable TZ in der aufrufenden Shell setzt, wenn es per Punkt-Kommando (`. /etc/TIMEZONE`) aufgerufen wird. Die Variable *TZ* ist automatisch exportiert, d.h. sie wird automatisch an Kindprozesse vererbt.

Das Skript */etc/TIMEZONE* wird bei jeder Art von POSIX-Login ausgeführt, also bei der Eröffnung einer Dialog- oder Batch-Session mit /START-POSIX-SHELL, bei *rlogin*, *telnet*, *rsh* und *ssh*. Außerdem wird es vor dem Starten von Dämonen durch die so genannten RC-Skripts ausgeführt.

Damit ist die Variable *TZ* in allen Dämonen und Login-Shells sowie in allen von diesen mit *fork()* erzeugten Prozessen versorgt und steuert damit die CRTE-Funktionen zur Ermittlung der lokalen Uhrzeit.

Die Datei */etc/TIMEZONE* wird auch bei einer Upgrade-Installation von POSIX-BC neu installiert. Eventuell durch den Administrator vorgenommene Änderungen werden dadurch überschrieben.

Einschränkungen für den POSIX-Mechanismus mit der *TZ*-Variablen

Der POSIX-Mechanismus erlaubt nur eine feste Regel für die Umschaltzeitpunkte zwischen Standard- und alternativer Zeitzone und keine Variationen von Jahr zu Jahr. Das entspricht den heute geltenden EU-Regelungen.

Daher werden Zeitstempel, die vor 1996 liegen, d.h. vor der Einführung einer festen Regel für die Umstellzeitpunkte, unter Umständen fehlerhaft "lokalisiert".

Fachwörter

In diesem Verzeichnis sind die wichtigsten Begriffe dieses Handbuchs in alphabetischer Reihenfolge aufgeführt und erklärt. Bei Betriebssystem-spezifischen Fachwörtern ist angegeben, aus welcher Betriebssystem-Umgebung sie kommen (BS2000, POSIX oder UNIX).
Querverweise auf andere Fachwörter sind durch *Kursivdruck* gekennzeichnet.

64-bit-Dateischnittstellen
64-bit file interfaces

Menge von neuen Dateischnittstellen, mit denen *große POSIX-Dateien* bearbeitet werden können. Diese Dateischnittstellen wurden von existierenden Dateischnittstellen abgeleitet und besitzen das Suffix 64, z.B. open64().

64-bit-Integer-Arithmetik
64-bit integer arithmetic

Die 64-bit-Integer-Arithmetik benutzt Integer-Zahlen mit einer Länge von 64 Bits (ohne Vorzeichen) bzw. 63 Bits und ein Vorzeichenbit. Sie ist über ein Paar von jeweils 32-bit-Integers implementiert (Datentyp long long).

Abrechnungsnummer
account number

BS2000: Bezeichnet ein Abrechnungskonto für die zugehörige *Benutzerkennung*. Mehreren Benutzerkennungen kann dieselbe Abrechnungsnummer zugewiesen werden. Eine Benutzerkennung kann über maximal 60 Abrechnungsnummern verfügen. Die Abrechnungsnummer wird bei SET-LOGON-PARAMETERS und ENTER-JOB ausgewertet.

Absoluter Pfadname
absolute pathname

POSIX/UNIX: *Pfadname* für eine *Datei* oder ein *Dateiverzeichnis*, der beim *Root-Verzeichnis* (/) des *POSIX-Dateisystems* beginnt und durch alle übergeordneten Dateiverzeichnisse führt. Jede Datei und jedes Dateiverzeichnis hat einen eindeutigen absoluten Pfadnamen.

Aktuelles Dateiverzeichnis
current directory

POSIX/UNIX: *Dateiverzeichnis*, in dem der *Benutzer* gerade arbeitet; es kann mit dem POSIX-Kommando pwd angezeigt werden. Im aktuellen Dateiverzeichnis kann der Benutzer auf sämtliche *Dateien* und *Unterverzeichnisse* direkt zugreifen.

Application Programming Interface (API)

Schnittstelle zwischen den Anwendungen und den von diesen benutzten System-/Subsystemfunktionen.
siehe auch *Distributed Computing Environment*

ASCII

Abkürzung für **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.
Standardisierter Code zur Umwandlung von Groß- und Kleinbuchstaben, Ziffern, Sonder- und Steuerzeichen in digitale Ziffern, die im Rechner verarbeitet werden können. *UNIX* und *SINIX* arbeiten mit dem ASCII-Code.
POSIX kann ASCII-Daten nach der Konvertierung in das EBCDIC-Format bearbeiten.

Authentisierung
authentication

BS2000: Überprüfung der Angaben eines *Benutzers* beim Systemzugang. Die Benutzerattribute „*Benutzerkennung*“ und „*Kennwort*“ werden gegen die Einträge im *Benutzerkennungs-Katalog* geprüft.

Behälterdatei
container file

POSIX: BS2000-PAM-Datei, in der ein *POSIX-Dateisystem* abgelegt ist. Eine Behälterdatei wird auf einem Pubset abgelegt. Behälterdateien und andere BS2000-Dateien dürfen auf demselben Pubset liegen.

