Universität Dortmund
Lehrstuhl Informatik XII
44221 Dortmund

Endbericht

Projektgruppe 391

NetFire

Entwicklung eines CD-Brenners mit Netzwerkschnittstelle

Sommersemester 2002

Projektgruppe 391

NetFire

Entwicklung eines CD-Brenners mit Netzwerkschnittstelle

Endbericht

15. Juli 2002

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

1.1 PG 391 - NetFire

1.1.1 Thema

NetFire - Entwicklung eines CD-Brenners mit Netzwerkschnittstelle

1.1.2 Zeitraum

Wintersemester 2001/2002 und Sommersemester 2002

1.1.3 Umfang

Jeweils 8 Semesterwochenstunden

1.1.4 Veranstalter

Universität Dortmund
Fachbereich Informatik
Lehrstuhl 12 - Technische Informatik und Eingebettete Systeme
Dipl.-Inform.(FH),Ing. Jens Wagner
Dipl.-Inform. Heiko Falk

1.1.5 Internetadresse

`http://ls12-www.cs.uni-dortmund.de/netfire/`

1.1.6 PG-Aufgabe

1.1.6.1 Motivation

Betrachtet man die Tendenzen bei der Entwicklung von Software- und Hardware-Produkten im Laufe der letzten Jahre, so kann generell festgestellt werden, daß Software mitsamt zugehörigen Daten immer komplexer und umfangreicher wird. Aus diesem Umstand resultiert die Notwendigkeit nach einem Medium zur Datensicherung, das große Datenvolumen schnell, dauerhaft und preiswert verarbeiten kann.

Zusätzlich ist eine erhöhte Mobilität in zweierlei Hinsicht zu beobachten: zum einen werden immer mehr Daten zwischen Benutzern ausgetauscht, wobei eine rein elektronische Form dieses Austausches via Internet nicht immer möglich ist, so daß der Transfer häufig nach wie vor über physikalische Datenträger vorgenommen wird. Andererseits werden Computersysteme zunehmend mobil eingesetzt, wobei verschiedenste Rechner vernetzt in den unterschiedlichsten heterogenen Umgebungen zum Einsatz kommen. Diese Rechner verfügen heute nur in den seltensten Fällen über Geräte zum Speichern großer Datenvolumina. Wenn, dann handelt es sich meist um spezielle Lösungen, welche mit Geräten im Bereich des privaten Einzelnutzers nicht kompatibel sind, und deshalb einen Datenaustausch verhindern.

Die Kombination der Faktoren Datensicherheit, Preis für Speichermedien, Datenaustausch über Wechselmedien und Mobilität legt nahe, einen universellen CD-Brenner zu entwickeln, der in beliebigen vernetzten heterogenen Rechner-Umgebungen einsetzbar ist.

Gerade die Suche nach der Möglichkeit, ein Gerät von einer Vielzahl von Rechnern aus zu betreiben, stößt in der Praxis bislang auf unüberwindbare Schwierigkeiten. Dies liegt zum einen daran, daß für jedes Gerät eigene betriebssystemspezifische Treiber benötigt werden, wobei häufig viele gängige Betriebssyteme nicht unterstützt werden. Andererseits sind die gängigen Standards zum Anschluß von Peripheriegeräten, wie CD-Brenner aneinem Rechner, nicht für den Einsatz in heterogenen vernetzten Umgebungen geeignet, wie die folgende Übersicht verdeutlicht:

SCSI [1]: 16 Bit breiter Datenbus zum Anschluß von max. 15 Geräten am einem Kontroller bei einer max. Buslänge von drei Metern. Es werden keine Treiber benötigt, aber die Adressen der Gräte werden mechanisch codiert, Hot Plug-and-Play ist nicht möglich, die mechanische Verbindung ist wegen diverser verschiedener Steckverbinder und Terminatoren sehr aufwendig und unflexibel.

PCMCIA [2]: Wird hauptsächlich im Notebook-Bereich eingestzt, so daß Desktop-Rechner i.d.R. nicht unterstützt werden. Der Standard definiert 6 verschiedene Bauformen, so daß mechanische Inkompatibilitäten gegeben sind. PCMCIA erlaubt Hot Plug-and-Play, wobei allerdings lediglich eine 1-zu-1-Verbindung zwischen Gerät und PCMCIA-Anschluß gegeben wird. Der Einsatz spezifischer Treiber ist notwendig.

USB/USB2 [3]: Mit Hilfe von Hubs können bis zu 127 USB-Geräte an einen Rechner angeschlosssen werden. Hot Plug-and-Play ist möglich, wobei dies durch das Betriebssystem unterstützt werden muß. Plattformabhängige Treiber sind notwendig. Der größte Nachteil ist, daß ein USB-Gerät nur von einem Rechner aus genutzt werden kann.

Firewire [4]: Erlaubt den Anschluß von bis zu 63 Geräten via Hot Plug-and-Play, wobei mittels Daisy-Chaining ein Gerät sogar von mehreren Rechnern aus genutzt werden kann. Jedoch müssen geräteabhängige Treiber installiert sein, der Standard unterliegt aufgrund seiner Neuheit noch starken Modifikationen und wird nur von wenigen Rechner-Plattformen unterstützt.

Ethernet [5]: Durch Einsatz von Hubs können beliebige Kaskadierungen aufgebaut werden, so daß beliebig viele Rechner mit beliebig vielen CD-Geräten verbunden werden könnten. Komponeten können während des laufenden Betriebs hinzugefügt und entfernt werden. Im allgemeinen kann man davon ausgehen, daß jede Plattform Zugang zu einem Ethernet-Anschluß hat. Alle Betriebssysteme und Hardware-Plattformen unterstützten jetzt und in Zukunft den Ethernet-Standard.

Die aufgeführten Eigenschaften der verschiedenen Schnittstellen läßt nur Ethernt zu, um ein plattformübergreifendes und beliebig skalierbares Interface zu erhalten.

1.1.6.2 Stand der Technik

Compact Disk

Nach dem Legen der wissenschaftlichen Grundlagen [6, 7] erfuhren Compact Disk (CD)-basierte Multimedia- und Computersysteme eine relative schleppende Markteinführung. Ein Grund waren die astronomischen Preise der Geräte und Medien. Die ersten Geräte im Consumer Markt waren für das Abspielen von Musik entwickelt. Mit Hilfe eines globalen Standards gelang es, die Akzeptanz der Geräte in Folge von gesteigerter Investitionssicherheit und Kompatibilität zu erhöhen. Die Firmen Sony und Philips waren federführend in der Entwicklung der Standards. Bereits 1980 legten sie die physikalischen Abmessungen und die Datenstruktur auf der Audio CD (CD-DA) im Red Book fest. Die Daten werden auf einer Scheibe von 120mm Durchmesser und 1,2mm Dicke gespeichert. Die CD besteht aus zwei durchsichtigen Plastikscheiben, zwischen denen eine teilreflexive Schicht aus Aluminium aufgebracht ist. Die Informationen sind mit Hilfe von Erhebungen oder Löchern in der Reflextionsschicht kodiert. Ähnlich einer Schallplatte ist die Datenspur spiralförmig um den Mittelpunkt angeordnet, nicht etwa in kreisförmigen Sektoren, wie bei magnetischen Datenträgern.

Erste Geräte, die Daten von einer Compact Disk (CD-ROM) lesen konnten, erreichten den europäischen Markt 1987 [8]. Mit der Verfügbarkeit von Lesegeräten wurde die Kluft zwischen den damaligen Archivierungsmöglichkeiten deutlich. Schließlich betrug die Kapazität einer gewöhnlichen Harddisk 20 MByte - die Kapazität einer damaligen CD-ROM immerhin 650 MByte. Entsprechend wurde die Entwicklung beschreibbarer Medien fokusiert [9]. Bereits 1988 waren die benötigten Materialien entwickelt, um CDs einmal zu beschreiben [10, 11]. In weiteren Arbeiten der Folgejahre [12, 13, 14, 15] wurde die Schreibgeschwindigkeit und Datendichte erhöht.

Datenübertragung via Ethernet

Geräte, die Daten auf CD schreiben können, verlangen während des gesamten Schreibvorgangs einen Datenstrom mit einem bestimmten Durchsatz. Ein CD-Brenner hat i.Allg., wenn er das Schreiben der Datenspur abbricht, keine Möglichkeit zum Wiederansetzen. Nun liegt es in der Natur des Ethernetprotokolls, daß auch ein minimaler Datendurchsatz nicht garantiert werden kann. Würde ein CD-Brenner die Schreib-Daten via Ethernet empfangen, wäre ein außerordentlich unzuverlässiges Gerät die Folge. Es gibt zwar neuere Ansätze, dieses Problem durch Zusätze zum Ethernetprotokoll zu lösen [16], man verliert so aber die Kompatibilität zu bestehenden Systemen. Interessanter ist die Lösung von Plextor, ein beliebiges Abbrechen und Wiederaufsetzen des Schreibvorgangs zuzulassen. Diese Technologie wird als Brun-Proof [17] bezeichnet.

1.1.6.3 Ziel

Im Rahmen der Projektgruppe sollte ein CD-Brenner ein Ethernet-Interface erhalten, so daß dieser plattformübergreifend (z.B. zeitgleich von Sun Workstations und PCs aus) ferngesteuert werden kann.

Da sich der Einsatzbereich eines CD-Brenners auf den Transfer von Dateien beschränkt, sollte der Ethernet-fähige CD-Brenner über einen integrierten FTP-Server [18] angesteuert werden. Zum Brennen muß auf einem beliebigen Rechner zunächst ein ISO CD-Image erzeugt werden. Dieses wird dann via FTPan den CD-Brenner übermittelt.

Dieses Konzept bietet den Vorteil maximaler Plattformumabhängigkeit, da zum einen jeder Rechner, der über einen Ethernet-Anschluß verfügt, auch das FTP-Protokoll unterstützt. Die Verbindung zum CD-Brenner von einem Rechner aus kann somit leicht über beliebige kostenlose Internet-Browser wie z.B. Netscape hergestellt werden. Weiterhin existiert frei verfügbare komfortable Software für alle Arten von Betriebssystemen zum Erzeugen des ISO-Images einer CD (z.B NeroBurn, xcdroast). Die zum Einsatz kommende Software zum Betrieb des CD-Brenners ist also vollkommen geräteunabhängig, auf keinem Rechner ist irgendeine Form von Treibern zu installieren. Während der PG wurde dieses Ziel in der Form erweitert, daß nun das Erzeugen des Iso-Images auf dem Quellrechner nicht mehr vorausgesetzt wird. Vielmehr sollte auch die Möglichkeit bestehen, beliebig viele Dateien mit einer Verzeichnisstruktur komfortabel und direkt zum CD-Brenner zu übertragen.

Es ist eine Lösung entstanden, die zum Nachbau einlädt. Sämtliche Ergebnisse sind im Internet veröffentlicht (siehe Kapitel 1.1.5) und bieten Interessierten die Möglichkeit zum Aufbau einens eigenen Brenners.

Die Arbeit der Projektgruppe läßt sich grob in die folgenden Teilaufgaben untergliedern:

- **Simulation auf einem PC:**
  Die Software wurde auf einem PC simuliert. Die Algorithmen wurden in C implementiert.
- **Implementierung in der Zielhardware:**
  Die getestete Software wurde auf unserer Zielhardware implementiert. Es erfolgten Test und Inbetriebnahme.
- **Erstellung technologischer Unterlagen:**
  Um die Möglichkeit des Nachbaus und der späteren Erweiterung zu erhalten, sind Unterlagen erstellt worden, die es ermöglichen, das Projekt später exakt nachzuvollziehen.

1.1.7 Teilnahmevoraussetzungen

Folgende Kenntnisse waren notwendig für die Teilnahme an der PG:

- Vorlesung „Rechnerarchitektur“ oder „Rechnergestützter Entwurf/Produktion“
- Programmierkenntnisse in C
- Die im Projekt verwendeten Bauelemente und Baugruppen sind in englischer Sprache dokumentiert. Es war unverzichtbar, daß jedes PG-Mitglied in der Lage ist, sich den Inhalt englischsprachiger Dokumente zu erarbeiten.

Weiterhin wünschenswerte, aber nicht notwendige Voraussetzungen:

- Kenntnisse von Internet-Protokollen (TCP, IP, ARP, FTP)
- Erfahrung in Aufbau von Hardwareschaltungen und im Umgang mit Laborausstattungen
- Vorlesungen „Prozeßrechnertechnik“, „Rechnernetze“
- Kenntnisse in Assemblerprogrammierung
- Grundkenntnisse im Schaltungsaufbau der E-Technik

1.1.8 Minimalziel

1. Erfolgreiche Analyse der im Bereich von CD-Brennern eingesetzten Varianten des ATAPI-Protokoll. Hierzu gehört insbesondere die
   - Identifizierung verschiedener ATAPI-Betriebsmodi
   - Auswahl möglichst einfacher für die Brenner-Applikation geeigneter Betriebsmodi,
   - Dokumentation der vorigen Schritte.
2. Implementierung einer funktionsfähigen ATAPI-Steuerungssoftware auf der Grundlage der erfolgten Protokollanalyse. Die Funktionsfähigkeit der Steuerungssoftware soll auf einer gängigen PC-Plattform mit extern angeschlossenem CD-Brenner demonstriert werden.
3. Realisierung eines Ethernet-fähigen Benutzer-Interface in Form eines für den Einsatzbereich abgewandelten FTP-Server.
4. Portierung der PC-basierten Software auf eine geeignete Zielhardware.

1.2 Teilnehmer

Name	Hauptfach	Nebenfach	Fachsemester
Dmitri Barski	Informatik	E-Technik	12
Martin Beckmann	Ingeniuerinformatik	E-Technik	14
Heiko Bihr	Informatik	BWL	10
Markus Göbbels	Ingenieurinformatik	E-Technik	14
Stefan Hüsken	Ingenieurinformatik	E-Technik	12
Jörg Kleinophorst	Informatik	E-Technik	10
Konstantin Koll	Informatik	Theoretische Medizin	8
Patrick Nagelschmidt	Informatik	BWL	10
Christian Rehahn	Informatik	BWL	10
Mark Rzepka	Informatik	E-Technik	8
Gregor Standers	Informatik	E-Technik	10
Michael Vogt	Informatik	Philosophie	10

1.3 Nutzungsvereinbarung

Die Projektteilnehmer haben sich entschieden das ganze Projekt im Einvernehmen mit den Richtlinien der GNU General Public License [19] (deutsche Übersetzung im Anhang) zu erstellen. Alle entwickelte Software im Rahmen dieser Projektgruppe ist somit freie Software, d.h. es ist erlaubt, die Software weiterzugeben und zu verändern.

1.4 Struktur des Berichts

In Kapitel 2 wird vorgestellt welche Möglichkeiten das NetFireSystem hinsichtlich der Bedienerführung, der möglichen Brennformate und der Optionen bietet. Kapitel 3 beschäftigt sich mit der frei zur Verfügung stehenden Standartsoftware. Es werden verschiedene Betriebssysteme und Programme auf ihre Nutzbarkeit für NetFirehin untersucht. In Kapitel 4 wird auf die eigenentwickelte Software wärend dieser Projektgruppe eingegannen. Dazu werden die einzelnen Komponenten vorgestellt und im einzeln Beschrieben. In Kapitel 5 wird die verwendete Hardware erläutert und beschrieben welche Kriterien zu ihrer Auswahl führten Kapitel 6 beschäftigt sich mit dem Testen von NetFireEinen Überblick über die von uns angefertigte Archiv-CD wird in Kapitel 7 gegeben. In Kapitel 8 wird beschrieben wie NetFireinstalliert. Zum Schluß wird in Kapitel 9 ein Fazit gezogen.

Kapitel 2

Features des NetFire-Systems

Das NetFire-System verfügt über eine Vielzahl von Features, die den Benutzer beim Erstellen einer CD unterstützen sollen. Die grundlegenden Features des Gesamtsystems sind:

- Dem Benutzer wird eine WebGUI zur Verfügung gestellt.
- Das System kann man auch per FTP-Kommandozeile bedienen.
- Es werden verschiedene Brennformate unterstützt.
- Ein burnproof-Brenner sorgt dafür, daß keine buffer underruns eintreten können.
- In das System ist ein LCD intergriert worden, das über zwei Funktionstasten verfügt.
- Das NetFire-Gerät kann durch Konfiguration per LCD-Tasten in beliebige Netzwerke eingebunden werden.
- Die kompakte, praktische Würfelform spart Platz und ermöglicht einen vielseitigen Einsatz.
- Es gibt ein Archiv der Images, um wiederholte Uploads zu vermeiden.
- Die Unterstützung von CDRW ist vorhanden.

Nachdem der User einen Brennauftrag generiert hat, wird dieser in die NetFire-Warteschlange eingereiht. Der erste Auftrag in der Queue wird direkt gebrannt, während der zweite Job für den eigentlichen Brennvorgang vorbereitet wird (eine gepackte Datei muß z.B. entpackt werden). Ist der aktuelle Brennauftrag abgearbeitet worden, wird dieser aus der Warteschlange entfernt, alle anderen Jobs rücken nach. NetFire verfügt auch über Administrationsfunktionen, die unter anderen auch die Möglichkeit zur Verfügung stellen, Brennaufträge aus der Queue zu entfernen.

2.1 Bedienung

Es gibt grundsätzlich zwei Möglichkeiten über ein beliebiges Netz, in das NetFire eingebunden wurde, das Systen zu bedienen. Zu einem gibt es die komfortable WebGUI, mit deren Hilfe der User mittels eines HTTP-Browsers eine CD per Mausklick erstellen kann. Die WebGUI bietet dem Benutzer viele hilfreiche Funktionalitäten an. Unter anderen werden die Stati der Brennaufträge graphisch angezeigt, desweiteren gibt es die Möglichkeit einige Netzwerk- und Systemeinstellungen in dem Administrationsbereich durchzuführen. Neben der Bedienung über die Web-Oberfläche gibt es die Möglichkeit das NetFire-System über eine Konsole zu nutzen. Eine FTP-Kommandozeile hat den Vorteil, daß man keinen HTTP-Browser braucht, um eine CD zu erstellen.

2.1.1 Bedienung per WebGUI

2.1.1.1 Die Startseite

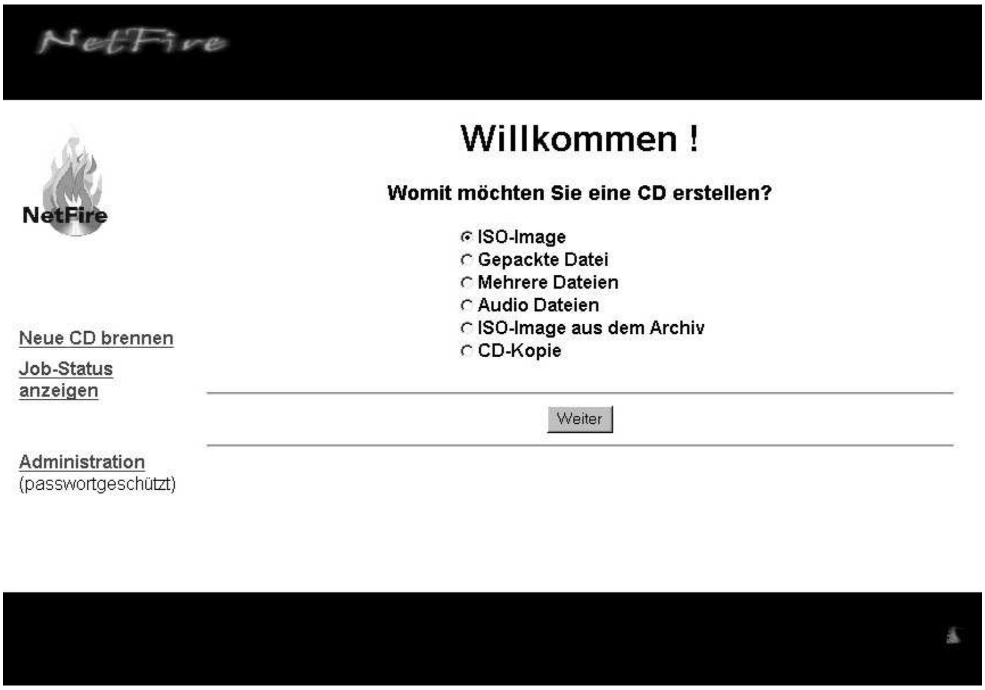

Abbildung 2.1: Die Startseite der WebGUI

Nachdem der Benutzer die entsprechende URL oder IP-Adresse in einem Web-Browser eingegeben hat, erscheint die NetFire-Willkommensseite. Auf dieser Startseite hat man die Möglichkeit, zwischen mehreren Vorgehensweisen zu wählen, abhängig davon, wie und welche Art von CD man brennen möchte bzw. welche Art von Daten man uploaden will. Im Detail stehen dem User folgende Auswahlmöglichkeiten zur Verfügung, die man mittels der Radio-Buttons betätigen kann:

- `ISO-Image`: Hiermit kann der Benutzer ein eigenes ISO-Image uploaden und brennen.
- `Gepackte Datei`: Eine gepackte Datei (z.B. ein tar-Archiv oder eine zip-Datei) kann upgeloadet werden. Sie wird automatisch entpackt und aus dem Inhalt wird ein ISO-Image erzeugt, das dann gebrannt wird.
- `Mehrere Dateien`: Auch einzelne Dateien oder auch Verzeichnisse (je nach Browser-Möglichkeiten, siehe Kapitel 3.1.3) können, nachdem aus diesen ein ISO-Image erzeugt wurde, gebrannt werden.
- `Audio Dateien`: Damit hat der User die Möglichkeit, eine Audio-CD zu erstellen. Es können mp3- und wav- Dateien upgeloadet und gebrannt werden.
- `ISO-Image aus dem Archiv`: Ein zu brennendes ISO-Image wird aus dem vorhandenen Archiv ausgewählt und gebrannt.
- `CD-Kopie`: Eine Kopie einer schon vorhandenen CD wird erstellt.

Nachdem sich der Benutzer für einen der sechs Menüpunkte entschieden hat, gelangt er mittels des `Weiter`-Buttons zum nächsten Schritt des Brennvorgangs, der je nach Menüpunkt unterschiedlich ausfallen kann.

Links unter dem NetFire-Logo sind drei Links sichtbar, die auch auf jeder folgenden Seite zu finden sind. Auf die Funktionalität dieser und die dahinter verborgenen Möglichkeiten wird weiter unten noch eingegangen.

2.1.1.2 Der FTP-Upload

Die Auswahl eines der ersten vier Menüpunkte auf der Startseite und das anschließende Betätigen des `Weiter`-Buttons hat zur Folge, daß mittels einer neuen Seite der Benutzer aufgefordert wird, ein FTP-Upload-Fenster zu öffnen. Dies geschieht indem er den Link `FTP-Upload` betätigt. Zu diesem Zeitpunkt wird auch für diese Brennsitzung gültiger Account mitgeteilt. Mittels des `Zurück`- oder auch `Abbruch`-Buttons gelangt der User zur Startseite. Sein Account wird in diesem Fall gelöscht.