Benutzer
user

BS2000: Repräsentant einer *Benutzerkennung*. Der Begriff Benutzer ist ein Synonym für Personen, Anwendungen, Verfahren etc., die über eine Benutzerkennung Zugang zum *Betriebssystem* erhalten können.

Benutzerattribute
user attributes

BS2000/POSIX: Alle Merkmale einer *Benutzerkennung*, die im *Benutzerkennungs-Katalog* hinterlegt sind.

Benutzergruppe
user group
BS2000: Zusammenfassung einzelner *Benutzer* unter einem Namen.

Benutzerkatalog
user catalog
siehe *Benutzerkennungs-Katalog*

Benutzerkennung
user identification, USER-ID
BS2000: Maximal acht Zeichen langer Name, der im *Benutzerkennungs-Katalog* eingetragen wird. Anhand der Benutzerkennung wird der *Benutzer* beim Systemzugang identifiziert. Alle *Dateien* und *Jobvariablen* werden unter einer Benutzerkennung eingerichtet. Die Namen der Dateien und Jobvariablen werden mit der Benutzerkennung im Dateikatalog hinterlegt.

Benutzerkennungs-Katalog
join file
BS2000: *Datei*, die die *Benutzerattribute* aller *Benutzerkennungen* eines *Pubsets* enthält.

Benutzerkennungs-Kennwort
password for user identification
Dient zur *Authentisierung* des *Benutzers* und gewährleistet den Zugangsschutz.

Benutzerklasse Andere
file other class
POSIX/UNIX: Eigenschaft einer *Datei*, die das *Zugriffsrecht* für einen *Prozess* anzeigt, der mit der Benutzeridentifikation eines Prozesses verbunden ist. Ein Prozess gehört zur Benutzerklasse Andere einer Datei, wenn er nicht der *Benutzerklasse Eigentümer* oder der *Benutzerklasse Gruppe* angehört.

Benutzerklasse Eigentümer
file owner class
POSIX/UNIX: Eigenschaft einer *Datei*, die das *Zugriffsrecht* für einen *Prozess* anzeigt, der mit der Benutzer- und Gruppenidentifikation eines Prozesses verbunden ist. Ein Prozess gehört zur Benutzerklasse Eigentümer einer Datei, wenn die effektive *Benutzernummer* des Prozesses zur Benutzernummer der Datei passt.

Benutzerklasse Gruppe
file group class
POSIX/UNIX: Ein *Prozess* gehört zur Benutzerklasse Gruppe einer Datei, wenn er nicht der *Benutzerklasse Eigentümer* angehört und die effektive *Gruppennummer* oder eine der zusätzlichen Gruppennummern des Prozesses zur Gruppennummer der Datei passt.

Benutzernummer
user ID
POSIX/UNIX: Positive ganze Zahl, durch die ein Systembenutzer identifiziert wird.

Benutzerorganisation
BS2000: Zusammenfassung von *Benutzerkennungen* zu *Benutzergruppen*. Dadurch wird die Nachbildung bestehender Organisationsformen ebenso gestattet wie die projektorientierte Zusammenfassung von *Benutzern*.

Benutzerrechte
user privileges
BS2000: Alle an eine *Benutzerkennung* vergebenen und im *Benutzerkennungs-Katalog* hinterlegten Attribute, die Rechte darstellen.

Benutzerverwaltung
user administration
siehe *Systemglobale Benutzerverwaltung*

Betriebssystem
operating system
Gesamtheit aller Software- und Firmware-Programme, die, ohne auf einen bestimmten Anwendungsfall zugeschnitten zu sein, den Betrieb eines Computers ermöglichen. In der Regel wird das Betriebssystem vom Computerhersteller mitgeliefert.

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.

Blockterminal
block-mode terminal
Terminal, das keine zeichenweisen Ein- und Ausgabe-Operationen unterstützt.

Client
client

Rechner, der von einem anderen Rechner (*Server*) Dienste in Anspruch nimmt. Ein Rechner kann gleichzeitig für bestimmte Funktionen als Client Dienste anfordern und für andere Rechner als Server Dienste zur Verfügung stellen.

Client-Server-Architektur
client server architecture

Systemarchitektur, in der Rechnerkapazitäten und Anwendungen auf Dienstnehmer *(Clients)* und Dienstanbieter *(Server)* verteilt werden.
Server-Funktionen werden überwiegend von Mainframe- und UNIX-Systemen ausgeübt, die dafür bestimmten Bedingungen genügen müssen.
Client-Funktionen haben vor allem PCs, Workstations und UNIX-Systeme.
Client- und Server-Systeme können beliebig kombiniert werden; jeder Client kann grundsätzlich auf jeden Server zugreifen.

Dämonprozess
daemon

POSIX/UNIX: Systemprozess, der permanent und meistens im Hintergrund abläuft; er führt allgemeine Aufgaben durch. Bekanntes Beispiel ist der Drucker-Dämon, der dafür sorgt, dass eine Datei ausgedruckt wird, während der Benutzer bereits wieder arbeitet.