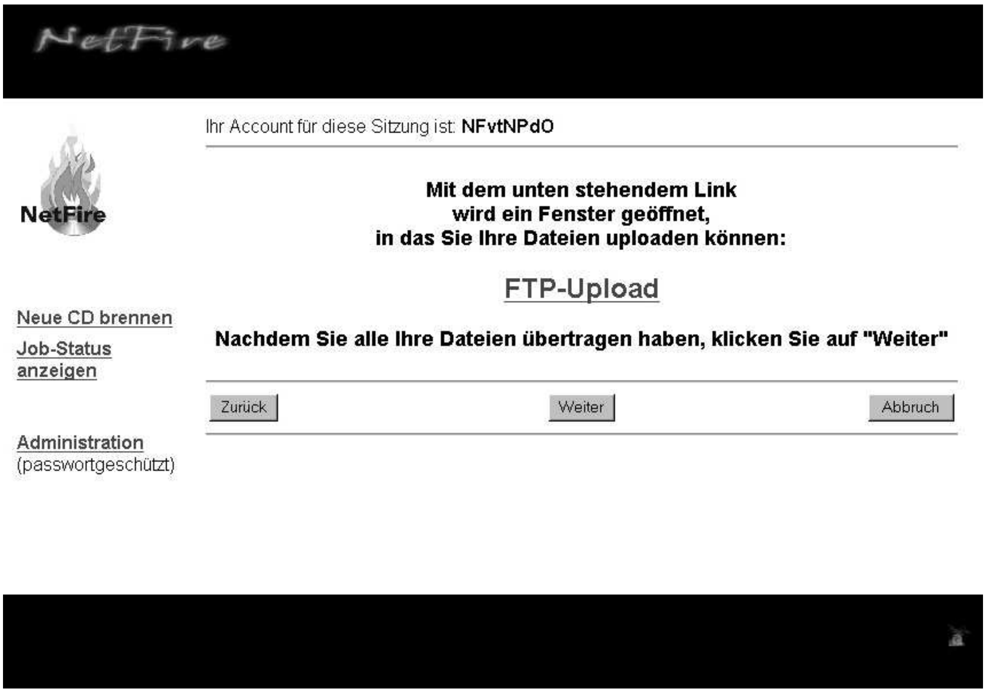

Abbildung 2.2: Link zum FTP-Upload

Nachdem das FTP-Upload-Fenster geöffnet wurde, können die Dateien auf den FTP-Server übertragen werden. Es wird automatisch das Verzeichnis angezeigt, in das die Files upgeloadet werden sollen. In Abhängigkeit von dem benutzten HTML-Browser geschieht das mittels Drag&Drop (z.B. Konqueror) oder der Auswahl entsprechender Menüpunkte in der Menübar (z.B. Netscape). Auf welche Weise und ob überhaupt ein anderer Browser den Upload unterstützt, kann dem Kapitel 3.1.3 entnommen werden.

Wenn das FTP-Upload abgeschlossen ist, kann das FTP-Upload-Fenster geschlossen werden. Mit einem Klick auf den `Weiter`-Button gelangt man zur nächsten Seite, auf der die gewünschten Brenneinstellungen vorgenommen werden können.

2.1.1.3 Die Brenneinstellungen-Seite

Auf dieser Seite befindet sich eine Reihe von Einstellungsmöglichkeiten, die den darauffolgenden Brennvorgang betreffen. Dazu gehören folgende Optionen:

- `Geschwindigkeit`: Die möglichen Optionen reichen von der 1-fachen bis 32-fachen Geschwindigkeit, wobei ab “2-fach” bis “12-fach” die Geschwindigkeitseinstellung in Zweier-Schritten und dann in Vierer-Schritten wächst. Man kann auch die Option “max” auswählen, wobei diese vom eingesetzten Brenner und/oder Rohling abhängig ist. In diesem Fall wird mit der maximal möglichen Geschwindigkeit gebrannt.

- `Exemplare`: Diese Einstellung dient der Angabe der gewünschten Anzahl von Exemplaren der gebrannten CD. Es können ein, zwei oder drei Exemplar(e) der selben CD erstellt werden.

- `CD-RW löschen`: Hiermit wird mitgeteilt, daß die CD, auf die gebrannt werden soll, eine wiederbeschreibbare CD ist, die vor dem Brennen noch gelöscht werden soll. Es kann entweder die "Table Of Contents" (TOC) der CD gelöscht werden, was der Option "schnell" entspricht, oder die ganze CD (Option "alles"). Dabei sollte der Unterschied in der Dauer des Löschvorgangs berücksichtigt werden. Bei der Voreinstellung "nein" wird der Brennvorgang gestartet ohne die CD zu löschen.

- `Mode`: Ein erfahrener Benutzer kann hier zwischen mehreren Modi wählen. Ein Modus schreibt die Anzahl der Bytes in einem Sektor vor. Die einzelnen Modi sind in dem Beispiel für eine Job-Control-Datei beschrieben (siehe Abbildung **??**).

- `Multisession`: Mittels dieser Checkbox kann die Option wahrgenommen werden, eine multisessionfähige CD zu erstellen.

- `Bootfähig`: Analog zu der oberen Option kann man hierdurch eine bootfähige CD erstellen. Wurde diese Option ausgewählt, wird dem User im nächsten Schritt nahegelegt, ein Boot-Image zu übertragen. Dieser Vorgang wird noch weiter unten näher erläutert.

- `Ins Archiv`: Mit dieser Einstellung werden die zu brennenden Dateien (außer im Fall einer Audio-CD oder einer CD-Kopie) als ein ISO-Image nach dem eigentlichen Brennvorgang ins Archiv abgelegt. Dieses Image kann man zu einem späteren Zeitpunkt nochmals brennen.

- `Kommentar`: Falls der Benutzer die Option "Ins Archiv" gewählt hat, muß er einen beschreibenden Kommentar zu dem späteren ISO-Image in das dazu vorgesehene Feld schreiben. Andernfalls wird er aufgefordert, dies nachzuholen.

- `E-Mail`: Wenn eine Benachrichtigung per E-Mail nach dem abgeschlossenen Brennjob erwünscht ist, muß die Mail-Adresse in dem Textfeld angegeben werden.

Es werden nur die Einstellungsmöglichkeiten angezeigt, die für den jeweiligen Brennauftrag auch einen Sinn machen. So kann z.B. ein ISO-Image aus dem Archiv nicht nochmals ins Archiv verschoben werden. Eine Audio-CD kann ebenfalls nicht ins Archiv gestellt werden, da dieses nur aus ISO-Images besteht.

Mit Hilfe des `Zurück`-Buttons gelangt der User zu der Seite, auf der er mittels des FTP-Upload-Links wieder auf sein FTP-Verzeichnis auf dem Server Zugriff hat. Somit hat er die Möglichkeit weitere Datei-Uploads durchzuführen. Beim Klicken auf den `Abbruch`-Button gelangt der User zur Startseite. Sein Account sowie alle übertragenen Dateien werden in diesem Fall gelöscht. Mit dem `Weiter`-Button wird ein Brennauftrag generiert und in die Warteschlange eingefügt. Danach erscheint die Statusseite seines Jobs (siehe unten). Möchte der User eine bootfähige CD erstellen, wird zuvor eine Seite angezeigt, auf der er das Boot-Image auswählen kann (mehr dazu im nächsten Unterkapitel).

2.1.1.4 Boot-Image-Upload

Wählt der Benutzer die Option zur Erstellung einer bootfähigen CD, so kann er auf einer separaten Seite das gewünschte Boot-Image auf den Server übertragen. Anders als bei den

Abbildung 2.3: Die Brenneinstellungen-Seite

übrigen Uploads geschieht dies nicht mittels FTP sondern per HTTP-Upload. Dazu muß er mit Hilfe eines Datei-Browsers, der ihm nach Betätigen des Browse- bzw. Durchsuchen-Buttons zur Verfügung steht, ein solches Image auf seinem Rechner auswählen. Mit `Weiter` wird der Vorgang der Übertragung gestartet und anschließend die Status-Seite angezeigt.

Abbildung 2.4: HTTP-Upload des Boot-Images

Der `Zurück`-Button führt zu der Brenneinstellungen-Seite, `Abbrechen` löscht den Account und alle übertragenen Dateien und führt zu der Startseite.

2.1.1.5 Audio-Dateien

Der Vorgang, der Erstellung einer Audio-CD läuft bis auf wenige Ausnahmen analog zu den anderen, oben beschriebenen Brennvorgängen. Nach dem FTP-Upload der Audio-Dateien (.wav oder .mp3) kann die Reihenfolge der zu brennenden Tracks bestimmt bzw. verändert werden. Zu jeder Track-Position kann in dem entsprechenden Select-Menü die jeweilige Audio-Datei ausgewählt werden. Es kann auch die Anzahl der Audio-Tracks bestimmt werden. Dazu dienen die Buttons `Track löschen` und `Track hinzufügen`. Auch hier sind die schon oben beschriebenen `Zurück`- und `Abbrechen`-Buttons verfügbar. Mit dem `Weiter`-Button gelangt man zu der Brenneinstellungen-Seite, wobei zu beachten ist, daß danach die Reihenfolge der Tracks nicht mehr geändert werden kann, d.h. der `Zurück`-Button steht nicht mehr zur Verfügung.

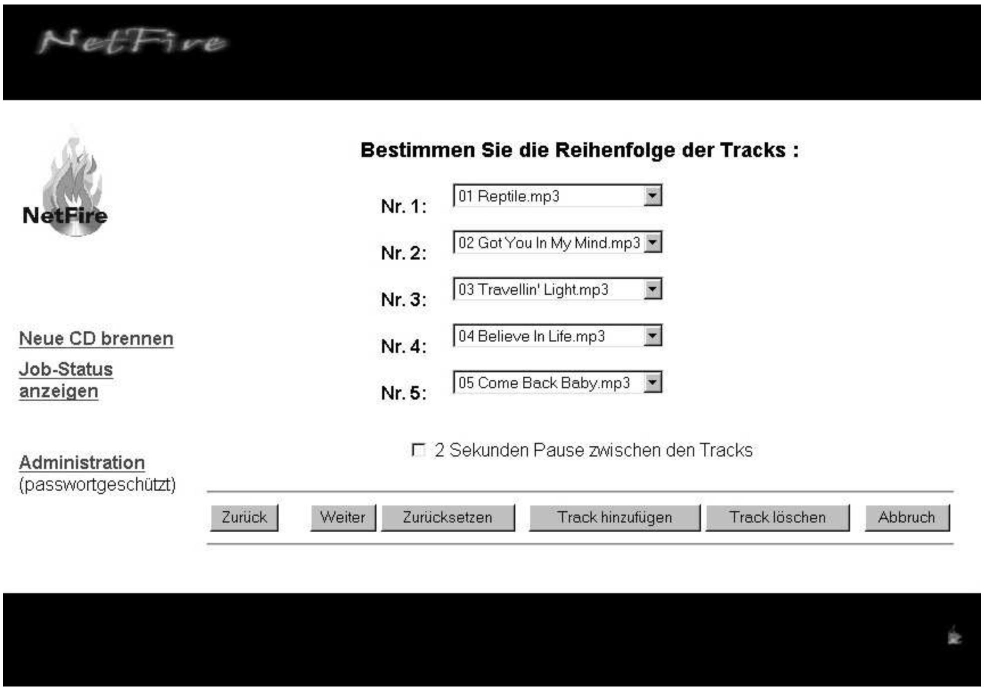

Abbildung 2.5: Die Bestimmung der Reihenfolge der Audio-Tracks

Als Option wird eine Checkbox angeboten, mit der zwei Sekunden Pause zwischen den einzelnen Tracks eingefügt werden kann.

2.1.1.6 ISO-Image aus dem Archiv brennen

Nach der Auswahl des entsprechenden Menüpunktes in der Start-Seite gelangt man zu einer Seite, auf der die Auswahl des gewünschtes ISO-Images betätigt werden soll. Es werden in einer Tabelle alle sich im Archiv befindenden Images aufgeführt. Zu jedem Eintrag gibt es die folgenden Informationen: Die fortlauende Nummer, das Datum der Erstellung sowie eine kurze Beschreibung. Ein Eintrag wird ausgewählt, indem die Nummer, die als ein Link realisiert wurde, angeklickt wird. Dieser Link führt den Benutzer direkt zu der Brenneinstellungen-Seite. Die Buttons `Zurück` und `Abbruch` führen zu der Startseite zurück.

2.1.1.7 CD-Kopie

Bei der Wahl dieses Menüpunktes wird der User sofort zu den Brenneinstellungen-Seite, die nur dem Kontext entsprechend wenige Optionen zur Auswahl bietet, geführt. Zu den entsprechenden Zeitpunkten wird er dazu aufgefordert, die Orginal-CD bzw. die noch leere CD ins Laufwerk einzulegen.

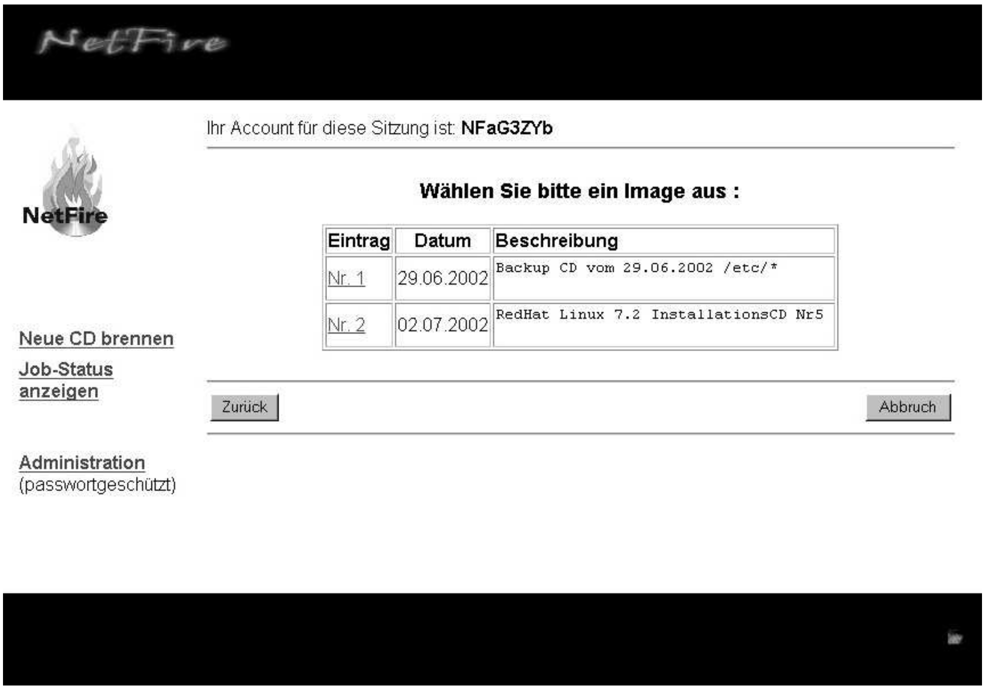

Abbildung 2.6: Anzeige des Inhalts des Archivs

2.1.1.8 Die Job-Status-Seiten

Nachdem der Brennauftrag fertiggestellt und angenommen wurde, d.h. die Brenneinstellungen vorgenommen wurden und eventuell ein Boot-Image upgeloadet wurde, wird dem Benutzer anschliessend eine neue Seite mit dem aktuellen Job-Status angezeigt. Als wichtige Informationen werden ganz oben die Job-ID sowie der zugehörige Account angezeigt. Darunter befindet sich die eigentliche Statusanzeige des Brennauftrags. Sobald der CD-Brenner mit der Abarbeitung des Jobs begonnen hat, erscheint ein Fortschrittsbalken sowie eine Prozentanzeige, die den Fortschritt des Brennvorgangs wiederspiegeln. Zu diesem Zeitpunkt ist es auch nicht mehr möglich, den Brennauftrag mittels des `Löschen`-Buttons zu entfernen. Außer den gerade erwähnten, gibt es auf dieser Seite einen `Aktualisieren`-Button, der dazu dient die Seite auf den aktuellen Stand zu bringen. Nach der Abarbeitung des Jobs wird dies auch dem User mitgeteilt.

2.1.1.9 Navigationsframe

Die schon am Anfang erwähnten Links auf der Willkommensseite helfen dem User, indem sie ihn zu den drei wichtigen WebGUI-Bereichen führen. Dies sind die Erstellung einer CD, die Seiten zur Statusabfrage sowie die weiter unten beschriebenen Administrationsseiten. Mit einem Mausklick gelangt der User automatisch zu den jeweiligen Anfangsseiten.

Der Link `Neue CD brennen` führt zu der Startseite, auf der man einen neuen Brennauftrag initiieren kann.

Mittels `Job-Status anzeigen` gelangt man zu dem Bereich, in dem die Stati aller Brennaufträge in einer Tabelle aufgeführt werden. Zu jedem Eintrag gibt es die folgenden Informationen: Die fortlauende Nummer, die Zeit der Einfügung in die Warteschlange, sowie die ungefähre Zeit bis zum Beginn der Bearbeitung des jeweiligen Jobs. Ein Eintrag wird ausgewählt, indem seine Job-ID, die als ein Link realisiert wurde, angeklickt wird. Dieser Link führt den Benutzer zu der Job-Status-Seite. Der Button `Zurück` führt zu der Startseite zurück.

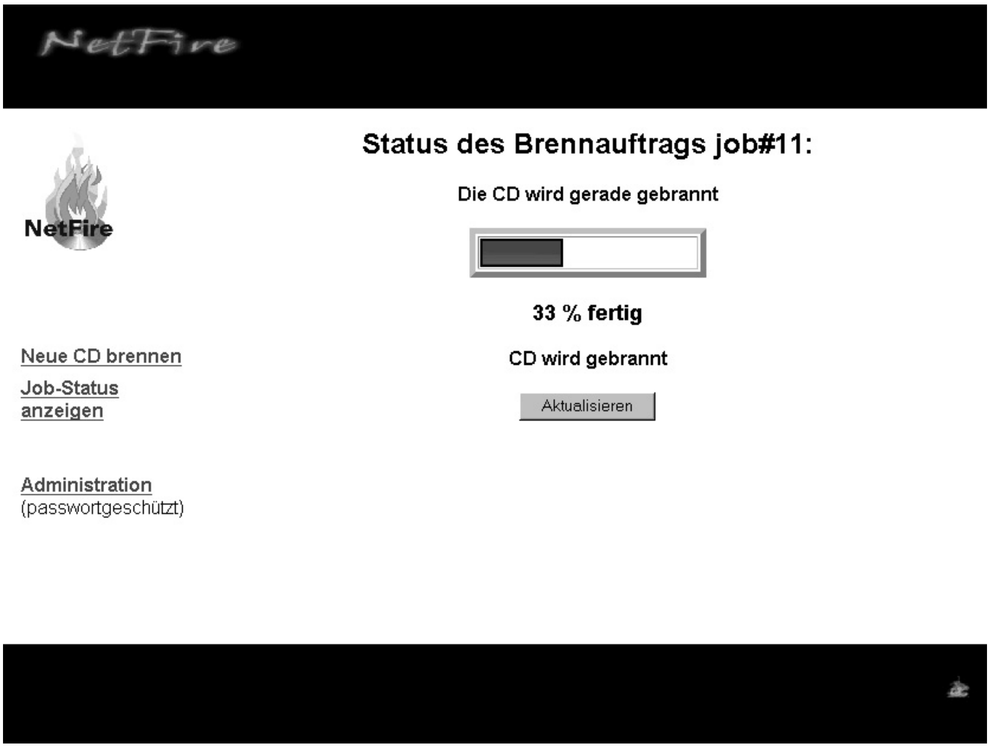

Abbildung 2.7: Beispiel einer Job-Status-Seite

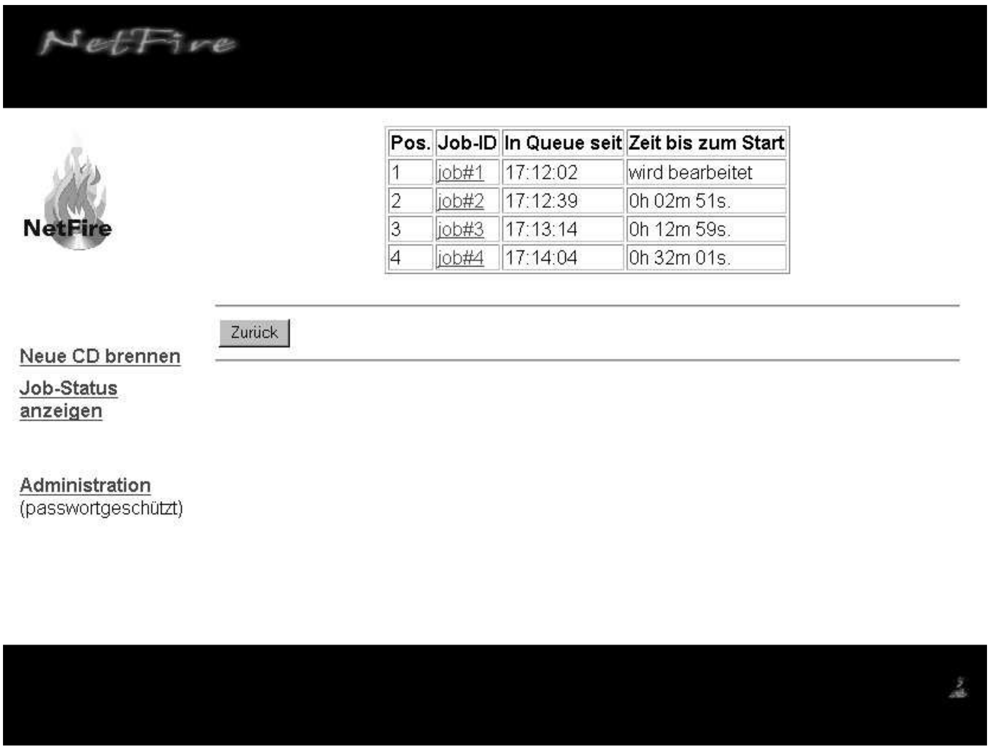

Abbildung 2.8: Die Tabelle mit den Brennaufträgen

2.1.1.10 Die Administrationsseiten

Auf der Startseite befinden sich die oben erwähnten Links, die den Benutzer zu den jeweiligen Bereichen führen. Mittels des letzten Links gelangt man zu dem Administrationsbereich, in dem man einige wichtige Einstellungen des Systems vornehmen, Job-Einträge aus der Warteschlange löschen oder das Passwort des Admins ändern kann.

Die Administrationsseiten sind mittels `.htaccess` geschützt. Nach dem Aufruf wird dem User ein Abfragefenster angezeigt. In dieses muss der Administrator als Benutzernamen `admin` eingeben. Dieser Name ist voreingestellt. Nach der Installation des gesamten Systems lautet das einzugebende Passwort `netfire`. Dieses kann später nach belieben verändert werden (siehe weiter unten).

Die Anfangsseite des Admin-Bereiches ermöglicht dem User die Wahl einer der drei Möglichkeiten, die in Form von Radio-Buttons sichtbar sind. Hat sich der Admin entschieden und den `Weiter`-Button betätigt, werden die entsprechenden neuen Seiten geladen.

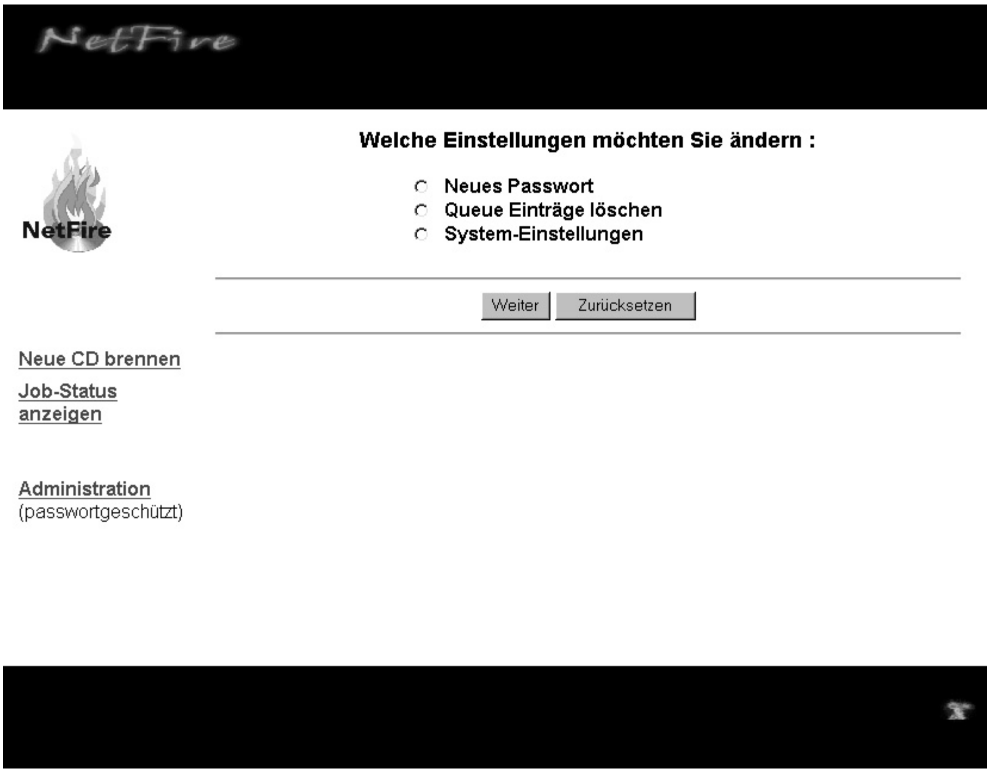

Abbildung 2.9: Die Startseite für den Administrator

Neues Passwort

Auf dieser Seite kann die Passwortänderung des Adminbereiches vollzogen werden. Dazu stehen dem User drei Textfelder zur Verfügung, in denen er zuerst das alte, dann das neue Passwort eingeben kann. Zur Sicherheit muss in dem dritten Textfeld das neue Passwort nochmals eingegeben werden. Mittels des `Zurücksetzen`-Buttons werden die Eingabefelder gelöscht. Beim Klicken auf den `Zurück`-Button gelangt man zu der vorherigen Admin-Anfangsseite. `Weiter` vollzieht die gewünschte Änderung. Falls das alte Passwort nicht stimmen sollte, erscheint die selbe Seite erneut mit einer entsprechenden Meldung.

Abbildung 2.10: Passwortänderung

Queue Einträge löschen

Die aktuelle Brenneinträge werden in einer Tabelle angezeigt. Zu jedem Eintrag werden folgende Informationen bereitgestellt: Die laufende Nummer, Account des Benutzers, die voraussichtliche Startzeit und die Uhrzeit, seit wann sich der Job in der Warteschlange befindet, sowie der Typ des Jobs. Außerdem kann der Administrator in der zweiten Spalte der Tabelle den jeweiligen Auftrag zum Löschen auswählen, indem die Checkbox markiert und der `Jobs entfernen`-Button angeklickt wird. Falls ein Job gerade gebrannt wird oder sich in der Vorbereitung befindet, wird keine Checkbox angezeigt, da es nicht mehr möglich ist, diesen zu löschen. Mit dem `Alle Jobs entfernen`-Button kann der Administrator die ganze Warteschlange der Aufträge leeren. Der Button `Aktualisieren` liest die Warteschlange erneut ein. Mit dem `Zurück`-Button gelangt man zu der Admin-Startseite.

System-Einstellungen

Hier hat der Administrator die Möglichkeit, einige wichtige Netzwerkeinstellungen, sowie die SCSI-Brenneradresse nach Bedarf zu verändern. Die einzelnen Einstellungen können der Abbildung 2.11 entnommen werden. Bei den Netzwerkeinstellungen ist eine Checkbox sichtbar, die dem Admin die Option zu Verfügung stellt einen DHCP-Server zu benutzen, falls dieser vorhanden ist, und die anderen Einstellungen übergehen. Diese werden nämlich von dem DHCP-Dienst vorgegeben. Bei der Veränderung der SCSI-Brenneradresse ist Vorsicht geboten, da fehlerhafte Einstellungen zum Zusammenbruch des Systems führen können.

2.1.2 Die FTP-Only Bedienung

Neben einer optisch ansprechenden Bedienung über die Web-Oberfläche sollte es auch mit klassischen Konsolen-Programmen möglich sein, das NetFire-System zu nutzen. Dies hat den

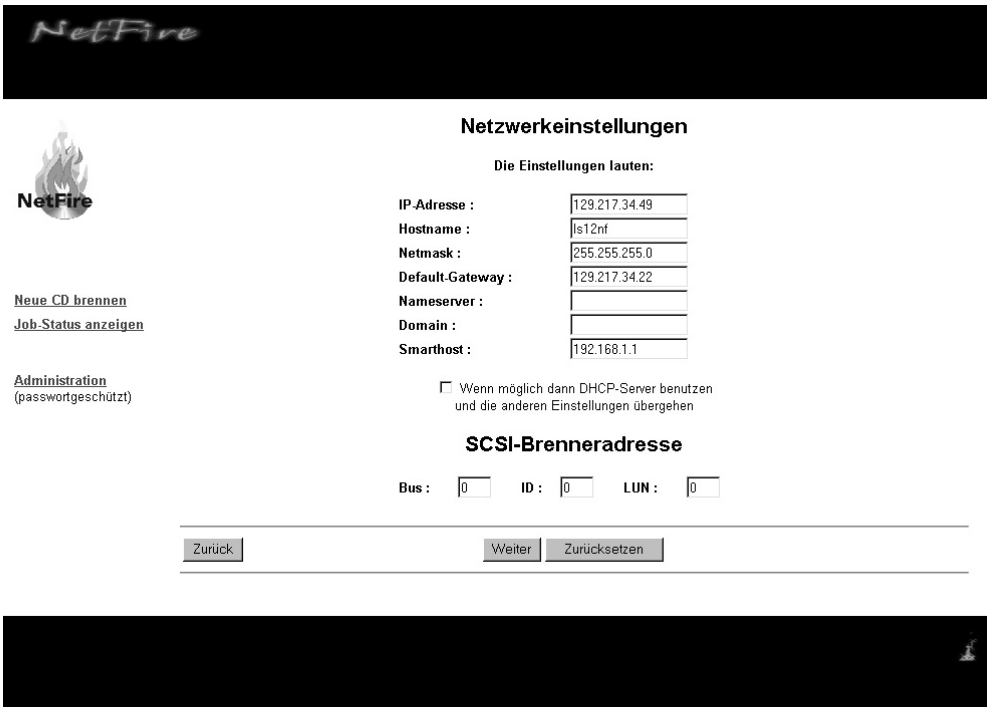

Abbildung 2.11: Vornahme der Systemeinstellungen

Vorteil, daß auch ohne einen Web-Browser, der eine vernünftige FTP-Implementierung beinhalten muß, gearbeitet werden kann. Denn wie der Vergleich der unterschiedlichen Browser gezeigt hat (siehe Tabelle 3.2 auf Seite 26), ist dies ein oftmals vernachlässigtes Feature.

Außerdem ist es in den meisten Fällen, in denen eine CD remote gebrannt werden soll, einfacher, direkt in der Konsole zu arbeiten, als erst noch die graphische Oberfläche auf den lokalen Rechner umzuleiten.

Da der Upload der Daten auch mittels WebGui über FTP erledigt wird, haben wir uns dazu entschieden, einen Teil der Funktionalität direkt über den FTP-Server zu realisieren. Die Vorgehensweise bei der Bedienung via FTP ist dabei grundsätzlich die gleiche, wie über die WebGUI.

Zunächst muß der Benutzer einen neuen temporären Account anlegen, unter dem er dann seine zu brennenden Daten ablegen kann. Dazu loggt er sich mit dem Usernamen 'new' auf dem FTP-Server ein. Das Passwort ist dabei egal. Der Benutzer bekommt nach dem Login, der bereits unter dem neu angelegten Account geschieht, seinen Usernamen mitgeteilt, wie Abbildung 2.12 zeigt:

```
ncftp> open -u new netfire
Connecting to 192.168.1.2...
Netfire FTP Server ready
Logging in...
Password requested by 192.168.1.2 for user "new".

Password:

User NFd940rl logged in.
```

Abbildung 2.12: FTP-Login

Jeder Account enthält ein Verzeichnis namens `Data`. Dieses Verzeichnis ist dazu gedacht, daß dort die Dateien abgelegt werden können, die auf die CD gebrannt werden sollen. Es ist nur in diesem Verzeichnis erlaubt, Nutzdaten abzulegen. Dies liegt daran, daß sonst eventuell nicht eindeutig zugeordnet werden kann, welche Dateien gebrannt werden sollen und welche zur Abarbeitung des Brennvorganges gedacht sind.

Nachdem der Benutzer also seine Daten in das o.g. Verzeichnis gelegt hat, muß er eine Job-Control-Datei (siehe Abbildung 4.16) erstellen, die die für den Brennvorgang relevanten Informationen enthält. Dies kann er entweder manuell machen, oder aber durch Aufruf des jtools. Das jtool ist, neben dem Parsing der Job-Control-Dateien, auch dazu in der Lage, mittels entsprechender Kommandozeilen-Parameter gültige Job-Control-Dateien zu erstellen. Diese sind in Tabelle 2.1 aufgeführt.

Parameter	Beschreibung
`-t`, `--type` *type*	Typ der zu brenennden CD
`-m`, `--writemode` *mode*	Mode der zu brenennden CD
`-o`, `--daomode` *bool*	Brennen im DAO- oder TAO-Mode
`-d`, `--deletemode` *mode*	CD-RW komplett, schnell oder gar nicht löschen
`-s`, `--speed` *number*	Brenngeschwindigkeit
`-c`, `--copies` *number*	Anzahl der Kopien
`-u`, `--multisession` *bool*	Multisession-CD
`-x`, `--fix` *bool*	Multisession-CD abschliessen
`-a`, `--archive` *bool*	Iso nach dem Brennen ins Archiv legen
`-f`, `--filename` *filename*	Pfad auf das zu brennende Image
`-i`, `--bootimage` *filename*	Pfad zum Bootimage für die CD
`-e`, `--email` *address*	Email-Addresse für die Benachrichtigungen
`-n`, `--name` *filename*	Name der zu generierenden Job-Control-Datei
`-p`, `--parse` *filename*	Name der zu parsenden Job-Control-Datei
`-h`, `--help`	Ausgabe aller Optionen
`-v`, `--version`	Ausgabe der Version des jtools

Tabelle 2.1: Parameter für das `jtool`

Eine dritte Möglichkeit ist es, das newtjt zu benutzen. Dieses bietet eine einfache NEWT[1]-Oberfläche, die in Abbildung 2.13 zu sehen ist. In dieser Oberfläche können die gewünschten Einstellungen für die zu brennende CD einfach ausgewählt werden. Danach wird automatisch das jtool mit den korrekten Kommandozeilen-Parametern aufgerufen und die Job-Control-Datei erstellt. Somit ist es in der Nutzung einfacher als ein manueller Aufruf des jtools.

Nachdem die Job-Control-Datei erstellt wurde und damit für den späteren Brennvorgang feststeht, wie die hochgeladenen Daten gebrannt werden sollen (Audio-CD, Daten-CD etc.), kann der Benutzer die Job-Control-Datei hochladen. Im Gegensatz zu den Daten wird diese Datei direkt in das Home gelegt. Nach dem Upload wird die Datei geparsed und im Erfolgsfall der Brennauftrag gequeued.

Ist das Parsing fehlgeschlagen, da die Job-Control-Datei offensichtliche logische oder syntaktische Fehler enthielt, so wird der Upload der Datei gar nicht erst positiv bestätigt, sondern vom FTP-Server eine Fehlermeldung zurückgeschickt. Diese sollte normalerweise von FTP-Clients angezeigt werden.

Bei vielen Clients wird allerdings nicht die tatsächlich vom Server übermittelte Fehlermeldung angezeigt, sondern eine Standard-Meldung des Clients. Damit der Benutzer trotzdem den Grund für den Parsingfehler erfährt, wird beim nächsten Wechsel in das Home-Verzeichnis eine Nachricht vom FTP-Server zum Client geschickt. Diese Nachrichten haben sich als beste Möglichkeit erwiesen, um Meldungen durch den Client an den Benutzer zu leiten, da sie in al-

[1] `http://www.oksid.ch/gnewt`

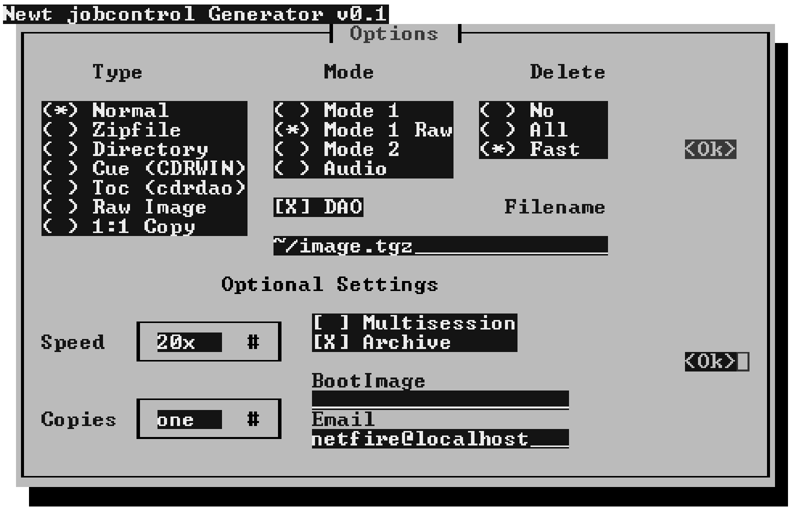

Abbildung 2.13: Screenshot des NEWT-jobtools

len RFC-konformen [20] FTP-Clients angezeigt werden. Es reicht hierbei auch aus, statt eines neuen Logins oder eines Wechsels in ein anderes Verzeichnis, gefolgt von einem Wechsel zurück in das Homeverzeichnis, einfach ein `CWD` auszuführen. Ein Beispiel dafür ist in Abbildung 2.14 zu sehen.

```
ncftp /NF1qyCQY > put jobcontrol
jobcontrol:                 137.00 B 45.57 B/s
put jobcontrol: could not send file to remote host.

ncftp /NF1qyCQY > quote cwd .
Your uploaded jobcontrol was invalid.

You are NOT queued!

Reason was:
"Audrio" is no valid Type.

CWD command successful.
ncftp /NF1qyCQY >
```

Abbildung 2.14: Fehlermeldung beim Upload einer ungültigen Job-Control-Datei

Wurde die Job-Control-Datei ohne Fehlermeldungen angenommen, so ist der Job damit in der Queue. Ansonsten muß man seine Job-Control-Datei berichtigen und erneut hochladen.

Sollen die Daten nach dem Brennen ins Archiv geschoben werden, von wo aus sie zu einem späteren Zeitpunkt noch einmal gebrannt werden können, so ist es ratsam, eine Datei mit dem Namen `description.txt` hochzuladen. Diese wird nicht in das Data-Verzeichnis sondern direkt in das Home gelegt und soll einen beschreibenden Text enthalten, anhand dessen später die Images im Archiv zugeordnet werden können. Der Upload dieser Datei kann auch nach der Job-Control-Datei erfolgen, jedoch nicht, nachdem bereits mit der Abarbeitung des Auftrags durch den `netfired` begonnen wurde.

Über den aktuellen Status seines Brennauftrags kann man sich informieren, indem man sich den Inhalt der Datei 'status' anzeigen läßt. Diese liegt ebenfalls im Home-Verzeichnis. Die Tatsache, daß diese Datei scheinbar keinen Inhalt hat, da sie im Verzeichnis-Listing mit Null Bytes Größe angezeigt wird, hat dabei jedoch nichts zu bedeuten. Dies liegt daran, daß die Datei erst beim Herunterladen durch den Benutzer mit der vom System zurückgelieferten Statusanzeige gefüllt wird. Damit ist sichergestellt, daß der Inhalt der Datei auch immer dem aktuellen Status entspricht.

Die Status-Datei kann nicht gelöscht werden, weder zufällig noch vorsätzlich.

Sollte der Benutzer seinen Brennauftrag vorzeitig aus der Queue entfernen wollen, um z.B. noch etwas an den übertragenen Daten zu ändern oder neue hinzuzufügen, so ist dies ebenfalls möglich. Dazu reicht es, die Job-Control-Datei zu löschen. Der FTP-Server wird den Auftrag daraufhin automatisch aus der Queue des Systems entfernen.

Damit solche Vorgänge nur dann ausgeführt werden können, wenn der Auftrag noch nicht vorbereitet oder gebrannt wird, wird der Account für diese Zeit vom System gesperrt. Sollte es also nicht mehr möglich sein, sich einzuloggen, so wird der Auftrag gerade vom System bearbeitet.

Nachdem der Brennvorgang abgeschlossen ist, werden die Daten, je nach Eintrag in der Job-Control-Datei, entweder ins Archiv verschoben oder gelöscht. Der temporäre Account wird in jedem Fall entfernt, beim nächsten Brennauftrag muß man also wieder einen neuen Account benutzen.

2.2 Brennformate

Das NetFire-System ist in der Lage, CDs in verschiedenen Formaten zu erzeugen. Nachfolgend sind diese Formate aufgelistet.

2.2.1 ISO Images

Diese Option ermöglicht das Brennen von bereits auf der Client-Seite erstellten ISO-Images.

2.2.2 Audio

Diese Option erlaubt das Erzeugen von Audio-CDs, indem man einzelne Tracks (als WAV oder MP3 Dateien) überträgt und danach ihre Reihenfolge festlegt. Die MP3-Files werden dabei von dem Netfire-System in entsprechende WAV-Files umgewandelt. Sollten mehrere Tracks aus derselben Datei erzeugt werden, muß die Datei trotzdem nur einmal übertragen werden.

2.2.3 Ganze Verzeichnisstrukturen

2.2.3.1 Durch Upload der kompletten Struktur

Man kann eine komplette Verzeichnisstruktur auf die CD bringen, indem man diese in Form mehrerer einzelner Dateien überträgt (entweder als einzelne Dateien oder als komplettes Verzeichnis).

2.2.3.2 Durch Upload einer gepackten Datei

Alternativ kann man im voraus auf der Client-Seite ein ZIP, TGZ, ARJ oder RAR Archiv erstellen und dieses übertragen. Vor dem Erzeugen der CD wird es auf der Serverseite durch das NetFire-System automatisch entpackt.

2.2.4 CD Direktkopie

Es können 1:1 Kopien von CDs erstellt werden. Die Original-CD wird dabei im NetFire-System eingelesen.

2.2.5 Kopie einer einer bereits bearbeiteten CD erstellen

Man kann die Images von erzeugten CDs in das Archiv stellen, so daß später bei Bedarf weitere Kopien angefertigt werden können. Das Archiv wird vom NetFire-System automatisch verwaltet.

2.3 Optionen eines Brennauftrages

In der Job-Control-Datei eines Brennauftrages (siehe Abbildung 4.16) können – entweder manuell oder durch die WebGUI – verschiedene Optionen gesetzt werden, die die Abarbeitung des Jobs beeinflussen.

2.3.1 Anzahl der Exemplare

Von einem Job können mehrere Kopien angefertigt werden. So müssen die einzelnen Tracks nur einmal über das Netzwerk übertragen werden, und auch die Vorbereitungszeit eines Jobs wird minimiert. Beim Kopieren einer CD muß die Ursprungs-CD ebenfalls nur ein einziges Mal ausgelesen werden.

Gegenwärtig können maximal 3 Kopien erzeugt werden. Auf diese Weise wird verhindert, daß ein Job das Gerät zu lange belegen kann.

2.3.2 Multisession

„Multisession" ist eine Option, die in vielen Brennprogrammen zu finden ist. Dort wird dieses Feature sehr oft undurchsichtig und unnötig komplex implementiert, so daß hier im Detail darauf eingegangen werden soll.

Eine Anzahl von Tracks auf einer CD wird immer zu einer Sitzung („Session") zusammengefaßt. Eine Session wird in der Regel immer am Stück ohne Unterbrechung aufgezeichnet. Neben den eigentlichen Tracks enthält eine Sitzung einen Leadin- und Leadout-Bereich. Der Leadin enthält das Inhaltsverzeichnis der Sitzung, also die Startpositionen und Modi der einzelnen Tracks. Das Leadout ist datenleer und ist eine Art „Bremsspur" für CD-Player. Wird bei der Musikwiedergabe vom Player ein Sektor des Leadouts gelesen, so wird dieser Sektor anhand seines speziellen Typs erkannt. Der CD-Player stoppt darauf sofort die Wiedergabe.

Wenn sich auf einem Rohling eine „normale" Sitzung befindet und das Leadout bereits geschrieben wurde, ist die CD „abgeschlossen". Es kann also nichts mehr gebrannt werden. Das ist sehr unbefriedigend, vor allem, wenn immer nur kleine Datenmengen aufgezeichnet werden sollen. Vor genau diesem Problem stand die Firma Kodak bei der Einführung der Photo-CD: hier sollte eine CD auch nach dem Aufzeichnen einer Sitzung erweiterbar bleiben. Die Lösung ist die „Multisession-CD": wird eine Sitzung im Multisession-Modus gebrannt, so schreibt der Brenner beim Schließen der Sitzung (nicht der CD !) im Leadin der Sitzung einen Zeiger auf das Ende des Leadouts. Dadurch kann ein CD-Brenner das Ende der CD finden und weitere Sitzungen aufzeichnen. Ein CDROM-Laufwerk kann anhand der Zeiger im Leadin die einzelnen Sitzungen auf der CD lokalisieren und aus den einzelnen Leadins das Inhaltsverzeichnis der CD zusammenstellen.