Datei
file

UNIX: Eine Datei wird bei *UNIX* über einen Indexeintrag identifiziert. Dieser Eintrag enthält die Informationen, ob die Datei eine *normale Datei*, eine *Gerätedatei* oder ein *Dateiverzeichnis* ist. Eine normale Datei enthält Text, Daten, Programme oder sonstige Informationen. Eine Gerätedatei bezeichnet ein Gerät oder einen Teil eines Gerätes, wie zum Beispiel ein Laufwerk oder eine Festplattenpartition. Ein Dateiverzeichnis enthält andere Dateien.

BS2000: Sätze, die zueinander in Beziehung stehen, werden in einer benannten Einheit, der Datei, zusammengefasst. Dateien sind beispielsweise: Konventionelle Ein-/Ausgabedaten von Programmen; Lademodule; Textinformation, die mit einem Dateiaufbereiter erstellt und verarbeitet wird.

Dateisystem
file system

POSIX/UNIX: Ansammlung von *Dateiverzeichnissen* und *Dateien* und bestimmter Attribute von Dateien.

Dateiverzeichnis
directory

POSIX/UNIX: Ein Dateiverzeichnis wird verwendet, um *Dateien* oder Dateiverzeichnisse zu gruppieren und zu organisieren.

Distributed Computing

siehe *Verteilte Verarbeitung*

EBCDIC

Abkürzung für **E**xtended **B**inary **C**oded **D**ecimal **I**nterchange **C**ode.
EBCDI-Code ist ein auf 8 Bit erweiterter BCD-Code, der auf BS2000-Rechnern, TRANSDATA-Kommunikationsrechnern und Maschinen des Industriestandards verwendet wird.

Einhängen von Dateisystemen
mounting file systems

POSIX/UNIX: Mit dem Kommando mount können *Dateisysteme* in ein lokales Dateisystem eingehängt werden. Der *Einhängepunkt* muss vorher als *Dateiverzeichnis* definiert worden sein. Nach dem Einhängen des Dateisystems ist dieses Dateiverzeichnis nicht mehr sichtbar.

Einhängepunkt
mount point

POSIX/UNIX: Name eines Dateiverzeichnisses, unter dem eine ferne Ressource, z.B. ein Dateibaum, eingehängt ist.

Ethernet

Standardverfahren zur Kopplung von zwei Rechnern; daraus entsteht ein *lokales Rechnernetz* (Local Area Network).

Ferner Rechner
remote system

In einem lokalen *Netz* werden ferne und *lokale Rechner* unterschieden. Alle Rechner im Netz, an denen ein *Benutzer* nicht direkt arbeitet, sind für diesen Benutzer ferne Rechner.

First-Start
first start

BS2000: Beim First-Start des BS2000 werden Systemdateien neu eingerichtet. Das System vergibt eine Reihe von *Benutzerkennungen*, z.B. TSOS, SYSPRIV und SYSHSMS. Beim First-Start wird immer der *Benutzerkennungs-Katalog* angelegt.

fork

POSIX/UNIX: Systemaufruf, der einen *Prozess* in zwei Prozesse teilt: den *Vater-* und den *Sohnprozess*

Gerätedatei

special file

Eine auch als Gerätetreiber bezeichnete *Datei*, die als Schnittstelle zu einem Ein-/Ausgabegerät (z.B. Terminal, Plattenlaufwerk) benutzt wird.

Große POSIX-Dateien

Large POSIX files

POSIX-Dateien, die eine Größe von mehr als 2 Gbyte haben können (2 Gbyte = 2^{31} -1 byte). Für die Adressierung innerhalb einer solchen Datei ist die *64-bit-Arithmetik* notwendig. Große POSIX-Dateien können nur in *großen POSIX-Dateisystemen* angelegt werden und müssen mit *64-bit-Dateischnittstellen* bearbeitet werden.

Große POSIX-Dateisysteme

Large POSIX file systems

POSIX-Dateisysteme, die größer als 2 Gbyte werden können.Für die Adressierung innerhalb eines solchen Dateisystems ist die *64-bit-Arithmetik* notwendig. Die maximale Größe eines großen POSIX-Dateisystems beträgt 1024 Gbyte (abhängig von der OSD-Version). Große POSIX-Dateisysteme sind Voraussetzung für *große POSIX-Dateien*.

Gruppenmitglied

group member

BS2000: *Benutzerkennung*, die einer *Benutzergruppe* zugeordnet ist. Der Gruppenverwalter kann einem Gruppenmitglied Ressourcen zuweisen.

Gruppennummer

group ID

Positive ganze Zahl zum Identifizieren einer Gruppe von Benutzern. Jeder Benutzer ist Mitglied mindestens einer Gruppe.

Heterogenes Rechnernetz

heterogeneous network

siehe *Offenes Rechnernetz*

Hintergrundprozess
background process

Prozess, der die Ressourcen des Rechners nicht vollständig ausschöpft, sondern die gleichzeitige Durchführung von weiteren Prozessen ermöglicht. Ein Hintergrundprozess nutzt normalerweise die Zeitnischen aus, in denen der Prozessor sonst unbeschäftigt wäre.

Home-Verzeichnis
home directory

POSIX: *Dateiverzeichnis*, in das der *Benutzer* automatisch gelangt, wenn er mit POSIX verbunden wird.

Homogenes Rechnernetz
homogeneous network

Rechnernetz, in dem die einzelnen Rechner dieselbe oder eine ähnliche Architektur haben.