Natürlich sind auch hier einzelne Punkte zu beachten:

- Nicht jedes CDROM-Laufwerk und vor allem nicht jeder CD-Player kann mit Multisession-CDs umgehen. Vor allem in CD-Playern ist dann immer nur die erste Sitzung sichtbar, so daß diese Option nicht für Audio-CDs verwendet werden sollte. Auch in CDROMs können Multisession-Audio-CDs nicht am Stück abgespielt werden, weil die Wiedergabe bei jedem Leadout abgebrochen wird.

- Wird eine Sitzung nicht im Multisession-Modus gebrannt, wird beim Schließen der Sitzung auch gleichzeitig die CD abgeschloßen.
- Die letzte Sitzung einer Multisession-CD kann im Multisession-Modus gebrannt werden. Der Leadin-Zeiger zeigt dann zwar auf einen leeren Sektor am Ende der CD, was aber unbedenklich ist.

2.3.3 Modi der CD

Diese Option ist zwar vorhanden, gegenwärtig aber noch ohne Funktion. CDs werden immer im ,,Track-At-Once"-Modus gebrannt. Einzige Ausnahme ist das direkte Kopieren einer CD, was im ,,Disc-At-Once"-Modus geschieht.

2.3.4 CD in das Archiv aufnehmen

Diese Option steht nur bei Daten-CDs zur Verfügung. Das erzeugte ISO-Image kann im Archiv für zukünftige Brennvorgänge aufbewahrt werden.

2.3.4.1 Wahl der Brenngeschwindigkeit

Die Geschwindigkeit, mit der die CD aufgezeichnet werden soll, kann ebenfalls eingestellt werden. Wahlmöglichkeiten sind alle sinnvollen Einstellungen von 1x bis 32x. Für noch schnellere Brenner wurde die Option ,,maximal" implementiert. Dahinter verbirgt sich intern 100-fache Brenngeschwindigkeit. Wird eine höhere Geschwindigkeit angegeben als der Brenner unterstützt, so wird automatisch die Maximalgeschwindigkeit des Brenners benutzt.

2.3.5 Vorhergehendes Löschen einer CD/RW

Eine CD/RW kann vor dem eigentlichen Brennvorgang gelöscht werden. Zur Wahl stehen die Optionen ,,schnelles Löschen" und ,,vollständiges Löschen". Beim schnellen Löschen wird nur das erste Leadin gelöscht, beim vollständigen Löschen wird jeder Sektor des Rohlings überschrieben. Das Löschen erfolgt immer mit maximaler Geschwindigkeit.

Kapitel 3

Standardsoftware

Um die Entwicklungszeit des Projekts möglichst gering zu halten, wurde so weit wie möglich auf frei verfügbare Software gesetzt.

3.1 Untersuchung der verfügbaren freien Standardsoftware

Das Netzfiresystem benötigt zum Betrieb folgende Komponenten:

- Betriebssystem
- HTTP-Server
- HTTP-Browser
- Brennsoftware
- FTP-Server

Da diese Komponenten in der vorhandenen Zeit nicht von Grund auf neu entwickelt werden konnten, hat sich die PG dafür entschieden vorhandene freie Software zu verwenden und nur die notwendigen Anpassungen vorzunehmen, wobei jeweils die Lizenz beachte wurde.

3.1.1 Betriebssysteme

3.1.1.1 Einleitung

Im Embedded System Bereich gibt es eine Reihe von verfügbaren Betriebssystemen: QNX, LynxOS, Windows CE, Embedded-Linux und DOS. Für unsere Zwecke kommen QNX, LynxOS und Windows CE nicht in Frage, da hier zum Teil beträchtliche Lizenzkosten für die Software fällig werden. Zu untersuchen bleiben also noch DOS und Embedded-Linux:

3.1.1.2 DOS

DOS gibt es in verschiedenen Varianten, neben der offiziellen Version von Microsoft auch diverse Ableger, die auf Embedded Systems spezialisiert sind. Zu nennen wäre hier etwa DR-DOS (jetzt Caldera) und @CHIP-RTOS[1], ein DOS kompatibles System für den IPC@CHIP. Dieses zeichnet sich durch stark verbesserte Funktionalität wie etwa einen TCP/IP Stack und einen integrierten Webserver aus. Da es sich bei DOS um ein 8-bit Betriebssystem handelt, stellt es keine besonders hohen Anforderungen an die Hardware. Zu den Nachteilen von DOS: Es ist

[1] http://www.goblack.de/desy/sc12chip/apis/index.html

kein besonders robustes System, da es etwa keinen Speicherschutz unterstützt. Die Verfügbarkeit von Treibern insbesondere für Ethernet und IDE-Interface stellt ein Problem dar. Eine Funktionsgarantie gibt es nur, wenn Hardware und die speziell angepasste DOS-Version aus einer Hand gekauft werden. Die verfügbare Software ist eingeschränkt, da nur noch wenige Programme für DOS entwickelt werden, insbesondere auf dem Gebiet von Netzwerksoftware. Die Programmierumgebung ist wenig effizient. Programme können nicht einfach für DOS crosscompiliert werden und müssen auf DOS Rechnern getestet werden. Dies macht die Installation von DOS auf den Entwicklungsrechnern nötig. Die verfügbaren Entwicklungsumgebungen sind proprietär und erfordern kommerzielle Lizenzen. Die DOS-API unterscheidet sich von der Unix libc-API und von der Win32-API. Mit einer hohen Einarbeitung muß gerechnet werden, da einige API-Funktionen nur über die DOS Interrupt Schnittstelle verfügbar sind. Auch muß damit gerechnet werden, daß Teile der Software in 80x86 Assembler kodiert werden müssen. Weiterhin schränkt DOS die Zahl der verfügbaren CPUs auf die x86 Familie ein, da es nicht portierbar ist.

3.1.1.3 Embedded-Linux

Embedded Linux ist ein Sammelbegrif für eine Reihe von Varianten des Linux Betriebssystems. Neben dem „normalen“ Linux-Kernel, der mit angepaßter Systemsoftware als Embedded System eingesetzt werden kann, gibt es noch μClinux, das auf CPUs ohne MMU[2] läuft, und RT-Linux, das harte Echtzeitanforderungen erfüllt. RT-Linux soll hier nicht weiter betrachtet werden, da für das Brennen von CDs ein Task, der mit RT Priorität im normalen Linux Scheduler läuft, ausreichend ist. μClinux und Linux können gemeinsam betrachtet werden, da hier im wesentlichen dieselbe Codebasis genutzt wird und sich die API nicht unterscheidet. Die Nachteile von Linux sind die höheren Anforderungen an die Hardware. Eine 32bit CPU ist notwendig und eine minimale Menge von 2MB RAM. Um den Kernel und die Applikationen zu speichern, werden etwa 2MB an Flash Speicher benötigt. Zu den Vorteilen von Linux: Es ist ein modernes 32bit Betriebssystem, für das eine große Zahl an Treibern und Software frei verfügbar ist. Sowohl IDE als auch Ethernet werden gut unterstützt, TCP/IP ist Bestandteil des Kernels. Es ist sehr robust, und es gibt eine Menge frei verfügbarer Software, die FTP/HTTP-Server implementiert. An Brennsoftware ist *cdrecord* verfügbar, und zum Erzeugen der Images *mkisofs*. Beide Programme sind unter der GPL verfügbar[3]. Die Linux-API ist POSIX kompatibel. Cross-compilation ist sowohl von Linux als auch von Windows (mittels Cygwin[4]) möglich. Dank der einheitlichen Linux API kann auf einem normalen PC entwickelt und getestet werden, bevor die Software auf das Embedded System übertragen wird. Lizenzkosten werden weder für die Entwicklungsumgebung noch für das Laufzeitsystem fällig. Ein weiterer Vorteil ist die Verfügbarkeit der Quellcodes unter der GPL[5].

3.1.1.4 Fazit

Anhand von Tabelle 3.1 werden die wichtigsten Unterschiede noch einmal gegenübergestellt. Zu beachten ist, daß zusätzlicher RAM als Pufferspeicher für den CD-Brenner benötigt wird. Es werden mindestens 8MB zusätzlich benötigt.

Als Betriebssystem scheint der Einsatz von Linux als Systemsoftware geboten, da für Linux bereits eine Menge der benötigten Software im Quellcode vorhanden ist. Dies ermöglicht im Vergleich zu DOS eine höhere Flexibilität und eine kürzere Entwicklungszeit.

[2] Memory Management Unit

[3] `http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html`

[4] `http://sources.redhat.com/cygwin/`

[5] Gnu General Public Licence, siehe `http://www.gnu.org/copyleft/gpl.html`

	DOS	Embedded-Linux
Verfügbare Treiber	gering	hoch
Verfügbare Software	gering	hoch
Speicherschutz	nein	ja
Multitasking	nein	ja
API	proprietär	POSIX
Entwicklungsumgebung	proprietär	GNU
Cross-compilation	nein	ja
CPU	8086 oder kompatibler	diverse 32bit CPUs
Speicher (RAM/ROM)	512kb/512kb	2Mb/2Mb
Lizenzkosten	Entwicklungsumgebung/ Laufzeitsystem	keine
Sourcecode verfügbar	nein	ja

Tabelle 3.1: Übersicht DOS/Embedded Linux

3.1.2 HTTP-Server

Einleitung

Es standen zwei Alternativen zur Auswahl: entweder ein minimal ausgebauter Apache[6] oder irgendein spezieller Webserver, wie z.B. thttpd[7], boa[8] usw.

Boa

Da von den betrachteten Webservern (außer Apache, der zunächst als überdimensioniert angesehen wurde) nur boa die Möglichkeit bot, CGI-Skripte auszuführen, wurde die Entscheidung zunächst zugunsten boa gefällt, da er sehr klein ist (unter 50 Kb) und damit auch der Installation auf einem sehr kleinen Flash-Modul (16 Mb) nicht im Wege stehen würde.

Apache

Als jedoch sich herausgestellt hat, daß boa CGI-Skripte nicht anhand des Dateityps erkennen (und dann mit dem entsprechenden Interpreter aufrufen) kann, sondern die Angabe des Interpreters explizit in dem Skript erfordert, wurden Anpassungen notwendig. Der Quellcode von boa hat sich allerdings als nicht wart- oder anpassungsfähig erwiesen. Da der Aufwand für die Reinigung des Quellcodes als zu hoch angesehen wurde, wurde die frühere Entscheidung revidiert, und die neue Wahl fiel auf Apache.

Apache-Konfiguration

Apache wird dabei mit einer minimalen Konfiguration betrieben, d.h. es werden nur die notwendigen Module geladen und die Konfigurationsdatei selbst wurde soweit es sinnvoll möglich war, auf ein Minimum reduziert [9].

[6] `http://www.apache.org`
[7] `http://www.acme.com/software/thttpd/`
[8] `http://www.boa.org`
[9] Die Module sind: config_log_module, mime_module, negotiation_module, status_module, autoindex_module, dir_module, cgi_module, userdir_module, alias_module, rewrite_module, access_module, auth_module, expires_module, unique_id_module, setenvif_module, php4_module

3.1.3 HTTP-Browser

Die PG NetFire hat sich zum Ziel gesetzt, ein Client-Server-System zu realisieren, bei dem die Client-Seite vollkommen unabhängig von der Benutzerplattform gestaltet wird. Diese Plattformunabhängigkeit wird dadurch hergestellt, daß die Client-Seite in Form von HTML-Seiten dargestellt wird, durch die der Benutzer alle nötigen Einstellungen zum Brennen vornehmen und den Brennvorgang starten kann. Weiterhin ist auch die Oberfläche zur Administration des Systems durch HTML-Seiten realisiert worden. Alle Funktionen des entwickelten Systems stehen also auf jedem Betriebssystem zur Verfügung, auf dem ein Webbrowser installiert ist.

Der Upload der ISO-Images auf den Server wird in Form eines FTP-Uploads vorgenommen, der im Gegensatz zum HTTP-Upload folgende Vorteile bietet:

- Der Upload ist resume-fähig, es kann also auch ein abgebrochener Upload ohne Datenverlust fortgeführt werden

- Kaum Minimalanforderungen an den Client, die bei einem HTTP-Upload vorausgesetzt werden müssen (temporärer Speicherplatz auf dem Client-Rechner)

- Geringere Anforderungen an den Server, der weniger Cache/Swap benötigt

- Weniger Overhead beim Übertragen der Daten

Aufgrund der gestellten Anforderung muß jeder eingesetzte Webbrowser auf der Client-Seite die Möglichkeit bieten, einen FTP-Upload durchzuführen. Idealerweise findet dieser FTP-Upload per *Drag&Drop* statt, so daß dieser in die Benutzerführung der HTML-Seiten eingebunden werden kann. Alternativ kann der Benutzer auf sehr minimalen Plattformen auch einen FTP -Upload mit einem FTP -Client durchführen.

Um festzustellen, ob für jedes Betriebssystem ein Browser zur Verfügung steht, der mindestens den FTP -Upload beherrscht, wurde eine Recherche für verschiedene Betriebssysteme vorgenommen, die die *Drag&Drop*-Fähigkeit sowie die FTP-Upload-Fähigkeit der einzelnen Browser untersucht. Tabelle 3.2 zeigt die Ergebnisse der Recherche.

Die Spalte *Upload-Menü* zeigt an, ob ein Browser die Möglichkeit eines FTP-Uploads im Menü unter dem Menüpunkt `Datei`, bzw. `File` bietet.

Die Recherche zeigt, daß es für jedes Betriebssystem mindestens einen Browser gibt, der den FTP-Upload beherrscht. Es lassen sich teilweise ganze Verzeichnisse rekursiv auf den FTP-Server uploaden, was in der Tabelle durch den Eintrag *mehrere Dateien* gekennzeichnet ist. Dies wird dann sinnvoll, wenn die PG ihr Minimalziel überschreitet, das vorerst darin besteht, einzelne ISO-Images vom Client zum Brennsystem zu übertragen und auf eine CD-R zu brennen. Es bleibt also die Option, auch mehrere einzelne Dateien, oder eine ganze Verzeichnisstruktur auf das System mit dem CD-Brenner zu transferieren und erst dort daraus ein ISO-Image zu erstellen. Dies erleichtert dem Benutzer das Brennen einer CD.

Besonders gute Ergebnisse bezüglich des FTP-Uploads mit einem Webbrowser lassen sich erzielen, wenn man den File-Manager benutzt, der zum Betriebssystem, bzw. zur benutzten grafischen Oberfläche gehört, und diesen auch als Webbrowser benutzt. Alternativ bietet sich die Benutzung des Netscape Navigator an, der auf allen Plattformen verbreitet ist, und der auf Windows und MacOS-Plattformen *Drag&Drop* unterstützt. Tabelle 3.3 zeigt Konfigurationen aus Betriebssystem, Browser und File-Manager, die *Drag&Drop* unterstützen und sich so besonders für eine komfortable Benutzung des NetFire-Systems anbieten.

Alle HTML-Seiten werden ohne die Verwendung von Java und Javascript erstellt, so daß die Benutzung mit allen aktuellen Browsern keine Probleme bereitet. In dieser Hinsicht werden also keine weiteren Anforderungen an die Webbrowser auf dem System des Benutzers gestellt.

OS	Browser	File-Manager	Drag& Drop	Upload-Menü	Verzeichnis
Windows	Netscape 4.x	Windows-Explorer	ja	ja	nein
Windows	Internet Explorer 5.x	Windows-Explorer	ja	nein	ja
Windows	Internet Explorer 6.x	Windows-Explorer	ja	nein	ja
Windows	Opera 5.x	Windows-Explorer	nein	nein	nein
Windows	Opera 6.x	Windows-Explorer	nein	nein	nein
Windows	Mozilla 5.0 / 0.9x	Windows-Explorer	nein	nein	nein
Linux	KFM	KFM	ja	nein	mehrere Dateien
Linux	Konqueror	Konqueror	ja	nein	ja
Linux	Nautilus	Nautilus	ja	nein	ja
Linux	Netscape 4.x	beliebig	nein	ja	nein
Linux	Opera 5.x	beliebig	nein	nein	nein
Linux	Opera 6.x	beliebig	nein	nein	nein
Linux	Mozilla 5.0 / 0.9x	beliebig	nein	nein	nein
Linux	Galeon	beliebig	nein	nein	nein
Solaris	Netscape 4.x	beliebig	nein	ja	nein
Solaris	Netscape 3.x	beliebig	nein	ja	nein
Solaris	Opera 5.x	beliebig	nein	nein	nein
Solaris	KFM	KFM	ja	nein	mehrere Dateien
MacOS 9.0	Opera 5.x	beliebig	nein	nein	nein
MacOS 9.0	Internet Explorer	beliebig	nein	nein	nein
MacOS 9.0	Netscape 4.x	beliebig	ja	nein	mehrere Dateien

Tabelle 3.2: Übersicht der FTP -Fähigkeit von Webbrowsern

OS	Browser	File-Manager
Windows	Netscape 4.x	Windows-Explorer
Windows	Internet Explorer 5.x	Windows-Explorer
Windows	Internet Explorer 6.x	Windows-Explorer
Linux	KFM	KFM
Linux	Konqueror	Konqueror
Linux	Nautilus	Nautilus
Solaris	KFM	KFM
MacOS 9.0	Netscape 4.x	beliebig

Tabelle 3.3: Ideale Systemkonfigurationen für den FTP-Upload per Drag&Drop

3.1.4 CD-ROM Brennersoftware

Die Auswahl an geeigneter und verfügbarer Brennsoftware für Linux ist begrenzt, so daß die Entscheidung für ein Softwarepaket schnell getroffen werden konnte. Die Wahl fiel auf die cdrtools von Jörg Schilling[10]. Mit den cdrtools wurde dann ein Funktionstest des Ricoh-Brenners vorgenommen. Das diente allerdings nicht nur dem Zweck, den Brenner auf Funktionstüchtigkeit zu prüfen, sondern auch, um die Kompatibilität mit dem Brenner und die Möglichkeiten des Softwarepakets auszuloten. Es hat sich gezeigt, dass der Brenner nicht nur einwandfrei funktioniert, sondern ebenfalls komptibel zu den cdrtools ist. Vor allem ist diese Software in der Lage, alle Informationen auszugeben, die die von uns zu entwickelnde Software für den Betrieb benötigt, wie z.B. Informationen über den eingelegten Rohling. Mit den cdrtools lassen sich Images (ISO9660) und Audio Tracks brennen, Images aus Verzeichnissen erstellen, und Audio Tracks auslesen.

Für Direktkopien sorgt CDRDAO[11].

3.1.5 FTP-Server

Bei der Untersuchung der verfügbaren FTP-Server galt es, ein Produkt zu finden, das einerseits einen möglichst großen Funktionsumfang mit sich bringt und andererseits durch Stabilität und Sicherheit überzeugt. Außerdem sollte es uns in der konzeptionellen Planung nicht behindern, sondern es sollte in jedem Fall sichergestellt sein, daß der FTP-Server an alle Erfordernisse der PG angepaßt werden kann und nicht umgekehrt. Diese Überlegung bedeutete, daß der Server unbedingt als Open Source vorliegen muß, da sonst eigene Erweiterungen nicht möglich sind. Bezüglich der gestellten Anforderungen wurden daraufhin einige bekannte FTP-Server für die Unix-Plattform überprüft:

Glftpd[12] Dieser FTP-Server ist vor allem für seinen enormen Funktionsumfang bekannt. Dies scheint zunächst vorteilhaft. Da der Server allerdings noch nicht als Open-Source verfügbar ist, bedeutet dies, daß viele Features, wie z.B. die mächtige Userverwaltung, die in unserem Fall gar nicht benötigt wird, auch nicht entfernt werden können. Da diese Features fest einkompiliert sind, ergibt sich damit ein Widerspruch zu der Idee, ein möglichst kompaktes System aufzubauen. Eigene Erweiterungen wären zwar auch möglich, müßten aber auf eine andere Art realisiert werden. So ist es z.B. möglich, für jeden FTP-Befehl ein oder mehrere selbsterstellte Skripte vor oder nach der Ausführung durch den Server abzuarbeiten. Damit könnte man z.B. eine gepackte Datei direkt nach dem Upload auf CRC-Fehler überprüfen und dem Client das Ergebnis der Prüfung direkt mitteilen. Allerdings sind Erweiterungen eben nur über diese externen Skripte möglich. Die Erweiterbarkeit und Anpassungsfähigkeit auf unsere Bedürfnisse ist daher nur begrenzt gegeben, weshalb die Entscheidung auch nicht auf den Glftpd gefallen ist.

Trollftpd[13] Dieser FTP-Server hat vor allem einen Vorteil: er ist extrem klein. Mit nur ca. 40KB hätte sich der Trollftpd für ein Embedded System natürlich sehr gut geeignet. Es kann allerdings davon ausgegangen werden, daß der Funktionsumfang auch entsprechend eingeschränkt ist, was für unsere PG einen entsprechend hohen Weiterentwicklungsaufwand bedeutet hätte. Außerdem wäre bei diesem Konzept wohl die Implementierung eigener Funktionen nur direkt im Source möglich gewesen, was bei neueren Versionen einen nicht unerheblichen Anpassungsbedarf ergeben hätte. Die Alternative, eventuell auftretende Sicherheitslöcher nicht durch Updates zu beheben, scheint da wenig pragmatisch. Ein umfassender Test auf die Tauglichkeit für unser Projekt mußte allerdings ausbleiben,

[10] `http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html`
[11] `http://cdrdao.sourceforge.net/`
[12] `http://www.glftpd.com`
[13] `http://www.trolltech.com/developer/download/ftpd.html`

da der Download-FTP-Server der Trollftpd-Entwickler über längere Zeit nicht erreichbar war.

Wu-ftpd[14] Wu-ftpd, der an der Washington University entwickelt wurde, ist wohl der am weitesten verbreitete FTP-Server im Internet. Das bedeutet jedoch nicht, daß er damit auch die beste Wahl wäre. Da der Wu-ftpd seit vielen Versionen ein ständiger Gast beim CERT[15] ist, war bezüglich der Sicherheit entsprechende Skepsis angebracht. Da eigene Implementierungen hier nur als SITE-Command realisiert werden können, hieße der Einsatz des Wu-ftpd Abschied vom Upload mit graphischen FTP-Clients zu nehmen. Denn diese unterstützen, wenn überhaupt, nur bekannte SITE-Commands und keine, die der Benutzer manuell eingeben muss. Die Alternatvie, die Funktionalität direkt in den Wu-ftpd zu programmieren, scheint allerdings in Anbetracht der zahlreichen Sicherheitspatches, die pro Version erscheinen, einer Sisyphusarbeit gleichzukommen.

NcFTPd[16] Der NcFTPd von Mike Gleason ist vor allem bei Administratoren beliebt, die auf einen effizienten FTP-Server wertlegen. So wurde beim NcFTPd darauf verzichtet, pro Verbindung eine neue Instanz zu erzeugen. Alle Anfragen werden von einer Instanz bearbeitet, was entsprechend Ressourcen schont. Außerdem wurde das Grundgerüst nicht, wie z.B. beim Wu-ftpd, vom originalen BSD Release übernommen, sondern mit Augenmerk auf Geschwindigkeit und Sicherheit neu geschrieben. Leider ist dieser Server aber weder über Skripte noch durch Veränderungen am Source erweiterbar, da es sich nicht um ein Open-Source Projekt handelt. Für den Einsatz mit vielen Usern ist er sogar kostenpflichtig und damit gänzlich ungeeignet für ein Projekt, was den Anspruch erhebt, einmal komplett unter der GPL zu stehen.

ProFTPd[17] Der ProFTPd wurde im Hinblick auf Sicherheit und leichte, aber mächtige Konfigurierbarkeit entwickelt. Die Programmierer wollten damit die Features des Wu-ftpd übertreffen und gleichzeitig einen sicheren Server anbieten. Daß dies offenbar gelungen ist, sieht man an den relativ langlebigen Versionen und an den vielen großen Projekten, die den ProFTPd als Server einsetzen. So wird u.a. ftp.kernel.org oder gnu.org mit dem ProFTPd betrieben. Was die Anpassung an die Erfordernisse anging, war bei diesem Programm vor allem von Vorteil, daß viele Funktionen, die sonst nur vom Glftpd geboten werden, bereits integriert sind. So können z.B. Verzeichnisse, auf die ein User keinen Zugriff hat, von vornherein versteckt werden, oder die Usernamen der anderen User auf dem FTP durch einen Standard-String ersetzt werden. Diese Funktionalität hat uns in der Entwicklung viel zusätzliche Arbeit erspart. Durch die vom Apache[18] übernommene Konfigurationssprache ist es auch möglich, deutlich komplexere Regelsätze als die von uns verwendeten aufzubauen, auch wenn dies zum jetzigen Zeitpunkt noch nicht nötig erscheint.

Besonders überzeugend war jedoch, daß der ProFTPd von Haus aus über eine Moduleinbindung verfügt. Darin lassen sich für jeden Befehl PRE- und POST-Commands erzeugen oder sogar komplette FTP-Befehle umschreiben, ohne den Sourcecode des ProFTPd selbst anzurühren. D.h. für uns, daß wir ohne Probleme neue Versionen der Entwickler mit unserem Modul kompilieren können. Der Zeitaufwand für ein solches Update liegt damit bei der reinen Kompilierungszeit. Ausserdem sind die Module, ebenso wie der Server selbst, in C geschrieben, womit uns eine deutlich leistungsfähigere und vor allem effizientere Sprache zur Verfügung steht, als z.B. beim Glftpd, wo lediglich Skripte ausgeführt

[14] `http://www.wuftpd.org`
[15] `http://www.cert.org`
[16] `http://www.ncftpd.org`
[17] `http://www.proftpd.org`
[18] `http://www.apache.org`

werden können.

Die Wahl, für dieses Projekt den ProFTPd zu nehmen, hat sich im nachhinein auch deshalb als richtig erwiesen, da wir uns dazu entschieden haben, die Kommunikation zwischen dem `netfired` (s. Kapitel 4.3), der für die Verwaltung und Bearbeitung der Brennaufträge zuständig ist, über Unix-Sockets laufen zu lassen. Ohne ein in einer leistungsfähigen Sprache programmiertes Modul wäre dies einerseits mit einem erheblichen Aufwand verbunden gewesen und andererseits vom Endergebins her nicht so elegant gewesen, wie der hier gewählte Ansatz.

Kapitel 4

Softwareentwicklung

Das Konzept des NetFire-Brenners sieht auf der Softwareseite die Verwendung eines hohen Anteils von Standardsoftware vor. Als Betriebssystem wurde die Linux-Distribution *emdebian*[1] gewählt, die an einigen Stellen weiterentwickelt wurde. Für die meisten anfallenden Aufgaben existieren bereits Softwarelösungen, wie z.B. `cdrecord`, um auf CDs zu brennen, `mkisofs`, um aus einzelnen Dateien ein ISO9660-Image zu erzeugen, etc.

Diese Standardsoftware wurde im Rahmen der PG durch selbstentwickelte Softwarekomponenten zum NetFire-System komplettiert. Dieses Kapitel beschreibt die selbstentwickelte Software und die Zusammenhänge zwischen den einzelnen Komponenten.

4.1 Grobstruktur der Softwareentwicklung

Das NetFire-System wird vom Benutzer über ein HTML-Webinterface angesteuert, das sowohl die normale Funktionalität zum Brennen von CDs, so wie auch administrative Aufgaben ermöglicht. Der Upload der Daten zum NetFire-System wird über einen FTP-Client vorgenommen. Weiterhin dient zur Kommunikation mit dem Benutzer das im Rahmen der PG entwickelte LC-Display, das über einen eigenen Treiber im Linux-Kernel angesprochen wird. Dieser Teil des Systems, der zur Benutzerinteraktion dient, wird im folgenden auch als Frontend bezeichnet.

Das Backend des NetFire-Systems besteht hauptsächlich aus dem *emdebian*, das als Betriebssystem verwendet wird, dem `ProFTPd`, der den Upload der Daten vom Benutzer zum NetFire-System übernimmt, so wie dem Apache-Webserver, der um PHP[2]-Funktionalität erweitert wurde, und dem Benutzer das HTML-Webinterface zur Ansteuerung des Systems zur Verfügung stellt. Alle diese Standardkomponenten wurden an vielen Stellen verändert und ergänzt um sie für NetFire-spezifische Aufgaben zu benutzen. Weiterhin gehören zum Backend des Systems noch einige selbstentwickelte Tools:

- `accountmaker`
  Erzeugt neue Nutzerkennungen

- `accountdeleter`
  Entfernt eine Nutzerkennung und die unter ihr gespeicherten Daten

- `support-scripts`
  Dienen zur Vorbereitung und Abarbeitung der Brennaufträge

Für ein Embedded System wie den NetFire-Brenner fehlt somit hauptsächlich ein Programm, das das Zusammenspiel zwischen all den Tools koordiniert. Diese zentrale Aufgabe

[1] `http://www.emdebian.org`
[2] PHP: Hypertext Preprocessor: `http://www.php.net`

übernimmt der von der PG entwickelte `netfired` Daemon. Der `netfired` bietet eine Schnittstelle zu den Benutzeroberflächen (WebGUI, modifizierter FTP-Server), eine Queue, in der die abzuarbeitenden Jobs verwaltet werden, und eine Routine, um die Standardsoftware aufzurufen, mit der letztendlich die gewünschte CD gebrannt wird.

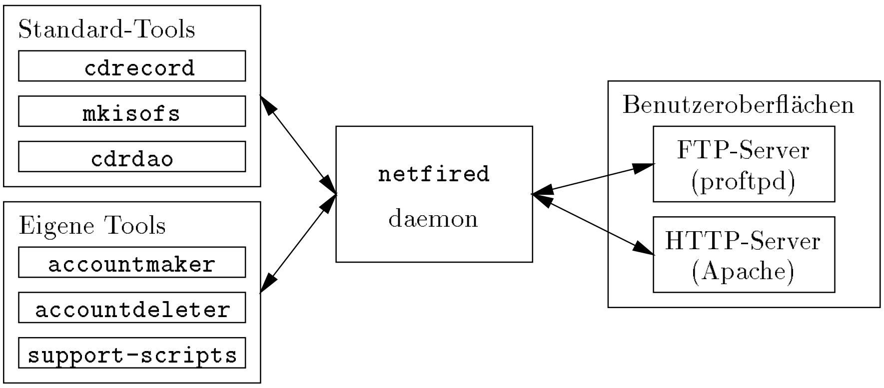

Abbildung 4.1: Grobe Softwarestruktur

Ein grobes Abbild der Softwarestruktur zeigt Abbildung 4.1. Diese Struktur wurde gewählt, um die Benutzeroberflächen unabhängig von den Schnittstellen der Standard-Tools zu halten. Dieser Ansatz gewährleistet ein einfacheres Austauschen der Tools, falls es neuere Versionen gibt, die sich nach außen hin anders als ihre Vorgänger verhalten. So genügt es, die Steuerung im `netfired` Daemon anzupassen, die Frontends bleiben davon unberührt.

4.1.1 Kontrollfluß der Software

Das Gesamtsystem wird als Client-Server-System realisiert. Auf der Client-Seite wird vom Benutzer ein Webbrowser benutzt, der die HTML-Seiten zur Steuerung des NetFire-Brenners darstellt. Auf dem NetFire-Brenner selbst ist ein Apache-Webserver mit CGI-Fähigkeiten installiert. Dieser stellt die Schnittstelle zwischen Client und den Applikationen auf dem NetFire-Brenner dar. Als CGI-Skriptsprache dient hier aufgrund der Mächtigkeit und der Einfachheit PHP.

Auf Serverseite interagiert der Webserver hauptsächlich mit dem `netfired` Daemon, der im Rahmen der Projektgruppe erstellt wurde. Dieser Daemon bedient sich einiger Shell-Skripte und einiger Systembefehle des *Embedded Debian*, das auf dem NetFire-System installiert wird. Teilweise benutzen die PHP-Skripte auf dem Webserver auch eigene selbsterstellte Shell-Skripte oder kommunizieren direkt mit dem FTP-Server. Dieser FTP-Server ist für den Upload der Daten vom Nutzer auf das NetFire-System zuständig.

Die Komplexität der zahlreichen interagierenden Entitäten des Gesamtsystems macht es nötig, die einzelnen Datenflüsse im Detail zu betrachten. Hierzu bietet es sich an, sämtliche Kommunikation zwischen Client und NetFire-System sowie zwischen den Anwendungen auf dem NetFire-System selbst in Diagrammen darzustellen. Im folgenden werden für alle wichtigen Datenflüsse bei der Benutzung des Systems Sequenzdiagramme angegeben.

4.1.1.1 Daten senden

Zunächst ist es interessant, wie der Datenfluß im einzelnen aussieht, wenn der Benutzer, über das oben genannte HTML-Interface, Daten zum CD-Brenner schicken möchte. Abbildung 4.2 zeigt den gesamten Datenfluß, der in diesem Fall auftritt.

Der Benutzer meldet sich zunächst durch den Aufruf der Startseite beim Webserver an. Der Webserver meldet dem `netfired` den Zugriff eines neuen Benutzers auf den Brenner. Da zu

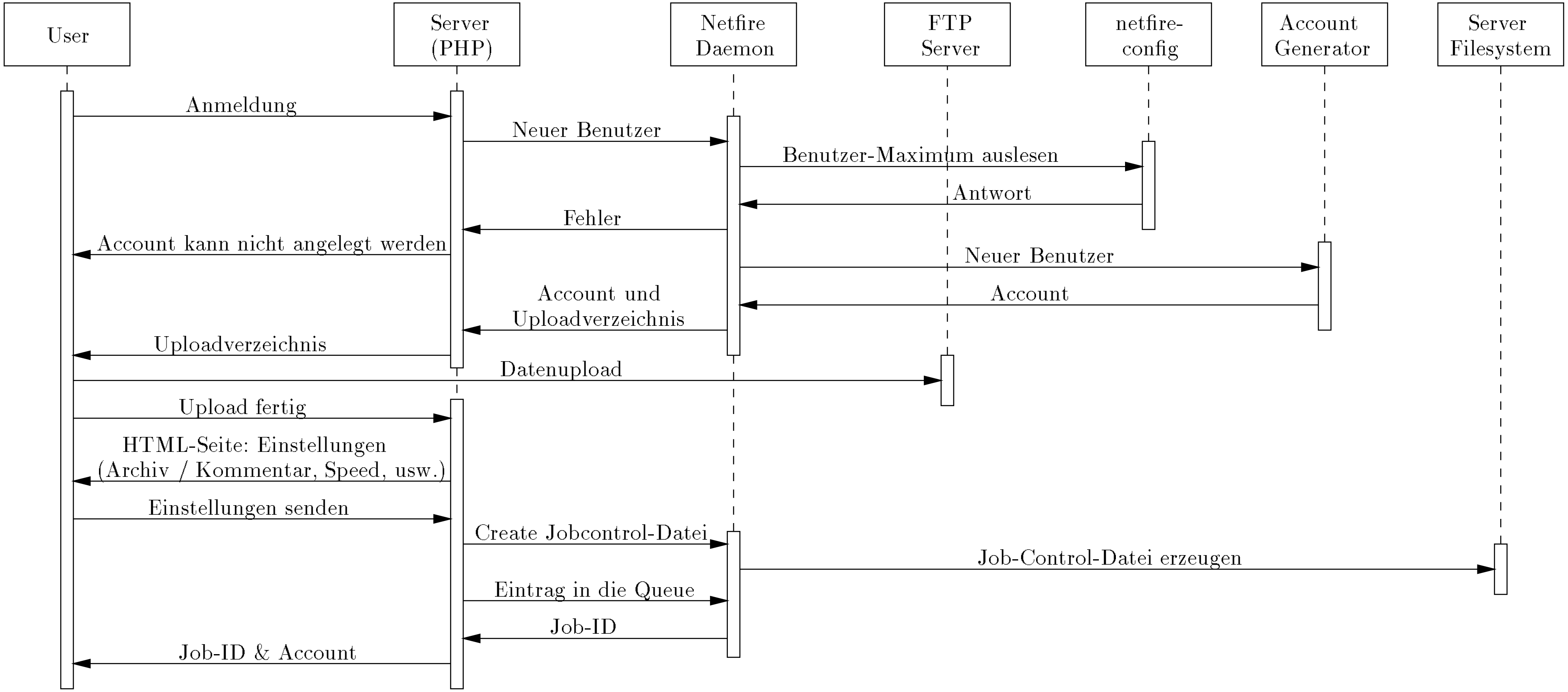

Abbildung 4.2: Der Datenfluß beim Senden von Daten an das NetFire-System

diesem Zeitpunkt unklar ist, ob genügend Speicherplatz frei ist, um den Upload des Benutzers anzunehmen, überprüft der Daemon anhand der Einstellung in eine Konfigurationsdatei, ob die Maximalzahl der verfügbaren Accounts schon erreicht ist. Ist die Maximalzahl erreicht, so erhält der Benutzer über den Webserver die Meldung, daß kein weiterer Account angelegt werden kann. Bei Erfolg bekommt der Benutzer ein Upload-Verzeichnis, das vom Account-Generator erzeugt wurde. Er kann anschließend die zu brennenden Daten zum Server schicken und nach vollendetem Upload auf einer HTML-Seite sämtliche Einstellungen tätigen, die für das Brennen einer CD nötig sind (Geschwindigkeit, Kopien, Optionen für CD-RW, etc.). Nachdem alle Einstellungen des Benutzers an den Webserver übermittelt sind, stellt ein PHP-Skript, das auf dem Webserver läuft, den Inhalt der Job-Control-Datei zusammen. Da der Webserver an der entsprechenden Stelle im Upload-Verzeichnis kein Schreibrecht hat, wird die Job-Control-Datei schließlich durch den Befehl `create` über den `netfired` geschrieben. Diese Datei enthält alle wichtigen Einstellungen zum Brennen. Außerdem wird dem `netfired` mitgeteilt, daß ein neuer Auftrag vorliegt, und dieser wird in die Queue eingetragen. Zum Schluß erhält der Benutzer eine Job-ID, die für weitere Statusabfragen oder auch zum Löschen des Auftrags verwendet wird.

4.1.1.2 ISO aus dem Archiv brennen

Das NetFire-System gibt dem Benutzer auch die Möglichkeit, gesendete Dateien auf der Festplatte in einem Archiv abzulegen. Dort werden auf Wunsch des Benutzers Daten archiviert und können zu einem späteren Zeitpunkt wieder gebrannt werden. Aus diesem Grund ist es nötig, auch den Datenfluß zu bedenken, wie er beim Brennen einer CD, deren Daten aus dem Archiv stammen, auftritt. Abbildung 4.3 zeigt den Datenfluß für diesen Fall.

Dieser Datenfluß ist dem in Kapitel 4.1.1.1 sehr ähnlich. Der Webserver zeigt direkt über ein PHP-Skript sämtliche ISO-Images an, die sich im Archiv befinden. Der Benutzer wählt über das HTML-Interface eines aus und nimmt wie auch im obigen Fall die Einstellungen zum Brennen vor. Es wird eine Job-Control-Datei generiert und der Brennauftrag in die Queue eingetragen. Am Ende erhält der Benutzer eine Job-ID und bekommt hier auch den Account angezeigt, der dem Webserver schon seit dem Anlegen des Benutzers bekannt ist. Dieser Account dient dem Benutzer später als Quasi-Passwort, wenn er z.B. den Job wieder löschen möchte.

4.1.1.3 Brennauftrag vorbereiten

Nachdem der Benutzer die zu brennenden Daten zum Server geschickt hat oder ein Image aus dem Archiv ausgewählt hat, das gebrannt werden soll, werden die Daten weiterverarbeitet. Abbildung 4.4 zeigt den Datenfluß der Abarbeitung im Detail.

Der `netfired` benutzt einen Scheduler, um zu kontrollieren, wann die Daten in der Queue vorbereitet werden, d.h. wann aus ihnen ISO-Images erstellt werden, die später durch einen Aufruf von cdrecord gebrannt werden können. Der `netfired` überprüft so, ob einer der ersten beiden Einträge in der Queue noch nicht vorbereitet ist, und bereitet diesen dann gegebenenfalls vor. Wenn ein Auftrag in der Queue liegt, der vorzubereiten ist, so erstellt der `netfired` ein temporäres Verzeichnis, das zum Vorbereiten des Jobs benötigt wird. Anschließend ruft dieser das Skript `nfprepare.sh` auf. Dieses gibt auf dem LCD über den `lcdfired` die Mitteilung aus, daß die CD vorbereitet wird. Anschließend liest dieses Skript die Job-Control-Datei aus und ermittelt den Typ der CD die gebrannt werden soll (ISO, Audio, gepackte Daten, mehrere Dateien, etc.). Wenn der Typ ermittelt ist, werden ein oder mehrere Hilfsskripte aufgerufen, die die Daten zu einem fertigen ISO aufbereiten, das anschließend gebrannt werden kann. Die fertigen Daten werden dann in das reservierte temporäre Verzeichnis verschoben. Anschließend bekommt der `netfired` über das Skript `nfprepare.sh` eine Rückgabe, daß der Job nun fertig vorbereitet ist. Außerdem wird das erfolgreiche Beenden der Vorbereitung auch mit einer Mitteilung auf dem LCD angezeigt.