Host
host

Zentralrechner eines *Rechnernetzes*. Auf dem Host werden Programme durchgeführt, Dateien gespeichert sowie Ein- und Ausgaben gesteuert. In vielen Fällen enthält ein leistungsfähiges Rechnernetz mehrere Hosts.

Institute of Electrical and Electronics Engineers (IEEE)

Organisation, die bedeutende Normen in der Computer- und Kommunikationsindustrie festgelegt hat, so z.B. die Norm IEEE1003 POSIX.

International Organization for Standardization (ISO)

Internationale Normungsbehörde, die im Jahre 1946 gegründet wurde und inzwischen aus über 90 nationalen Normungsgremien besteht. ISO legt unter anderem auch Normen für die Software von *Rechnernetzen* fest.

Internet

Weltweites *Rechnernetz* auf IP-Basis, das Tausende von verschiedenen Rechnernetzen miteinander verbindet und vom Network Information Center verwaltet wird. Über Internet können nur hardwareunabhängige Datenpakete verschickt werden.

Interoperabilität
interoperability

Fähigkeit, Systeme von verschiedenen Computerherstellern miteinander verbinden zu können und sie zusammen arbeiten zu lassen, um Arbeitsanforderungen - wie z.B. verteilte Datenhaltung - erfüllen zu können. Dabei muss der *Benutzer* die Eigentümlichkeiten der einzelnen Systeme gar nicht oder nur wenig kennen.

ISO-Referenzmodell

Rahmen für die Standardisierung der Kommunikation offener Systeme.
ISO, die internationale Organisation für Standardisierung, hat dieses Modell in dem internationalen Standard ISO 7498 beschrieben. Das ISO-Referenzmodell unterteilt die Funktionen, die für die Kommunikation von Systemen notwendig sind, in sieben logische Schichten. Diese Schichten haben jeweils klar definierte Schnittstellen zu den benachbarten Schichten und kommunizieren mit den jeweils entsprechenden Schichten auf dem Partner-Rechner über *Protokolle*.

Jobvariable
job variable

BS2000: Jobvariablen sind Speicherbereiche zum Austausch von Informationen zwischen Aufträgen (Jobs) untereinander sowie zwischen *Betriebssystem* und Aufträgen. Sie haben einen Namen und einen Inhalt (Wert). Der Inhalt kann zur Steuerung von Aufträgen und Programmen genutzt werden.
Der *Benutzer* kann Jobvariablen erzeugen, verändern, abfragen und löschen. Außerdem kann er das Betriebssystem anweisen, eine überwachende Jobvariable entsprechend zu setzen, wenn sich der Zustand eines Auftrags oder eines Programms ändert.

Katalogkennung
catalog identification, catid

BS2000: Kennzeichen eines *Pubsets*.
Die Katalogkennung besteht aus maximal 4 Zeichen, die innerhalb von Dateinamen bzw. *Pfadnamen* in Doppelpunkte eingeschlossen werden müssen. Die Katalogkennung ist Bestandteil des *Pfadnamens* einer *Datei*.

Kennwort
password

Folge von Zeichen, die der *Benutzer* eingeben muss, um den Zugriff zu einer *Benutzerkennung*, einer *Datei*, einer *Jobvariablen*, einem Netzknoten oder einer Anwendung zu erhalten.

Kompatibilität
compatibility
Fähigkeit, Tasks ohne größere Änderungen in verschiedenen Systemumgebungen mit gleichem Ergebnis durchführen zu können.

large file aware
Ein Programm ist „large file aware“, wenn es *große POSIX-Dateien* korrekt bearbeitet.

large file safe
Ein Programm ist „large file safe“, wenn es *große POSIX-Dateien* erkennt und deren Verarbeitung definiert zurückweist, z.B. mit einer entsprechenden Meldung.

Local Area Network (LAN)
Hardware-Konfiguration eines lokalen *Netzes*, in dem alle Datensichtgeräte und sonstigen Geräte in relativ geringem Abstand zueinander aufgestellt sind, z.B. innerhalb desselben Gebäudes. Die geringe Entfernung erlaubt einfachere Übertragungstechniken und damit höhere Geschwindigkeiten zu einem geringen Preis.
In der Bundesrepublik Deutschland ist die Größe eines LAN auf das Grundstück des Anwenders beschränkt. Ein LAN kann als privates Subnetz mit anderen *Rechnernetzen* verbunden und so Teil eines größeren Netzes sein, etwa eines WAN.
Synonyme: Lokales Rechnernetz, lokales Netz.

Login-Verzeichnis
login directory
siehe *Home-Verzeichnis*

Lokales Dateisystem
local file system
POSIX/UNIX:

Lokales (Rechner-)Netz
siehe *Local Area Network*

Lokaler Rechner
local system
Für einen *Benutzer* ist immer derjenige Rechner lokal, an dem er arbeitet. Alle anderen Rechner im *Rechnernetz* sind dann für ihn *ferne Rechner*.