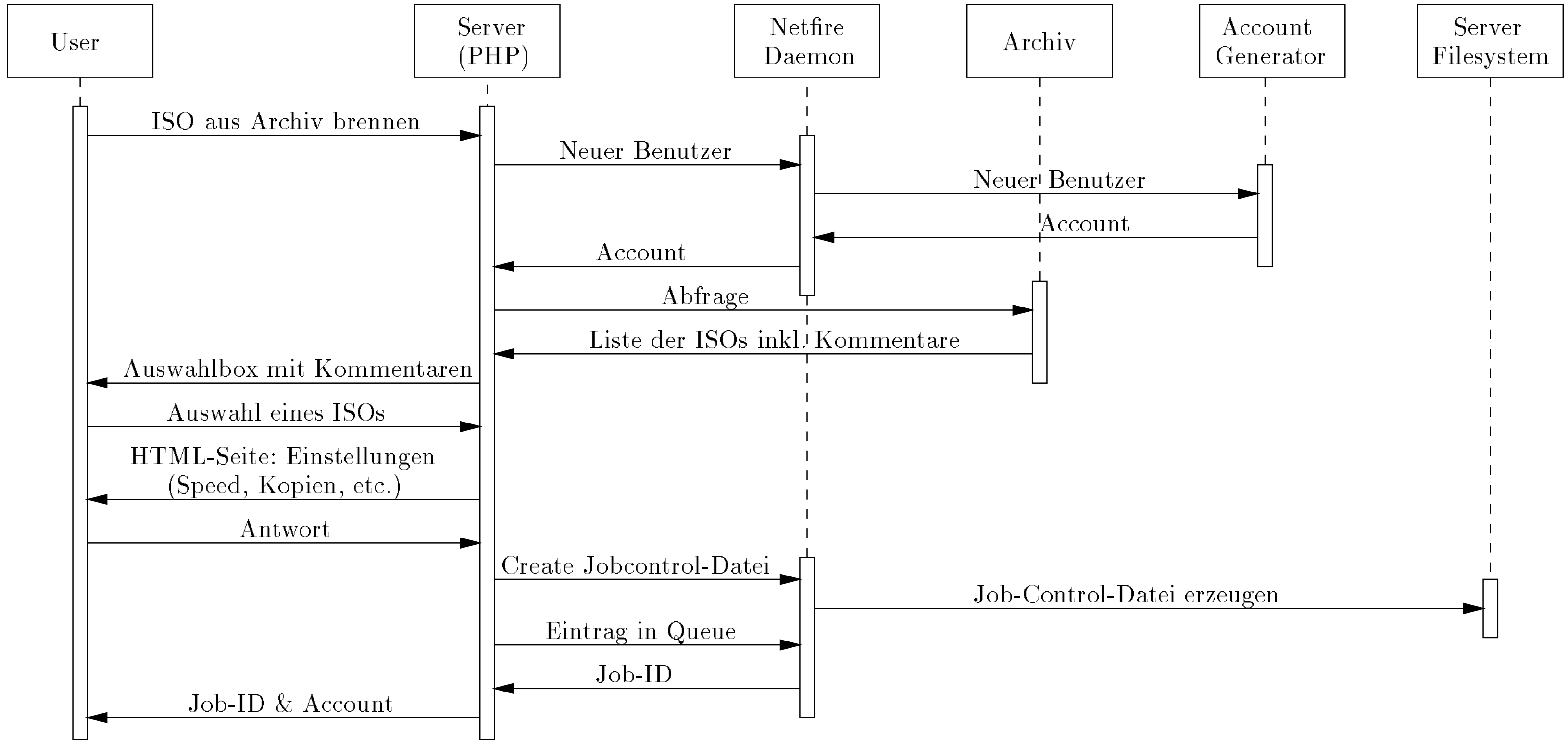

Abbildung 4.3: Ein ISO-Image aus dem Archiv des Systems wird gebrannt

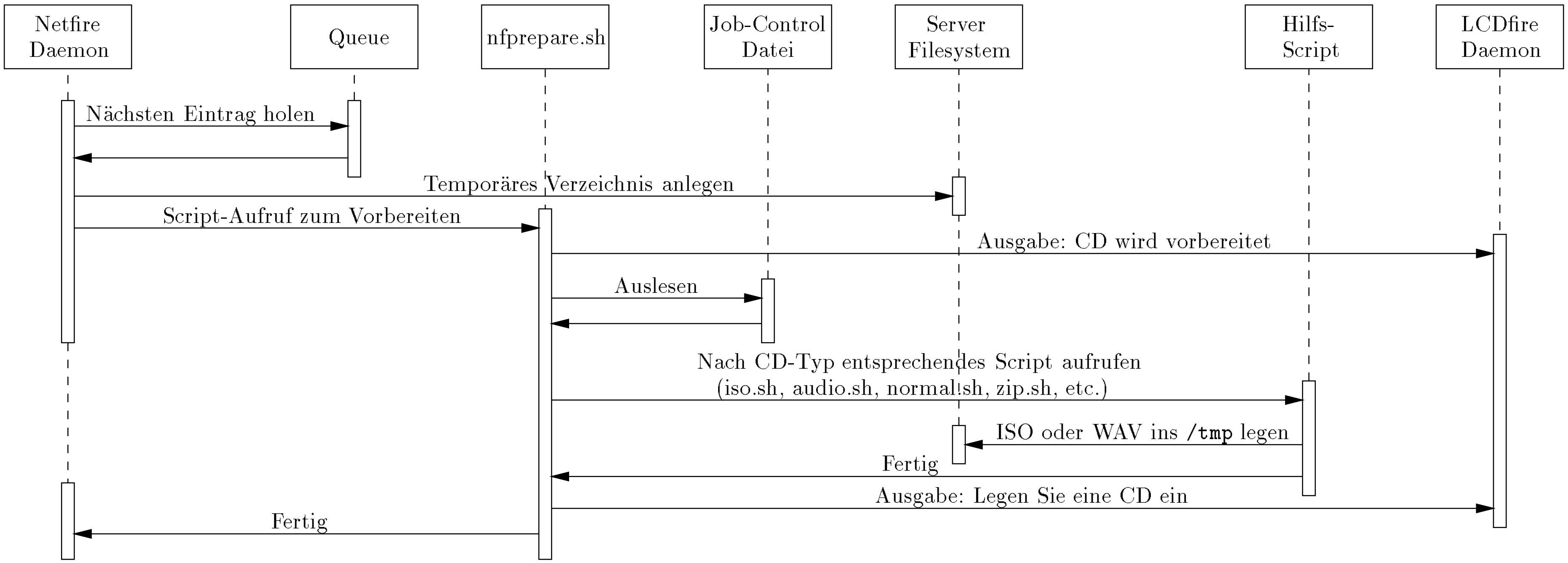

Abbildung 4.4: Ein Auftrag zum Brennen wird vorbereitet

4.1.1.4 Brennen der vorbereiteten Daten

Die vorbereiten Daten auf dem NetFire-System bleiben solange im temporären Verzeichnis liegen, bis der entsprechende Auftrag in der Queue des `netfired` an erster Stelle ist. Wenn der Auftrag an der Reihe ist, sorgt der `netfired` unter Zuhilfenahme einiger Support-Scripts für dessen Abarbeitung. Abbildung 4.5 zeigt den Datenfluß der Abarbeitung im Detail. Wenn der Auftrag an der Reihe ist, ruft der Daemon das Skript `nfrun.sh` auf. Dieses liest die Job-Control-Datei aus, und erhält so die nötigen Parameter, die beim anschließenden Brennen an `cdrecord` übergeben werden (Geschwindigkeit, Disc At Once, Track at Once, CD-Mode, löschen von CD-RW, etc.). Anschließend wird der Benutzer über das LCD und auch über die Statusseite der WebGUI aufgefordert einen CD-Rohling einzulegen. Sobald der CD-Rohling in den Brenner eingelegt wurde beginnt `nfrun.sh` mit dem Löschen des Rohlings, sofern dies den Einstellungen in der Job-Control-Datei entsprechend gewünscht sein. Dieser Vorgang wird auch auf dem LCD ausgegeben. Anschließend ruft `nfrun.sh` ein weiteres Skript auf. Für eine Daten-CD wird das Skript `recdata.sh`, für eine Audio-CD das Skript `recaudio.sh` aufgerufen. Das entsprechende Skript brennt die vorbereiteten Daten aus dem vorher angelegten temporären Verzeichnis. Während des Brennens werden noch diverse Statusmeldungen auf dem LCD ausgegeben, wie z.B. die Fortschrittsanzeige des Brennvorgangs in Prozent. Dieselben Statusmeldungen werden auch dem `netfired` übergeben, der sie an die WebGUIweiterreicht. So können die Statusanzeigen auch noch an anderen Stellen der Benutzeroberfläche angezeigt werden. Wenn die CD erfolgreich gebrannt wurde, wird dies dem Skript `nfrun.sh` mitgeteilt. Dieses gibt ein OK oder wenn nötig eine Fehlermeldung auf dem LCD aus, und reicht diese auch an den `netfired` weiter. Abbildung 4.5 zeigt den Datenfluß der des Brennvorgangs im Detail.

Der `netfired` benutzt einen Scheduler um zu kontrollieren wann die Daten in der Queue vorbereitet werden, d.h. wann aus ihnen Images erstellt werden, die später durch einen Aufruf von cdrecord gebrannt werden können. Der `netfired` überprüft so die ersten beiden Einträge in der Queue. Es werden immer nur die ersten beiden Aufträge in der Queue vorbereitet, wenn sie bisher noch nicht als ISO, oder für eine Audio-CD als WAV-Dateien, vorliegen. Wenn ein Auftrag in der Queue, noch nicht in brennbarer Form vorliegt, so erstellt der `netfired` ein temporäres Verzeichnis, in das temporäre Dateien bei der Erstellung des zu brennenden ISO-Images, so wie am Schluß das fertige ISO-Image abgelegt werden. Anschließend ruft dieser das Skript `nfprepare.sh` auf. Dieses gibt auf dem LCD über den `lcdfired` die Mitteilung aus, daß die CD vorbereitet wird. Anschließend liest dieses Skript die Job-Control-Dateiaus und ermittelt den Typ der CD, die gebrannt werden soll (ISO, Audio, gepackte Daten, mehrere Dateien, etc.). Wenn der Typ ermittelt ist, werden ein oder mehrere Hilfsskripte aufgerufen, die die Daten zu einem fertigen ISO aufbereiten, das anschließend gebrannt werden kann. Die fertigen Daten werden dann in das reservierte temporäre Verzeichnis verschoben. Anschließend bekommt der `netfired` über das Skript `nfprepare.sh` eine Rückgabe, daß der Job nun fertig vorbereitet ist. Außerdem wird das erfolgreiche Beenden der Vorbereitung auch mit einer Mitteilung auf dem LCD angezeigt.

4.1.1.5 Status abfragen und Löschen eines Jobs

Nachdem ein Benutzer dem System einen Auftrag zum Brennen gegeben hat, hat er jederzeit die Möglichkeit, den Status seines Auftrags zu erfahren. Die möglichen Statusinformationen, die angezeigt werden können, sind:

- Job-ID: Laufende Nummer des Auftrags
- Start-Time: Zeitpunkt zu dem der Auftrag in die Queue eingefügt wurde
- Estimated-Time: Geschätzte Dauer des Brennauftrags

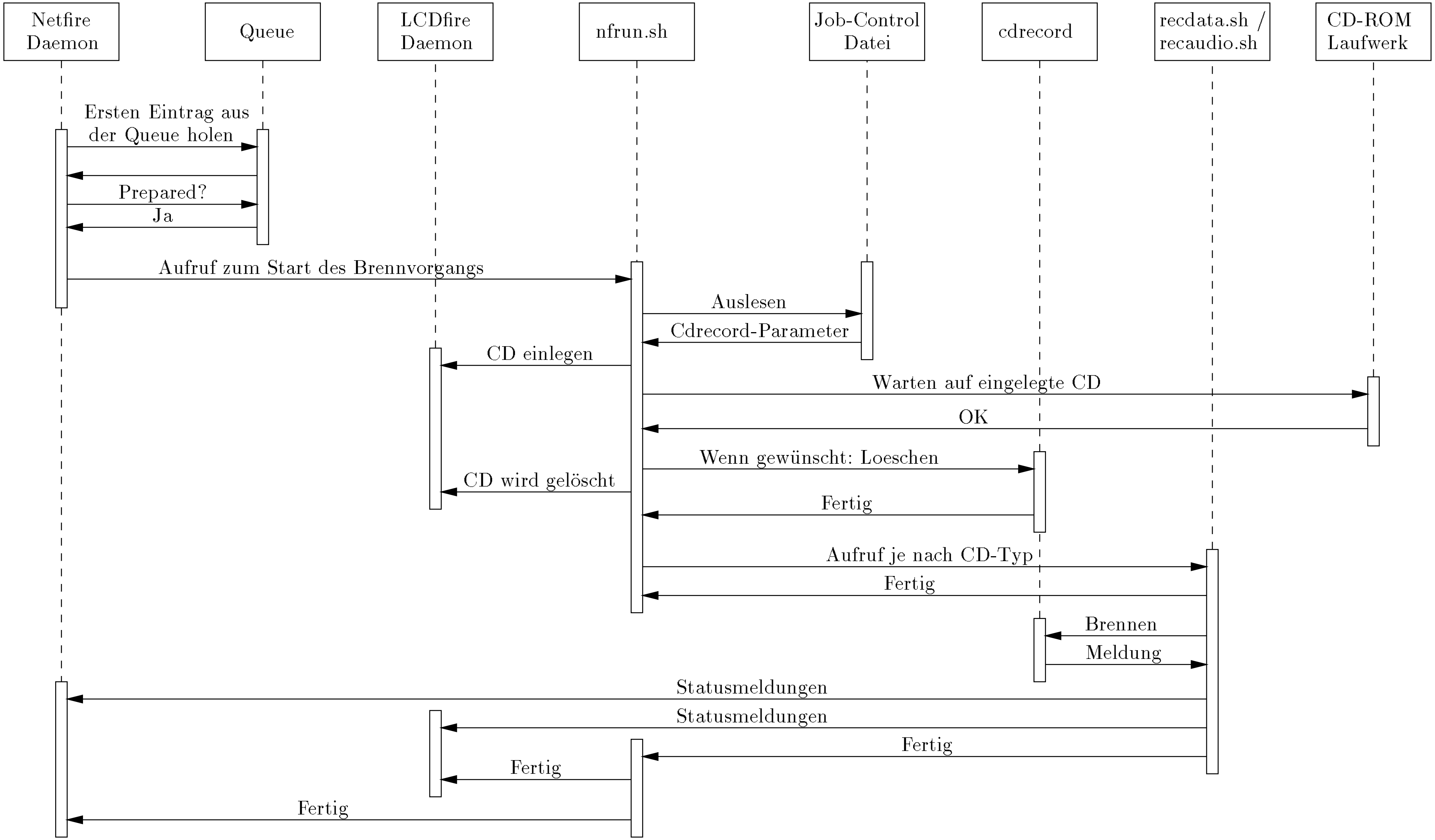

Abbildung 4.5: Ablauf beim Brennen der CD

Weiterhin hat der Benutzer die Möglichkeit, seinen Auftrag komplett zu löschen. Der `netfired` wird dann den Auftrag aus der Queue löschen, und ein PHP-Skript wird die Job-Control-Datei löschen. Abbildung 4.6 zeigt den Datenfluß für die Abfrage des Status und das Löschen eines Brennauftrags.

Der Benutzer wählt zunächst die HTML-Seite zur Abfrage des Status aus. Die Statusabfrage wird über den Webserver an den `netfired` weitergereicht. Der Daemon liest dann alle Informationen, die für die Statusanzeige relevant sind aus seiner internen Queue. Dies sind:

- Job-ID: Laufende Nummer des Auftrags
- Account: Zugehöriger Account zur Job-ID
- Start-Time: Zeitpunkt zu dem der Auftrag in die Queue eingefügt wurde
- Estimated-Time: Geschätzte Dauer des Brennauftrags
- Locked: Flag, das angibt ob der Job noch gelöscht werden kann, oder gerade gebrannt wird.

Alle Statusinformationen werden an den Webserver zurückgegeben. Dieser zeigt in einer Tabelle alle Daten an, die für den Benutzer interessant sind. Der Benutzer hat an dieser Stelle die Möglichkeit einen gequeueten Auftrag zu löschen. Damit nicht jeder Benutzer jeden Auftrag löschen kann, ist ein Sicherheitsmechanismus vorgesehen. Der Benutzer muß den zur ausgewählten Job-ID gehörigen Account eingeben. Da diesen Account nur der Benutzer kennt, der auch den Auftrag zum Brennen aufgegeben hat, wird dieser als Quasi-Passwort zum Löschen benutzt. Der Befehl zum Löschen wird dann vom Webserver, zusammen mit dem Accountnamen, an den `netfired` geschickt. Dieser entfernt den Benutzeraccount, das Upload-Verzeichnis des Benutzers und die Job-Control-Datei. Anschließend bekommt der Benutzer eine Bestätigung, daß der Auftrag gelöscht wurde.

4.1.1.6 Administration des Systems

Die Administration des NetFire-Systems beinhaltet folgende Einstellungen:

- Systemeinstellungen des NetFire-Systems
- Änderung des Administrations-Passworts
- Löschen der Queue oder einzelner Queue-Einträge

Im folgenden werden diese drei Funktionen detailliert beschreiben.

Systemeinstellungen in der Administration

Die Systemeinstellungen, die der Administrator ändern kann sind die folgenden:

- IP des NetFire-Systems
- Hostname des NetFire-Systems
- Netmask des NetFire-Systems
- Default-Gateway des NetFire-Systems
- Smarthost, der benutzt werden soll, um Benachrichtigungs-Mails an die Benutzer zu verschicken

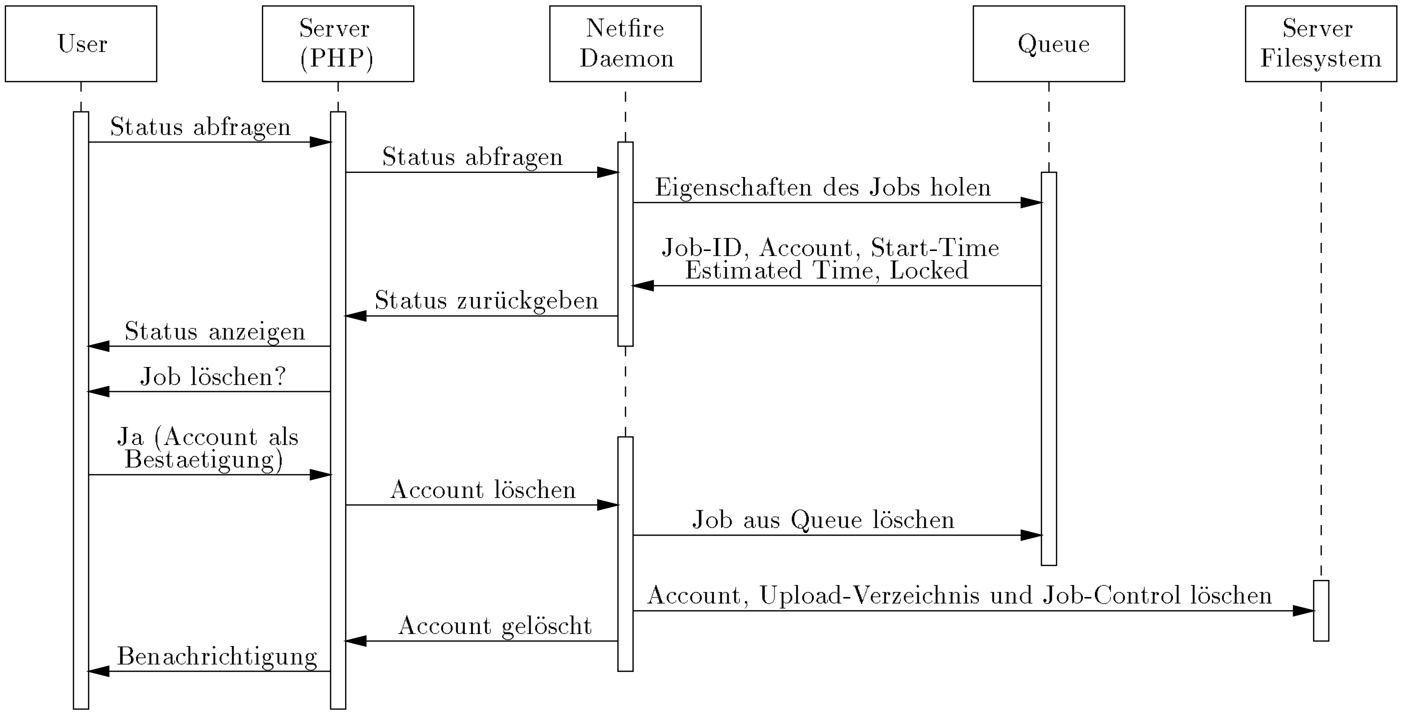

Abbildung 4.6: Datenfluß für die Abfrage des Status und das Löschen eines Brennauftrags

- SCSI-Brenneradresse unter der der Brenner angesprochen wird, bestehend aus LUN, ID, BUS

Abbildung 4.7 zeigt den genauen Datenfluß für die Änderung der Systemeinstellungen durch den Administrator.

Zunächst meldet sich der Administrator mit einem Passwort über das HTML-Interface an. Nachdem das Passwort über den `htaccess`-Mechanismus des Webservers überprüft wurde, hat der Administrator Zugriff auf die Systemeinstellungen. Ein PHP-Skript liest dann die notwendigen Systemdateien des NetFire-Systems aus und zeigt dem Administrator die aktuelle Konfiguration an. Dieser kann nun alle Einstellungen ändern, bei Bestätigung werden die Einstellungen in eine temporäre Systemdatei geschrieben, da der Benutzer des Webservers keine Schreibrechte auf die Konfigurationsdateien hat. Anschließend wird ein Konfigurationsskript aufgerufen. Dieses setzt die Einstellungen systemweit in den einzelnen Konfigurationsdateien.

Statusanzeige und Queue löschen für den Administrator

Die Anmeldung des Administrators verläuft genau wie bereits oben beschrieben. Es werden jedoch keine Konfigurationsdateien ausgelesen sondern die interne Queue des `netfired`, die die zu brennenden Aufträge enthält. Diese werden über den Webserver dem Administrator angezeigt, der nun einzelne Queueeinträge oder die gesamte Queue zum Löschen auswählen kann. Die zu löschenden Queueeinträge werden dem NetFire-Daemon übergeben. Dieser löscht zunächst alle Einträge seiner Queue und anschließend alle noch bestehenden Accounts und die dazugehörigen Verzeichnisse mit Inhalt. Anschließend bekommt der Administrator die Bestätigung für die erfolgreiche Löschung der Queue. Abbildung 4.8 zeigt den Datenfluß, der das Löschen der gesamten Queue vom Administrator darstellt.

Passwortänderung in der Administration

Zur Passwortänderung meldet sich der Administrator wieder zunächst mit seinem bisherigen Passwort an. Dieses wird vom Webserver anhand des `htaccess`-Mechanismus überprüft. War das Passwort korrekt, so wird dem Administrator die HTML-Seite zum Ändern des Passworts angezeigt. Dieser muß hier zur Verifikation noch einmal sein altes Passwort und das neue Passwort zwei mal eingeben. Die doppelte Eingabe des neuen Passworts dient dazu Tippfehler bei der Passworteingabe zu vermeiden. Nachdem der Administrator die Eingaben abgesendet hat, wird vom Webserver nach der Überprüfung des alten Passworts, das neue Passwort verschlüsselt und in die `passwd`-Datei geschrieben, wo es für zukünftige Passwortabfragen dem `htaccess`-Mechanismus des Webservers zur Verfügung steht.

4.2 Systemsoftware-Entwicklungen

In diesem Abschnitt wird näher auf die Eigenentwicklungen im Betriebssystembereich eingegangen. Das betrifft sowohl die Weiterentwicklungen an Emdebian, das von uns gewählte Betriebssystem, als auch die Erstellung des Treibers für das LCD und die Installations CD.

4.2.1 Software zur Installation

Aufbau der Installations CD

Die Installations CD besteht aus einem minimalen Linux System (in Form einer RAM Disk), einem Kernel und dem gepackten Archiv des zu installierenden Systems – Emdebian. Diese CD ist bootfähig und installiert ohne Eingriff des Benutzers die gesamte für NetFire notwendige Software auf dem Zielsystem. Es ist darauf zu achten, vorsichtig mit der Benutzung der

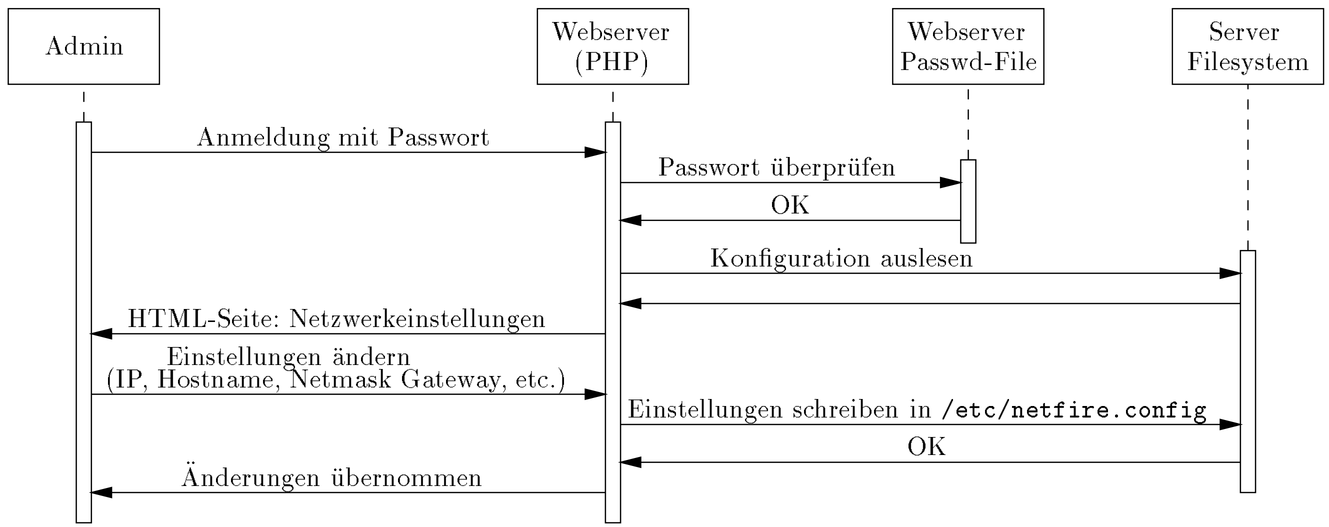

Abbildung 4.7: Datenfluß für die Systemeinstellungen des Administrators

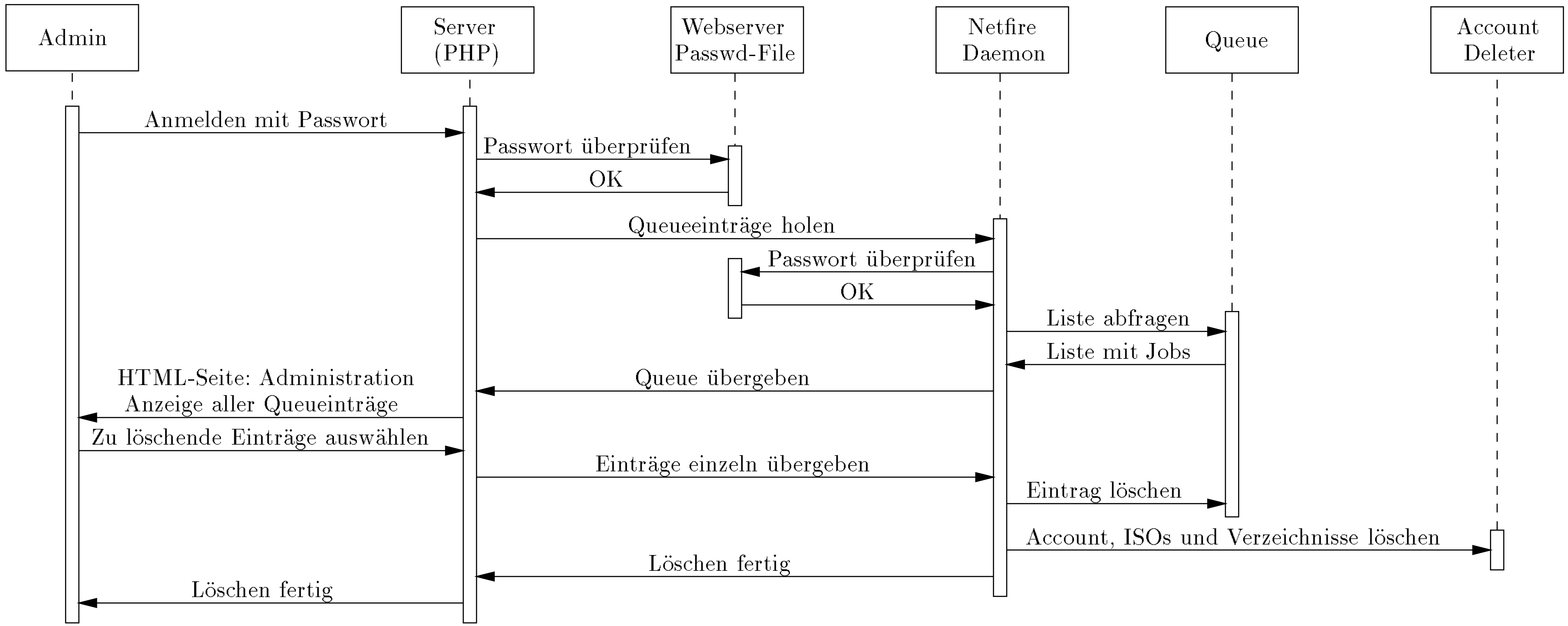

Abbildung 4.8: Datenfluß für das Löschen der gesamten Queue oder einzelner Queueeinträge durch den Administrator

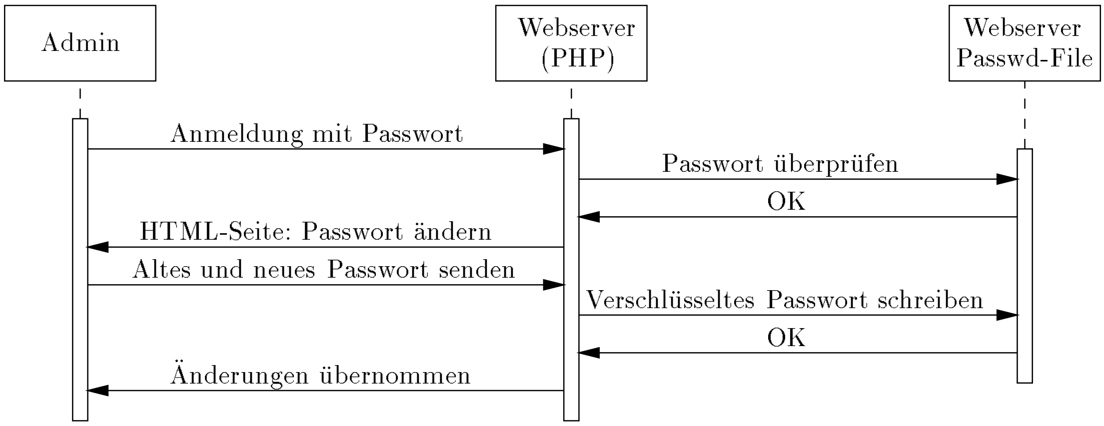

Abbildung 4.9: Datenfluß für das Ändern des Administrationspassworts

Installations CD umzugehen. Während der Installation werden – ohne Warnung – da jegliche Daten auf der ersten IDE Festplatte (Primary Master) gelöscht.

Unterstützte Hardware

Der auf der Installations CD befindliche Kernel unterstützt nur IDE Systeme, Defaultwert für die Zielfestplatte ist Primary Master, für das Quellen CD Laufwerk Secondary Master.

Installationsablauf

Die Installation durchläuft nach dem Booten folgende Schritte

- Partitionieren der Primary Master Festplatte
- Formatieren der erzeugten Partitionen
- Auspacken des Emdebian Archivs
- Vorbereiten des Systems auf einen Reset des Benutzers und Auswerfen der Installations CD

Beginn und Ende der Installation werden dem Benutzer durch Meldungen auf dem LCD sichtbar gemacht.

Anmerkungen

Mit den oben genannten Punkten ist die Installations CD unflexibel. Änderungen oder Abweichungen der benutzten Hardware erfordern ein Anpassen der Installationsskripte oder des Kernels. Das kommt daher, daß auf jegliche Benutzereingaben während der Installation verzichtet wird. Ziel ist es gewesen, eine Installationsroutine zu entwickeln, die auf einem NetFire System optimal funktioniert – schließlich sind an einem NetFire System standardmäßig weder Monitor noch Tastatur angeschlossen.

4.2.2 Embedded Debian Weiterentwicklung

Nach der Entscheidung der PG NetFire, Linux als Betriebssystem einzusetzen, stellt sich die Frage, welche Distribution eingesetzt werden soll. Grundsätzlich gibt es zwei Möglichkeiten: eine Desktop-Distribution wie SuSE oder RedHat oder eine Distribution, die speziell auf die Anforderungen von Embedded Systems zugeschnitten ist. Wir haben uns für eine Embedded

Distribution entschieden, da diese platzsparender ist und wir uns damit die Option offen halten, das NetFire System auch mit einem Flashspeicher zu booten. Bei der Evaluierung von freien Embedded Linux Distributionen ist aufgefallen, daß nur wenige flexible und gut skalierende Lösungen verfügbar sind. Eine besonders elegante Lösung stellt das Emdebsys System von Embedded Debian dar. Nach Diskussion der Möglichkeiten hat sich die PG zum Einsatz von emdebsys entschieden, obwohl klar war, daß einige Ressourcen zur Weiterentwicklung benötigt werden.

Das größte Problem beim Zusammenstellen kleiner Linux-Distributionen besteht darin, daß die Abhängigkeiten zwischen den Programmen, Libraries und Konfigurationsdateien nicht bekannt sind; und in dem großen Platzbedarf der meisten Programme, aus denen die aktuellen Distributionen bestehen. Emdebsys[3] erleichtert die Zusammenstellung einer solcher Distributionen.

Emdebsys ist ein flexibles System zur Erstellung einer Distribution für Embedded Systems. Es besteht aus einem Frontend mit der Konfigurationssprache CML2[4] und einem in Python geschriebenen Backend, das die Konfigurationsdatei ausliest und entsprechend der Vorgaben ein passendes Rootdateisystem mit passendem Kernel erzeugt.

Zum Zeitpunkt der Entscheidung wurde emdebsys nicht weiterentwickelt. Die verfügbare Version basierte auf CML2 0.7.6 und das Backend war in einem Prototyp Stadium. Die erste Weiterentwicklung von emdebsys im Rahmen der PG basierte auf CML2 1.8.4 und hatte ein flexibleres Backend, das jetzt über Konfigurationsdateien parametrisiert wurde (im Gegensatz zum Prototypen, der diese Parameter hart im Quellcode kodiert hatte). Im Laufe der Zeit wurde emdebsys von der PG immer weiter entwickelt und flexibler und mächtiger gemacht. Aus dem Prototypen ist ein gut funktionierendes Programm geworden, das wieder die Aufmerksamkeit der *Community* auf sich gezogen hat und derzeit auf Sourceforge[5] mit tatkräftiger Unterstützung der PG weiterentwickelt wird. Die momentane Version basiert auf CML2 2.1.0, integriert das Erzeugen des Kernels, kann Software aus Debian Paketen oder aus Quellcodes generieren und auf das Zielsystem installieren, hat einen flexiblen Mechanismus, um Scripte auf dem Zielsystem auszuführen, und eine auf Docbook basierte Dokumentation.

4.2.2.1 Einleitung

Wenn immer wir eine Distribution für einen neuen Rechner zusammenstellen wollen, begegnen wir dem Problem, passende Programme zu finden, die uns mit der gewünschten Funktionalität bedienen. Auf der anderen Seite müssen alle Abhängigkeiten zwischen den Programmen und ihren Konfigurationsdateien sowie den Libraries, gegen die sie gelinkt sind, erfüllt werden. Dieser Prozeß erfordert nicht nur enormes Wissen über Linux sondern ist auch sehr zeitraubend, denn er erfordert die Installation aller Programme, Modifikation vieler Konfigurationsfiles, anlegen der Einträge in `/dev` und *eine große Menge Debugging*. Das Programm Emdebsys soll dabei helfen, eine komplette Linux-Distribution zusammenzustellen, angefangen bei der Kernelkonfiguration, über die Wahl der Programme, Compilierung dieser, bis hin zur Konfiguration und Erzeugung des entsprechenden Rootfilesystems. Durch den Einsatz von Debian Paketen, die allerdings nie komplett installiert werden, sondern stets nur ausgewählte Komponenten, kann Emdebsys einen Kernel und ein Dateisystem für 11 Architekturen[6] erzeugen.

4.2.2.2 Systemvoraussetzungen

Um emdebsys zu benutzen und ein neues Rootfilesystem konfigurieren und erzeugen zu können, braucht man:

[3] `http://emdebian.sourceforge.net/emdebsys.html`
[4] `http://www.tuxedo.org/~esr/cml2/`
[5] `http://www.sourceforge.net`
[6] alpha, arm, hppa, i386, ia64, m68k, mips, mipsel, powerpc, s390, sparc

- Python 2.0 oder neuer, um das Filesystem zu konfigurieren und zu erzeugen
- Perl, um einige Skripte auszuführen
- Bourne Shell (s. Perl)
- Standardlinuxtools: `mount, umount, eject, dd, losetup, mkfs...`
- Crosscompiler, um die Quellcodes der Programme, die von emdebsys unterstützt werden, zu kompilieren (optional)
- Quellcodes des Kernels und einiger der Programme, deren Compilierung von emdebsys unterstützt wird, z.B.:
  - busybox
  - tinylogin
  - uClibc

Sind die genannten Programme installiert, kann mit dem Erzeugen des Emdedded Linux Dateisystems fortgefahren werden. Um das zu tun, müssen folgende Kommandos im Verzeichnis `emdebsys` ausgeführt werden:

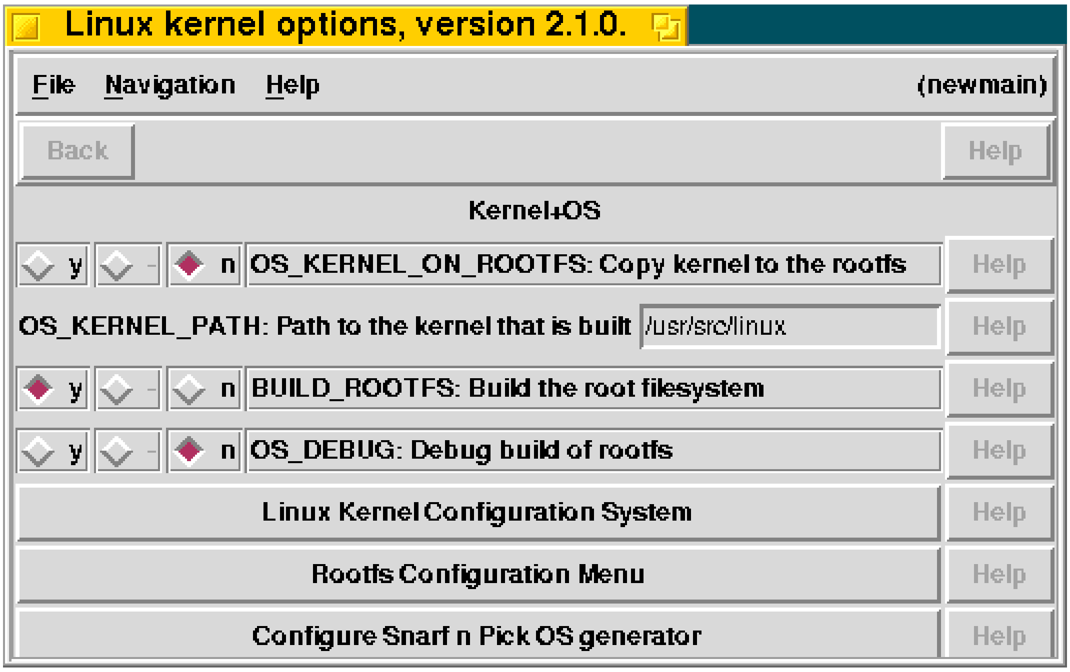

Abbildung 4.10: Konfigurationsprogramm von Emdebsys

- Die Menüs und die Regeln erzeugen:
  `./cmlcompile.py kernel+os-config.cml`
- Konfigurieren des gewünschten Emdedded Linux Dateisystems und die Konfiguration unter `config.out` abspeichern:
  `./cmlconfigure.py`
- CML2 in den Kernel installieren:
  `./install-cml2` (optional)

- Ins Verzeichnis `emdebsys/snp/` wechseln und snp anpassen. Dazu muß normalerweise die Datei `sources.list` angepasst werden, je nachdem wo sich der nächste Debian Spiegel befindet und ob eigene Debian Pakete installiert werden sollen:
  `cd $EMBEDSYS/snp/`.

- Das Embedded Linux Dateisystem erzeugen mittels snp.py:
  `./snp.py -c ../config.out`

Per Voreinstellung wird das Embedded Linux Dateisystem im Verzeichnis `snp/rootfs/` erzeugt. Ein komprimiertes `ext2` Image wird unter `/tmp/emdeb-rootfs.img.gz` angelegt. Zusätzlich wird eine komprimierte tar Datei mit den Dateien unter `/tmp/emdeb-files.tgz` generiert. Dies kann aber mittels der Datei `.snprc` und der Kommandozeile nach eigenen Wünschen angepaßt werden.

4.2.2.3 Konfigurieren und Starten von `snp.py` („Snarf'n'Pick OS generator")

Konfiguration

Das Programm `snp.py` erzeugt aus den gewählten Konfigurationsoptionen ein komplettes Embedded Linux Dateisystem, inklusive Compilieren aller benötigten Programme und dem Kopieren der gewählten Kernelmodule und des Kernels selbst.

Kommandozeilenoptionen von snp.py

- `-h`
  Gibt einen kurzen Hilfetext aus.

- `-V`
  Gibt die Version des Programms aus.

- `-v`
  Erhöht die Ausführlichkeit der erzeugten Ausgabe.

- `-c` *Dateiname*
  Wählt die Datei aus, die die Konfiguration des Embedded Linux enthält.

- `-s` *Dateiname*
  Wählt eine alternative `snp.conf` Datei aus.

- `-r` *directory*
  Wählt ein alternatives Verzeichnis zum Bauen des Embedded Linux Dateisystems.

- `-o` *filename*
  Wählt den Namen des `objdump` Programms aus.

Neben der Kommandozeile gibt es zusätzlich die Datei `.snprc`. Hier können zusätzliche Optionen gesetzt werden. Die Syntax der Datei hat die Form: `Option`=Wert, getrennt per Zeilenvorschub. Zeilen die mit einem „#" beginnen werden, als Kommentare gewertet, leere Zeilen sind erlaubt. Ein Beispiel:

```
# All the lines starting with # are comments, blank lines are allowed.
verbose=2
rootfs_image=/root/angelboot/rootfs.image
```

Folgende Optionen können per `.snprc` gesetzt werden

- `config_file=`*filename*
  Konfigurationsdatei
  (Voreingestellt: `config.out`).
- `cross_compile=`*prefix*
  Crosscompile-Prefix
  (Voreingestellt: „nichts“).
- `devices_file=`*filename*
  Die Datei, die beschreibt, welche Gerätedateien angelegt werden sollen.
  (Voreingestellt: `devices.conf`).
- `snpconf_file=`*filename*
  Alternative Datei zu `snp.conf`
  (Voreingestellt: `snp.conf`).
- `rootfs_dir=`*directory*
  Das Verzeichnis, in dem das Embedded Linux Dateisystem erzeugt wird.
  (Voreingestellt: `rootfs/`).
- `rootfs_size=`*size*
  Die Standardgröße des zu erzeugenden Dateisystem-Images in Kb
  (Voreingestellt: 8192).
- `rootfs_type=`*type*
  Typ des zu erzeugenden Embedded Linux Dateisystem-Images (ext2, vfat... etc).
  (Voreingestellt: `ext2`).
- `rootfs_image=`*filename*
  Name der Image-Datei des Embedded Linux Dateisystem-Images
  (Voreingestellt: `/tmp/rootfs.img`).
- `rootfs_tgz=`*filename*
  Name der erzeugten .tgz Datei des Linux Dateisystem-Images
  (Voreingestellt: `/tmp/rootfs.tgz`).
- `rootfs_gzip=`*bool*
  Soll das Linux Dateisystem-Image per gzip gepackt werden
  (Voreingestellt: y).
- `sources_list=`*filename*
  Die Datei, die die Quellen der Debian Pakete angibt.
  (Voreingestellt: `sources.list`).
- `cache_dir=`*directory*
  Verzeichnis, in dem die heruntergeladenen Debian Pakete zwischengespeichert werden.
  (Voreingestellt: `snp-cache/`).
- `src_cache_dir=`*directory*
  Das Verzeichnis, in dem die Quellcode Dateien lagern.
  (Voreingestellt: `src-cache/`).
- `sources_conf_file=`*filename*
  Die Datei, die die Konfiguration beim Compilieren von Quellcodepaketen enthält.
  (Voreingestellt: `sources.conf`).

- `tmp_dir`=*directory*
  Das Verzeichnis, in dem temporäre Dateien erzeugt werden.
  (Voreingestellt: `/tmp`).

- `cdrom_dir`=*directory*
  Der Pfad, in dem das CD-ROM eingehängt wird.
  (Voreingestellt: `/mnt/cdrom`).

- `cdrom_dev`=*device name*
  Die Gerätedatei des CD-ROM Laufwerks.
  (Voreingestellt: `/dev/cdrom`).

- `templates_dir`=*directory*
  Verzeichnis, in dem die grundlegende Dateistruktur festgelegt wird.
  (Voreingestellt: `../templates/`).

- `objdump`=*filename*
  Welche `objdump` Variante benutzt werden soll.
  (Voreingestellt: „nichts").

- `verbose`=*number*
  Legt die Ausführlichkeit der erzeugten Ausgaben fest.
  (Voreingestellt: 2).

4.2.2.4 Funktionsweise von snp.py

Nach dem Ausführen des Programms liest `snp.py` seine Optionen aus der Datei `.snprc` und von der Kommandozeile. Dann wird die Konfigurationsdatei, die per Kommandozeile oder der Konfigurationsoption `config_file` gewählt wurde, gelesen und die entsprechende Architektur für das Dateisystem gesetzt. Mittels der Datei `sources_list` werden die Dateien des Debian Spiegels geprüft, daraus die Paketliste gebaut, und anschließend werden die benötigten Debian Pakete in das Verzeichnis `cache_dir` heruntergeladen. Als nächstes werden die gewählten Dateien aus den Debian Paketen extrahiert und eventuell benötigte Bibliotheken ebenfalls auf das Dateisystem installiert. Anschließend werden die Programme, die aus den Quellen installiert werden, kompiliert und auf das Zieldateisystem kopiert (die Quellen liegen im Verzeichnis `src_cache_dir`, die Benennungen werden mittels `sources_conf_file` festgelegt). Dann werden die Einträge im `/dev` Verzeichnis erzeugt, dazu wird die Datei `devices_file` ausgewertet. Die Dateien in `/etc` dagegen werden im wesentlichen durch das Verzeichnis `templates/` erzeugt. Danach werden zahlreiche Konfigurationsoptionen mittels Skripten gesetzt, die jetzt ausgeführt werden (die Skripte liegen im Verzeichnis `scripts/`). Als letztes wird das Dateisystemimage erzeugt. Dabei sind die Optionen `rootfs_size`, `rootfs_image`, `rootfs_type` und `rootfs_gzip` von Interesse. Zu beachten ist, daß man im allgemeinen das Dateisystem nur als Superuser erzeugen kann, da man nur als Superuser Gerätedateien im Verzeichnis `rootfs/dev/` anlegen darf. Die nachfolgende Grafik verdeutlicht die Funktionsweise.

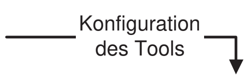

Abbildung 4.11: Funktionsweise von snp.py

4.2.3 Displaytreiber

4.2.3.1 Problemstellung

Um Display und Taster anzusteuern, muß es eine softwareseitige Schnittstelle geben. Folgendes Konzept wurde geplant:

- Erstellen eines Linux-Device-Treibers als Kernel-Modul für das pixelweise Ansteuern des Displays
- Erstellen einer Userspace-Library für die Verwendung von Zeichensätzen und Cursorsteuerung (siehe 4.4.2.1).
- Erstellen von NetFire-spezifischen Applikationen für die Benutzerführung (siehe 4.4.2.2 und 4.4.2.3).

4.2.3.2 Funktionsweise von Linux Kernel-Modulen

Ein Linux-Kernelmodul wird bei Systemstart geladen. Die Interaktion mit dem Benutzer und den Applikationen findet durch ein Device-File statt. Alle Zugriffe auf dieses File werden an entsprechende Funktionen des Kernel-Moduls weitergeleitet.

4.2.3.3 Anforderungen an den LCD-Treiber

Das Kernelmodul für die Displaysteuerung soll folgenden Ansprüchen genügen:

- Sequenzen von Bytes werden als Grafikdaten an das Display übertragen
- Für Pixelmanipulationen können aktuelle Grafikdaten ausgelesen werden
- Der Zustand der Tasten ist abfragbar
- Einfache Funktionen wie Display-Löschen, Festlegen der Schreibposition, etc. werden unterstützt
- Zugriffssteuerung durch Lock-Mechanismus

4.2.3.4 Anbindung an das System

Der Device-Treiber wird als Module geladen. Er ist ein „Character-Device“ (überträgt byteweise Datenströme) und ist über die Major-Number 240 an das System angebunden. Unter der Minor-Number 0 stehen Funktionen für den Zugriff auf das Display, unter Minor-Number 1 steht Funktionalität für das Auslesen der Taster bereit.