Mehrfach benutzbare Datei
shareable file

BS2000: Eine *Datei*, die der *Benutzer* mit dem Operanden USER-ACCES= *ALL-USERS katalogisiert hat. Dateien, die auf diese Weise als mehrfach benutzbar gekennzeichnet sind, können von allen Benutzern aufgerufen werden. Voraussetzung ist aber, dass der Benutzer die *Benutzerkennung* des Erstellers der Datei kennt und ggf. das *Kennwort* angeben kann, falls die Datei geschützt ist.

Mini-POSIX-Dateisystem

POSIX: NFS- oder DFS-Dateisystem, das mit BS2000/OSD V1.0 oder früheren BS2000-Versionen erstellt wurde. Es kann in *POSIX-Dateisystem*e migriert werden.

Motif

siehe *OSF/Motif*

Network File System (NFS)

BS2000: Softwareprodukt, mit dem verteilte Datenhaltung in einem heterogenen *Rechnernetz* möglich ist. Der *Benutzer* kann auf ferne *Dateien* so zugreifen, als ob sie an seinem *lokalen Rechner* vorhanden wären.

Netz
network

Komplexes Gebilde aus Leitungen und Steuerungseinrichtungen, das der Datenfernübertragung dient.

Normale Datei
regular file

Datei, die eine wahlfrei zugreifbare Folge von Bytes ohne jede weitere vom System festgelegte Struktur ist.

Offenes Rechnernetz
open network

Rechnernetz, in dem nach den Regeln von *ISO* kommuniziert wird. Durch festgelegte *Protokolle* können Rechner von verschiedenen Computerherstellern miteinander arbeiten.

Pfadname
pathname

POSIX/UNIX: Jede *Datei* und jedes *Dateiverzeichnis* besitzt einen eindeutigen Pfadnamen. Der Pfadname gibt die Position der Datei bzw. des Dateiverzeichnisses innerhalb des *Dateisystems* an und zeigt, wie darauf zugegriffen werden kann. Der Pfadname besteht aus den Namen aller darüberliegenden Dateiverzeichnisse, ausgehend von der Spitze des Dateisystems, und dem eigentlichen Namen der Datei oder des Dateiverzeichnisses. Die Namen der Dateiverzeichnisse werden jeweils durch einen Schrägstrich (Slash) voneinander getrennt. (Beispiel: c:verzeich1/verzeich2/protokoll)
Es wird zwischen *absoluten* und *relativen Pfadnamen* unterschieden.

BS2000: Jede im BS2000 katalogisierte Datei ist ebenfalls durch einen Pfadnamen eindeutig identifizierbar. Der Pfadname setzt sich zusammen aus der *Katalogkennung* (catid), der *Benutzerkennung* (userid) und einem vom Benutzer vergebenen vollqualifizierten Dateinamen:
:catid:$userid.dateiname

Pipe
pipe

POSIX/UNIX: Verkettung zweier *POSIX-/UNIX*-Kommandos. Eine Pipe bewirkt, dass die Ausgabe eines Programms die Eingabe des nächsten Programms wird, so dass die Programme nacheinander ausgeführt werden. Eine Pipe wird erzeugt, indem nach dem ersten Kommando das Pipe-Symbol | angegeben wird. Die Ausgabe des Prozesses links vom Pipe-Symbol wird an den *Prozess* rechts vom Pipe-Symbol geleitet.

Plattform

Betriebssystemumgebung, in der ein Programm abläuft.

Portabilität
portability

Fähigkeit eines Programms, unverändert auf unterschiedlichen *Betriebssystemen* ablaufen zu können. Sie wird durch die Verwendung standardisierter, offener Programmschnittstellen erreicht, die auf einer Vielzahl von *Plattformen* angeboten werden.

Portable Open System Interface for UNIX (POSIX)

Schnittstellennormen für offene Systeme, die von der IEEE festgesetzt wurden und auf UNIX basieren. POSIX enthält Normen für ein breites Spektrum von Betriebssystemkomponenten, angefangen von der Programmiersprache C bis zur Systemverwaltung. POSIX-Arbeitsgebiete sind unter anderem:

1003.00 Guide to POSIX Open System Environment
1003.01 System Application Program Interface (API)
1003.02 Shell and Utilities
1003.07 System Administration

Portierbarkeit

siehe *Portabilität*

POSIX.1-1988

IEEE Std 1003.1-1988, Standard für Informationstechnologie - POSIX - Teil 1: System Application Program Interface (API)

POSIX-Dateisystem

POSIX file system

BS2000/POSIX: *Dateisystem* im BS2000 mit der Struktur eines UNIX-Dateisystems (UFS). Es kann aus mehreren Dateisystemen bestehen. Es ist hierarchisch aufgebaut und besteht aus *Dateiverzeichnissen* und *Dateien* (POSIX-Dateien). An der Spitze der Hierarchie steht das Dateiverzeichnis *root*, das durch einen Schrägstrich (/) gekennzeichnet ist. Von hier aus setzt sich die Verzeichnisstruktur weiter nach unten fort. Von Dateiverzeichnissen aus kann in weitere Dateiverzeichnisse oder in Dateien verzweigt werden. Von einer Datei aus ist keine Verzweigung mehr möglich. Jede Datei eines Dateisystems ist über genau einen absoluten Pfad erreichbar.
Der Unterschied zwischen einem POSIX- und einem UNIX-Dateisystem besteht im Ablageort: Bei einem UNIX-Dateisystem ist der Ablageort ein physikalisches Gerät, bei einem POSIX-Dateisystem eine PAM-*Behälterdatei*.

POSIX-Shell

POSIX shell

BS2000/POSIX: Portiertes *SINIX*-Systemprogramm, das die Kommunikation zwischen dem *Benutzer* und dem System übernimmt. Die POSIX-Shell ist ein Kommando-Interpreter. Sie übersetzt die eingegebenen POSIX-Kommandos in eine Sprache, die das System erkennt.
Wenn beim Benutzerattribut „Programm“ die POSIX-Shell eingetragen ist, wird die POSIX-Shell gestartet, sobald sich der Benutzer durch Remote Login an POSIX angeschlossen hat.

POSIX-Verwalter
POSIX administrator

Inhaber des Privilegs POSIX-ADMINISTRATION. Dieses Privileg ist automatisch an die Systemkennung SYSROOT geknüpft und kann SYSROOT nicht entzogen werden. Anderen *Benutzerkennungen* kann der Sicherheitsbeauftragte dieses Privileg auch verleihen und entziehen.

Protokoll
protocol

Regeln für den Datenaustausch zwischen zwei Rechnern, die die Art der Verbindung, das Datenformat sowie die Abfolge der Daten bestimmen.

Prozess
process

POSIX/UNIX: Adressraum, in dem ein einzelner Programmcode ausgeführt wird, sowie die dafür benötigten Betriebsmittel des Systems.
Ein Prozess wird von einem anderen Prozess durch den Aufruf der Funktion *fork* erzeugt. Der Prozess, der fork aufruft, heißt Vaterprozess; der durch fork erzeugte Prozess heißt Sohnprozess.

Public Volume Set

BS2000: Satz gemeinschaftlich gekennzeichneter Platten. MPVS-Systeme arbeiten mit mehreren voneinander unabhängigen Pubsets.

Pubset

BS2000: Kurzform für *Public Volume Se*t

Rechnernetz
network

Zusammenschluss mehrerer Rechner über eine physikalische Verbindung mit dem Ziel, einen gleichberechtigten Datenaustausch zwischen diesen Rechnern zu ermöglichen. Es gibt lokale (LAN) und weite (WAN) Rechnernetze, *heterogene* und *offene Rechnernetze*.

Rechnernetz, offenes

siehe *Offenes Rechnernetz*

Relativer Pfadname
relative pathname

POSIX: Pfadname für eine *Datei* oder ein *Dateiverzeichnis*, der von der Position des *aktuellen Dateiverzeichnisses* innerhalb des *Dateisystems* ausgeht. Relative Pfadnamen beginnen nicht mit einem Schrägstrich (/).

Reliant UNIX

Nachfolger von *SINIX*, das mit der Version 5.43 als Konsequenz aus der Zusammenführung der UNIX-Versionen von ehemals Siemens Nixdorf und Pyramid Technology in Reliant UNIX umbenannt wurde. Der Name Reliant UNIX steht für die hohen Anforderungen an Zuverlässigkeit und Verfügbarkeit, die dieses standardisierte Betriebssystem erfüllt – im kommerziellen wie im technischen Einsatz – wobei alle bewährten *SINIX*-Eigenschaften in Reliant UNIX erhalten bleiben.

root

root

POSIX/UNIX: Benutzername (Systemverwalter-Kennung) mit den meisten Privilegien.

Root-Berechtigung

Kennung, der die *Benutzernummer* 0 und die *Gruppennummer* 0 zugeteilt ist. Die Systemkennung SYSROOT hat standardmäßig die Root-Berechtigung.

Root-Verzeichnis

root directory

POSIX/UNIX: Hauptdateiverzeichnis in einem hierarchisch strukturierten *Dateisystem*, von dem alle anderen *Dateiverzeichnisse* abzweigen.

Schutzattribute

security attributes

BS2000: Sicherheitsrelevante Eigenschaften eines Objekts (*Datei*, *Jobvariable* etc.), die die Art und Möglichkeit des Zugriffs auf dieses Objekt festlegen. Für Dateien gibt es beispielsweise folgende Schutzattribute: ACCESS/ USER-ACCESS, SERVICE-bit, AUDIT-Attribut, RDPASS, WRPASS, EXPASS, RETPD, BACL und GUARD.

Schutzbits

permission bits

POSIX/UNIX: Information über Lese-, Schreib- oder Ausführungsrecht einer Datei. Die Bits sind in drei Abschnitte eingeteilt: Eigentümer, Gruppe und andere Benutzer.

Semaphor

semaphore

POSIX/UNIX:

Server

server

Rechner, der anderen Rechnern (*Clients*) Dienste zur Verfügung stellt.

shutdown

siehe *Systembeendigung*

Sitzung

session

Vorgänge/Aktivitäten, die zwischen der *Systemeinleitung* und der *Systembeendigung* stattfinden.

Sockets

Schnittstelle für Netzzugriffe über TCP/IP.

Sohnprozess

child process

siehe *fork*

Systembeendigung

shutdown

BS2000: Vorgang der geordneten Systembeendigung einschließlich des Sicherns spezieller Systemdateien.

Systemeinleitung

startup

BS2000: Laden der *Betriebssystem*-Software des BS2000. Es gibt drei Varianten:

- AUTOMATIC-STARTUP
- DIALOG-STARTUP
- FAST-STARTUP

Diese Varianten unterscheiden sich durch den unterschiedlichen Automatisierungsgrad.

Systemglobale Benutzerverwaltung

user administration

BS2000: Umfasst die Verwaltung von *Benutzerkennungen* und *Benutzergruppen* bezüglich Ressourcen und Benutzerrechten sowie das Neueinrichten, Modifizieren und Löschen von Benutzerkennungen und Benutzergruppen.

Systemglobale Privilegien

user privileges

BS2000: Alle Rechte, die mit dem Kommando /SET-PRIVILEGE vergeben werden können, sowie das Recht des Sicherheitsbeauftragten und der Systemkennung TSOS.

Systemkern

kernel

POSIX/UNIX: Code des POSIX-/*UNIX-Betriebssystems*.

Systemverwalterrechte
siehe *Systemglobale Privilegien*

TCP/IP
Netztprotokoll der Internet-Architektur.

UNIX File System (UFS)
UNIX-Versionen <= IV:
Dateiverwaltungs-Komponente des UNIX-*Systemkerns*.
System V:
Dateisystem-Variante des Virtual File System für die Verwaltung lokaler Dateisysteme.

UNIX-Betriebssystem
Ein im Dialogbetrieb arbeitendes *Betriebssystem*, das 1969 von Bell Laboratories entwickelt wurde. Da nur ein zentraler *Systemkern* von UNIX hardwareabhängig ist, wird UNIX auf vielen unterschiedlichen Systemen verschiedener Computerhersteller eingesetzt.

UNIX95
Synonym für *XPG4.2*

Unterverzeichnis
subdirectory
POSIX/UNIX: *Dateiverzeichnis*, über dem ein weiteres Dateiverzeichnis liegt.

Vaterprozess
parent process
siehe *fork*

Verteilte Verarbeitung
Distributed Computing
Die Anwendungen und meistens auch die Daten und Ressourcen sind zwischen den einzelnen Rechnern eines *Netzes* aufgeteilt.

Verzeichnis
siehe *Dateiverzeichnis*

Vordergrundprozess
foreground process
Prozess, der das Terminal vollständig in Beschlag nimmt, so dass es von anderen Prozessen nicht genutzt werden kann.

Wide Area Network (WAN)

Rechnernetz, das nicht auf ein räumlich begrenztes Gebiet beschränkt ist.

X/Open (OPEN GROUP)

Unabhängige, weltweite Organisation für offene Systeme, die von fast allen großen Computerherstellern, Benutzerverbänden und Softwaregesellschaften unterstützt wird. Ziel von X/Open ist die Implementierung offener Systeme, damit die *Benutzer* ihre Rechner besser einsetzen können. X/Open wirkt bei einer Vielzahl von internationalen Normierungsgremien mit.

XPG4

X/Open Portability Guide, Issue 4.
Sammlung der X/Open-Schnittstellenstandards.

XPG4.2

X/Open Portability Guide, Issue 4, Version 2.
Erweiterung von *XPG4*.

Zugriffsrecht

access permission

POSIX/UNIX: Eigenschaft einer *Datei*, die den Zugriff auf diese Datei steuert. Zugriffsrechte werden getrennt vergeben an den Eigentümer (siehe *Benutzerklasse Eigentümer*), die Gruppe des Eigentümers (siehe *Benutzerklasse Gruppe*) und alle anderen Benutzer (siehe *Benutzerklasse Andere*). Es gibt drei grundlegende Zugriffsrechte: Lese-, Schreib- und Ausführungsrecht.

Abkürzungen

AID	Advanced Interactive Debugger
ANSI	American National Standards Institute
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BOOTP	BOOtsTrap Protocol
BSD	Berkeley Software Distribution
CRTE	Common RunTime Environment
DCE	Distributed Computing Environment
DFS	Distributed File System
DME	Distributed Management Environment
DMS	Data Management System
DNS	Domain Name Service
DSSM	Dynamic Subsystem Management
EBCDIC	Extended Binary Coded Decimal Interchange Code
EDT	EDiTor
HSMS	Hierachical Storage Management System
HSMS-SV	Hierachical Storage Management System - SerVer
IEEE	Institute of Electrical and Electronics Engineers

ISAM	Indexed Sequential Access Method
ISO	International Organization for Standardization
LAN	Local Area Network
NFS	Network File System
OPS	Output Presentation Service
OSD	Open Systems Direction
OSF	Open Software Foundation
PAM	Primary Access Method
PLAM	Program Library Access Method
PLIB	POSIX Library
POSIX	Portable Open System Interface for UNIX
SAM	Sequential Access Method
SCI	Software Configuration Inventory
SIA	System Interfaces for Application
SNMP	Simple Network Management Protocol
SPOOL	Simultaneous Peripheral Operation On Line
SRPM	System Resources and Privileges Management
Sun	Sun Microsystems
TCP/IP	Transmission Control Protocol / Internet Protocol
TC-IP-AP	TCP/IP Application Programs
TCP-IP-SV	TCP/IP SerVices
TFTP	Trivial File Transfer Protocol

TIAM	Terminal Interactive Access Method
TLI	Transport Layer Interface
TOG	The Open Group
TPR	Task Privileged
TU	Task Unprivileged
UDP/IP	User Datagram Protocol/Internet Protocol
UFS	UNIX File System
UNIX95	Synonym für XPG4.2
URL	Uniform Resource Locator
USL	UNIX System Laboratories
UTC	Universal Time Coordinated
WAN	Wide Area Network
WWW	World Wide Web
X/Open	X/Open Company Ltd.
XPG4	X/Open Portability Guide Issue 4
XPG4.2	X/Open Portability Guide Issue 4, Version 2
XTI	X/Open Transport Interface

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** (BS2000/OSD)
Kommandos
Benutzerhandbuch

[2] **POSIX** (BS2000/OSD)
BS2000-Dateisystem bs2fs
Benutzerhandbuch

[3] **POSIX** (BS2000/OSD)
SOCKETS/XTI für POSIX
Benutzerhandbuch

[4] **C/C++** (BS2000/OSD)
POSIX-Kommandos des C/C++-Compilers
Benutzerhandbuch

[5] **C/C++** (BS2000/OSD)
C/C++-Compiler
Benutzerhandbuch

[6] **C-Bibliotheksfunktionen** (BS2000/OSD)
für POSIX-Anwendungen
Referenzhandbuch

[7] **CRTE (BS2000/OSD)**
Common RunTime Environment
Benutzerhandbuch

[8] **NFS** (BS2000/OSD)
Network File System
Benutzerhandbuch

[9] **SECOS** (BS2000/OSD)
Security Control System - Zugangs- und Zugriffskontrolle
Benutzerhandbuch

[10] **SECOS** (BS2000/OSD)
Security Control System - Beweissicherung
Benutzerhandbuch

[11] **BLSSERV**
Bindelader-Starter in BS2000/OSD
Benutzerhandbuch

[12] **COBOL2000** (BS2000/OSD)
COBOL-Compiler
Benutzerhandbuch

[13] **COBOL85** (BS2000/OSD)
COBOL-Compiler
Benutzerhandbuch

[14] **EDT** (BS2000/OSD)
Anweisungen
Benutzerhandbuch

[15] **EDT** (BS2000/OSD)
Unicode-Modus Anweisungen
Benutzerhandbuch

[16] **BS2000/OSD-BC**
Einführung in die Systembetreuung
Benutzerhandbuch

[17] **BS2000/OSD-BC**
Dienstprogramme
Benutzerhandbuch

[18] **openFT für BS2000/OSD**
Enterprise File Transfer in der offenen Welt
Benutzerhandbuch

[19] **openFT für UNIX**
Enterprise File Transfer in der offenen Welt
Benutzerhandbuch

[20] **openFT für UNIX**
Enterprise File Transfer in der offenen Welt
Installation und Administration
Systemverwalterhandbuch

[21] **HSMS** (BS2000/OSD)
Hierarchisches Speicher Management System
Band 1: Funktionen, Verwaltung und Installation
Benutzerhandbuch

[22] **HSMS** (BS2000/OSD)
Hierarchisches Speicher Management System
Band 2: Anweisungen
Benutzerhandbuch

[23] **IMON** (BS2000/OSD)
Installationsmonitor
Benutzerhandbuch

[24] **JV** (BS2000/OSD)
Jobvariablen
Benutzerhandbuch

[25] **SDF** (BS2000/OSD)
SDF-Verwaltung
Benutzerhandbuch

[26] **SDF-A** (BS2000/OSD)
Benutzerhandbuch

[27] **BS2000/OSD-BC**
Einführung in die Systembetreuung
Benutzerhandbuch

[28] **BS2000/OSD-BC**
Kommandos
Benutzerhandbuch

[29] **DSSM /SSCM**
Verwaltung von Subsystemen in BS2000/OSD
Benutzerhandbuch

[30] **BS2000/OSD-BC**
Makroaufrufe an den Ablaufteil
Benutzerhandbuch

[31] **SDF-P** (BS2000/OSD)
Programmieren in der Kommandosprache
Benutzerhandbuch

[32] **SPOOL** (BS2000/OSD)
Benutzerhandbuch

[33] **Spool & Print - Kommandos** (BS2000/OSD)
Benutzerhandbuch

[34] **SORT** (BS2000/OSD)
Benutzerhandbuch

[35] **AID** (BS2000/OSD)
Testen unter POSIX
Ergänzung

[36] **AID** (BS2000/OSD)
Testen von C/C++-Programmen
Benutzerhandbuch

[37] **WebServe** (BS2000/OSD)
WWW-Server auf BS2000/OSD
Benutzerhandbuch

[38] **WebTransactions**
Anschluss an openUTM-Anwendungen über UPIC
Benutzerhandbuch

[39] **WebTransactions**
Anschluss an OSD-Anwendungen
Benutzerhandbuch

[40] **interNet Services** (BS2000/OSD)
Benutzerhandbuch

[41] **interNet Services** (BS2000/OSD)
Administratorhandbuch

[42] **SNMP Management**
SNMP Management für BS2000/OSD
Benutzerhandbuch

Stichwörter