Damit ein Modul automatisch gestartet wird, sollte es zu seiner Major-Number einen entsprechenden Eintrag in der Modulkonfiguration geben (üblicherweise `/etc/modules.conf`). Außerdem müssen für den Zugriff entsprechende Devicefiles für die Major- und Minornumbers angelegt werden. Dies geschieht über das Kommando „mknod“. Diese Files sollten `/dev/lcd` (240,0) und `/dev/lcduser` heißen.

4.2.3.5 Zusätzliche Funktionen (ioctl)

Unter der Minor-Number 0 stehen zusätzliche Funktionen zur Verfügung, die über den ioctl-Mechanismus aufgerufen werden können (siehe Tabelle 4.1).

ioctl-Nummer	Name	Parameter	Funktion
0	LCD_HOME	n.V.	Home, folgende Bytes ab Displaybeginn
1	LCD_HARDRESET	n.V.	Reset über Reset-Leitung des Displays
2	LCD_SOFTRESET	n.V.	Reset über Reset-Sequenz
3	LCD_GETPOS	n.V.	Aktuelle Position (Byteposition)
4	LCD_CLRSCR	n.V.	Löschen des Displays
5	LCD_SETPOS	position	Setzen der Position
6	LCD_VERSION	n.V.	Rückgabe der Versionsnummer
7	LCD_OFF	n.V.	Abschalten der Displaycontroller
8	LCD_ON	n.V.	Einschalten der Displaycontroller

Tabelle 4.1: ioctl Befehle für den LCD-Treiber

4.2.3.6 Interna

Intern gibt es drei verschiedene Funktionstypen:

Lowlevel-Funktionen sind solche Funktionen, die für die Ansteuerung einzelner Leitungen des Displays verantwortlich sind. Alle Signale werden direkt an den Port angelegt. Dazu wird erst die Variable *control_mirror* geändert, welche dann durch die Funktion `w_ctr()` auf den Port ausgegeben wird. Anschließend wird für 100ns gewartet. So wird sichergestellt, daß alle Signale einzeln und innerhalb der vorgegebenen Schaltzeiten gesetzt werden.

Highlevel-Funktionen fassen verschiedene Lowlevel-Operationen zu einem sinnvollen Kommando zusammen. Die Funktion `lcd_opcode(...)` z.B. sendet unter Berücksichtigung des Handshake-Mechanismus Daten an das Display. Hierzu sind diverse Lowlevel Zugriffe erforderlich.

Module-Funktionen dienen der Kommunikation mit dem System. Sie melden das Kernel-Module an oder ab, nehmen Daten vom Devicefile entgegen und sorgen für die Verwaltung der internen Datenstrukturen.

Es sind folgende Funktionen im Module implementiert:

- `static inline unsigned char r_str();`

  Diese Funktion liefert den Status des Parallelports (invertierte Bits). [lowlevel]

- `static inline void w_ctr();`

  Schreibt den Control-Mirror in das Control-Register des Parallelports und wartet 100ns. [lowlevel]

- `static inline void lcd_data(unsigned char y);`

  Schreibt y in das Datenregister des Parallelports. [lowlevel]

- `static inline void lcd_e_low();`

  Setzt E1 und E2 auf low [lowlevel]

- `static inline void lcd_e1_high();`

  Setzt E1 auf high. [lowlevel]

- `static inline void lcd_e2_high();`

  Setzt E2 auf high. [lowlevel]

- `static inline void lcd_a0_high();`
  `static inline void lcd_a0_low();`

  Setzt A0 auf high, bzw. low. [lowlevel]

- `static inline void lcd_read();`

  Setzt die R/W-Leitung am Display auf high (Display im Write-Modus, der Treiber liest Daten). [lowlevel]

- `static inline void lcd_wait();`

  Wartet auf Busy-Low am Display. Seiteneffekt: Display wird in den Zustand „Read Status" geschaltet. [lowlevel]

- `static inline void lcd_write();`

  Setzt die R/W-Leitung am Display auf low (Display im Read-Modus, der Treiber schreibt Daten). [lowlevel]

- ```
  static inline void lcd_opcode(int a0, unsigned char data, int left,
                                int right);
  ```

  Schreibt *data* in die durch die booleschen Werte *left* und *right* spezifizierte(n) Displaycontroller. Die Leitung A0 wird entsprechend dem Wert in dem booleschen Parameter *a0* gesetzt. [highlevel]

- `static inline void lcd_reset();`

  Führt einen Hardware-Reset am Display durch und initialisiert es anschließend. [highlevel]

- `static void lcd_parport_claim(struct pardevice *dev);`

  Reserviert den in *dev* spezifizierten Parallelport. [module]

- `static void lcd_parport_release(struct pardevice *dev);`

  Gibt den durch *dev* spezifizierten Parallelport frei. [module]

- `static void lcd_putbyte(unsigned char data);`

  Das Byte *data* wird in das Display-RAM geschrieben. Alle nötigen Parameter werden angepaßt (z.B. Position) und eine Kopie des Datenbytes im Speicher abgelegt. Wenn nötig, werden am Display die richtige Bank und Spalte entsprechend der Position eingestellt. [highlevel]

- ```
  static ssize_t lcd_fwrite(struct file * file, const char * buf,
                            size_t count, loff_t *ppos);
  ```

  Device-Funktion. Alle Schreibzugriffe auf das Device werden hier ausgewertet. Schreibzugriffe auf Minor 0 werden durch `lcd_putbyte(...)` in das Display-RAM geschrieben, alle anderen Zugriffe liefern einen Fehler. [module]

- `static unsigned char lcd_getbyte(void);`

  Liest ein Datum von der Kopie des Display-RAMs ab der aktuellen Position (und inkrementiert diese). [module]

- `static char lcd_getkeys(void);`

  Liest den Status der Taster und gibt diese als Bitcode zurück (Bit 0=Taster 1; Bit 1=Taster 2). [lowlevel]

- ```
  static ssize_t lcd_fread(struct file * file, char * buf,
                           size_t count, loff_t * ppos);
  ```

  Alle Lesezugriffe auf das Device-File werden hier abgearbeitet. Wird Minor 0 gelesen, erfolgt die Rückgabe von Werten über die Funktion `lcd_getbyte()`, wird Minor 1 gelesen resultiert der Rückgabewert aus der Funktion `lcd_getkeys()`. [module]
  ```

- `static int lcd_ioctl(struct inode *inode, struct file *file, unsigned int cmd, unsigned long arg);`

  Alle ioctl-Zugriffe auf das Device-File laufen über diese Funktion. [module]

- `static int lcd_open(struct inode * inode, struct file * file);`

  Alle Anfragen, das Device-File zu öffnen, werden hier abgearbeitet. Hier befindet sich auch der Locking-Mechanismus, um mehrmaligen Schreibzugriff zu unterbinden. [module]

- `static int lcd_release(struct inode * inode, struct file * file);`

  Alle Anfragen, das Device-File zu schließen, werden hier abgearbeitet. [module]

- `int init_lcd(void);`

  Initialisierung des Kernel-Modules. Diese Funktionen macht dem Kernel die Device-Funktionen bekannt, alloziert Speicher, bindet das Module an Char-Major-240 und reserviert den Parallelport. [module]

- `int cleanup_lcd(void);`

  Diese Funktion wird vor dem Entfernen des Kernel-Modules aufgerufen. Sie gibt belegten Speicher frei, trennt das Module von Char-Major-240 und schließt die Kommunikation mit dem Parallelport. [module]

4.3 Software-Entwicklungen zum Abarbeiten der Brennaufträge

Zur Realisierung des NetFire-Systems wuerden einige Software-Komponenten neu entwickelt oder bestehende Software erweitert. Zentrale Instanz ist der NetFire-Daemon, der beim Booten des Systems gestartet wird und alle anderen Komponenten koordiniert. Die eigentliche Funktionalität des Daemons wurde in Skripte ausgelagert, die leicht angepasst werden können, ohne den Daemon neu kompilieren zu müssen. Außerdem wurde der eingesetzte FTP-Server erweitert und verändert, um Manipulationen am System auszuschließen und die Interaktion mit dem NetFire-Daemon zu ermöglichen.

4.3.1 NetFire-Daemon

Der `netfired` ist die Kontrollinstanz zwischen den Benutzeroberflächen und den eigentlichen Programmen, die für das Brennen einer CD verantwortlich sind.

Die groben Aufgabengebiete des `netfired` sind:

- Nutzerverwaltung
  Jeder Benutzer des Gerätes erhält eine Nutzerkennung für seine Arbeit. Es gibt eine Nutzerdatenbank, die vor allem für den FTP-Server wichtig ist. Diese Nutzerdatenbank wird vollständig durch den `netfired` gekapselt, Aktionen auf der Nutzerdatenbank laufen nur über den Daemon. Der `netfired` sorgt auch dafür, daß Daten von Nutzern gelöscht werden, sobald ihre Aufträge abgearbeitet sind.

- Queue für die Jobs
  Es gibt nur einen CD-Brenner in dem NetFire-System, daher müssen die Jobs nacheinander abgearbeitet werden. Der `netfired` übernimmt die Verwaltung und Steuerung der dafür nötigen Queue (Warteschlange).

- Abarbeiten von Jobs
  Die eigentliche Arbeit des Brennens übernehmen Skripte, die der `netfired` aufruft. Von diesen Skripten wird meistens ein Status zurückgeliefert, den `netfired` auswertet.

- Statusinformationen
  Die von den Skripten zurückgegebenen Statusinformationen werden zur Verfügung gestellt, um sie zum Beispiel auf der WebGUI anzuzeigen.

- Hilfsfunktionen für die WebGUI
  Der Webserver darf nicht direkt in die Home-Verzeichnisse der Nutzer schreiben. Um der WebGUI dennoch die Möglichkeit zu geben, Dateien zu erstellen, besitzt der `netfired` einige Hilfsfunktionen für diese Zwecke.

4.3.1.1 Aufbau des `netfired`

Grob gesehen ist der `netfired` ein Spooler, der eine Queue mit abzuarbeitenden Jobs verwaltet. Gesteuert wird der `netfired` über einen UNIX-Domain-Socket. Über eine Verbindung auf diesem Socket nimmt der Daemon Befehle entgegen und führt dementsprechend Aktionen aus. Die einzelnen Befehle und deren Bedeutung werden in Kapitel 4.3.1.2 beschrieben.

Die eigentliche Abarbeitung von Brennaufträgen wird von Skripten übernommen (siehe Kapitel 4.3.2). Der `netfired` bietet Scripthooks zu verschidenen Ereignissen an (siehe Kapitel 4.3.1.3). Tritt ein bestimmtes Ereignis auf, wird das an den Scripthook gebundene Skript ausgeführt, und muß sich um die weitere Abarbeitung eines Jobs kümmern. Dadurch ist gewährleistet, daß das ganze NetFire-System ohne Neukompilation schnell an neue Standardsoftware angepaßt werden kann. Die einzelnen Scripthooks beschreibt Kapitel 4.3.1.3.

`netfired` synchronisiert die Queue mit einer Textdatei. Jedem Job in der Queue wird eine eindeutige Job-ID zugewiesen. Die Textdatei enthält immer die Liste der Job-IDs, die in der Queue sind. Sollte das System einmal unplanmäßig abgeschaltet werden, so kann der `netfired` nach dem Neustart die Queue wiederherstellen.

Nach dem Start wird der Daemon initialisiert und ein Thread gestartet, der die Queue abarbeitet. Dieser Thread liest das erste Element aus der Queue aus, arbeitet den zugehörigen Job ab, und entfernt das Element aus der Queue.

Das Hauptprogramm besteht aus einer Schleife, die auf eine Verbindung über den Socket wartet. Für jede Verbindung wird ein eigener Thread gestartet, der die Befehle des Clients, der über den Socket verbunden ist, entgegennimmt.

Die Struktur des `netfired` zeigt Abbildung 4.12.

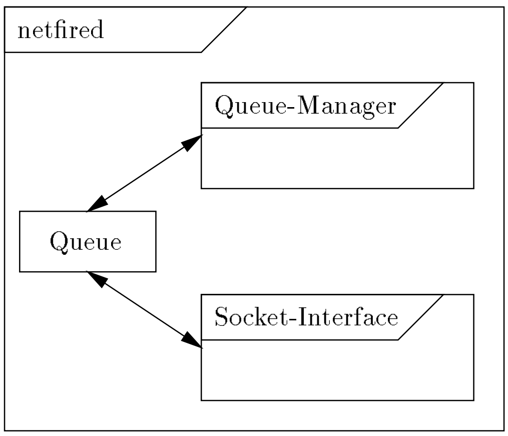

Abbildung 4.12: Struktur des `netfired`

Die Queue und auch die Datei mit den Accounts für den FTP-Server sind durch Semaphoren geschützt, um einen gleichzeitigen Zugriff von mehreren Threads auf sie zu unterbinden. Für jeden einzelnen Account ist ebenfalls ein Lockingmechanismus implementiert. Über diesen wird gewährleistet, daß sich WebGUI und das FTP-Interface nicht gegenseitig stören.

Die wichtigste Information für den `netfired` ist die Nutzerkennung. Über diese Nutzerkennung läßt sich das Verzeichnis mit den Daten des Benutzers ermitteln, und sie dient der Identifizierung des Benutzers gegenüber dem System. So darf nur der Benutzer die Jobs aus der Queue oder die Daten löschen, die seiner Nutzerkennung zugeordnet sind. Um die Nutzerkennungen für andere Benutzer unzugänglich zu machen, verwaltet der `netfired` zusätzlich eine Job-ID für jeden Job in der Queue. Diese Job-ID erscheint später auf allen Statusanzeigen, die von jedem lesbar sind.

4.3.1.2 Das Socket-Interface

`netfired` kommuniziert über einen UNIX-Domain-Socket mit der Außenwelt. Über diesen Socket nimmt er Befehle entgegen und liefert eine Rückmeldung. Befehle können Parameter haben, diese werden durch einen Zeilenvorschub (`\n`) getrennt. Ein Beispiel für einen Befehl zeigt Abbildung 4.13.

```
Kommando\nParameter1\nParameter2
```

Abbildung 4.13: Befehlsformat

Es ist wichtig, daß der Socket nach Absenden des Befehls für weiteren Datenfluß zum `netfired` hin geschlossen wird (Systemaufruf `shutdown(sock, 1)`). Ansonsten wartet der `netfired` noch auf weitere Daten und erkennt den Befehl nicht. Die Parameter selbst dürfen nie ein Stringabschlußzeichen (`\0`) oder einen Zeilenvorschub (`\n`) enthalten. Das Ende des letzten Parameters wird durch die Anzahl der über den Socket empfangenen Zeichen festgelegt. Auf den letzten Parameter dürfen `\0` und `\n` folgen.

Rückmeldungen

Nach jedem Befehl wird über den Socket eine Textmeldung zurückgeliefert. Jede Zeile der Rückgabe beginnt mit einem Buchstaben, der die Art der Meldung klassifiziert, und einem darauffolgenden Doppelpunkt. Daran schließt sich die eigentliche Meldung an. Ein `o` signalisiert eine erfolgreiche Ausführung des Befehls, ein `e` einen Fehler. Abbildung 4.14 zeigt zwei Beispielmeldungen.

`o:command succeded`
Ausführung war erfolgreich
`e:an error occured`
Es ist ein Fehler aufgetreten.

Abbildung 4.14: Rückgabeformat

Je nach Befehl können noch weitere Meldungstypen auftreten. Soweit nicht anders beschrieben, liefern die Befehle nur eine Zeile als Rückgabe. Sollten mehrere Zeilen übermittelt werden, sind sie durch `\n` getrennt.

Standardfehlermeldungen

- Falsche Parameter
  Alle Befehle, die einen oder mehrere Parameter haben, können `e:invalid arguments` zurückgeben, falls die Aufrufsyntax nicht korrekt ist.

- Falscher Account
  Befehle, die einen Accountnamen als Parameter haben, können `e:invalid account` zurückgeben, wenn der angegebene Account nicht existiert.

- Fehler beim Account-Locking
  Einige Befehle brauchen exklusiven Zugriff auf das Home-Verzeichnis, das zu dem Account gehört. Es wird vor der Operation versucht, das Verzeichnis zu locken. Falls dies fehlschlägt, sind folgende Fehlermeldungen möglich:

  – `e:account locked by a ftp session, try again`
    Der Account *account* ist durch eine Operation des FTP-Server gelockt. Solange der FTP-Server Dateitoperationen durchführt, darf der `netfired` keine weiteren ausführen.
  – `e:account locked by a netfired session, try again`
    Der Account *account* ist durch eine Operation des `netfired` gelockt. Solange der `netfired` Dateitoperationen durchführt, darf er keine weiteren ausführen.
  – `e:error locking account`
    Der Account konnte nicht gelockt werden, andere Prozesse können so das Löschen behindern. Der Account wurde daher nicht gelöscht.

Queue-Befehle

- Neuen Account anlegen

  Aufruf:
  `new`
  Legt einen neuen Account an.

  Rückgabe:

  – `o:`*account*
    Bei Erfolg wird der Name des Accounts als *account* zurückgeliefert.
  – `e:account limit reached`
    Das Accountlimit ist erreicht. Es dürfen keine weiteren Accounts angelegt werden, bevor nicht alte gelöscht werden.
  – `e:could not run accountmaker:` *error*
    Der `accountmaker` konnte nicht ausgeführt werden, *error* enthält den Fehlergrund.
  – `e:`*error*
    Der `accountmaker` war nicht erfolgreich, aus dem Grund *error*.
  – `e:error in accountmaker:` *error*
    Im `accountmaker` ist ein Fehler aufgetreten. *error* kann einen Fehlergrund enthalten. Dieser Fehler sollte eigentlich nicht auftreten, es sei denn der Accountmaker ist fehlerhaft.

- Account und Home-Verzeichnis löschen

  Aufruf:
  `del` *account*
  Löscht den Account *account*.

  Rückgabe:

  – `o:`
    Der Account *account* und das zugehörige Home-Verzeichnis wurden gelöscht.
  – Es können Account-Locking-Fehler auftreten (siehe weiter oben).

- Job in die Queue stellen

Aufruf:

`queue` *`account`*

Fügt den Account *`account`* als Job in die Queue ein.

Rückgabe:

Es können mehrere Zeilen, die mit dem Fehlercode `e:` anfangen, zurückgegeben werden.

- `o:`*`jobid`*
  Es wurde ein Job gequeued. Dieser Job ist unter der Job-ID *`jobid`* ansprechbar.
- `e:your job is already queued. jobid:`*`jobid`*
  Von diesem Account wurde bereits ein Job gequeued. Dieser hat die Job-ID *`jobid`*.
- `e:queueing error`
  Beim Einfügen des Jobs in die Queue ist ein Fehler aufgetreten.
- `e:`*`error`*
  `error` enthält den Fehlergrund, warum der Job nicht in die Queue gestellt wurde.
- Es können Account-Locking-Fehler auftreten (siehe Kapitel weiter oben).

- Job aus der Queue entfernen

Aufruf:

`dequeue` *`account`*

Entfernt den Job, der von dem Benutzer mit dem Account *`account`* gequeued wurde, aus der Queue. Alle Jobdaten bleiben erhalten, es werden keine Dateien gelöscht.

Rückgabe:

- `o:`
  Der zu dem Account gehörende Job wurde aus der Queue entfernt.
- `e:your job is currently being processed`
  Der zu dem Account gehörende Job wird gerade bearbeitet, und kann somit nicht gelöscht werden.

Status-Befehle

- Status (nach Account)

Aufruf:

`status_a` *`account`*

Liefert den Statusinformationen des Jobs, der zu dem Account *`account`* gehört.

Rückgabe:

- `o:`*`jobid pos f_locked f_prepared estimated_time percent f_status start_time queue_time jobtype`*
  Tabelle 4.2 zeigt die Bedeutung der einzelnen Positionen.
- `e:no job queued from this account`
  Von diesem Account wurde kein Job in die Queue gestellt.

- Status (nach Job-ID)

Aufruf:

`status_i` *`jobid`*

Liefert den Statusinformationen bezüglich des Jobs *`jobid`*. Der Befehl ist weitestgehend identisch zu `status_a`.

Position	Beschreibung
jobid	Job-ID des Jobs, der von diesem Account gequeued wurde.
pos	Queue-Position des Jobs
f_locked	Gibt an, ob der Job gelockt ist. Auf gelockten Jobs können keine Queue-Operationen (`dequeue`, `del`) angewendet werden. Diese schlagen sonst fehl. `0` Job ist nicht gelocked `1` Job ist gelocked
f_prepared	Gibt den Status der Vorbereitung des Jobs an: `0` noch nicht vorbereitet `1` vorbereitet `2` Vorbereitung läuft `3` Vorbereitung fehlgeschlagen `4` Vorbereitung wird nochmal wiederholt
estimated_time	erwartete Bearbeitungszeit in Sekunden
percent	Fortschritt der Abarbeitung in Prozent.
f_status	Status des Jobs: `0` Bisher unbearbeitet `1` In Vorbereitung oder vorbereitet. Details sind *flag_prepared* zu entnehmen `2` Wird gerade abgearbeitet `3` Job ist abgearbeitet
start_time	Zeit in Sekunden, bis die Abarbeitung des Jobs beginnt.
queue_time	Zeitpunkt, zu dem der Job gequeued wurde als UNIX-Timestamp
jobtype	Typ des Jobs (von `sh_nfcheck` bestimmt, siehe Kap. 4.3.1.3)

Tabelle 4.2: Rückgabepositionen des Befehls `status_a`

Rückgabe:

- `o:`*pos f_locked f_prepared estimated_time percent f_status start_time queue_time completed_time jobtype*
  Die einzelnen Positionen haben dieselbe Bedeutung wie bei `status_a` (siehe Tabelle 4.2). *completed_time* ist 0 oder enthält die Zeit, zu der der Job beendet wurde als UNIX-Timestamp. Die Job-ID wird hier nicht extra übermittelt.
- `e:invalid jobid`
  Es existiert kein Job mit der angegebenen Job-ID.

- Auflistung aller Jobs in der Queue

  Aufruf:

  `listjobs`
  Liefert alle Jobs zurück, die sich zur Zeit in der Queue befinden.

  Rückgabe:

  `o:(`*current_job*`)` *job1 ... jobn numjobs*
  Die Job-ID des Jobs, der gerade gebrannt wird steht in *current_job*. Danach folgt eine Auflistung aller anderen Job-IDs. Dabei hat *job** folgendes Format[7]:
  jobid, pos, f_locked, f_prepared, estimated_time, percent, f_status, start_time, queue_time, jobtype

  jobid ist die Job-ID, die anderen Positionen haben dieselbe Bedeutung wie bei `status_a` (siehe Tabelle 4.2). Die einzelnen Jobs werden in derselben Reihenfolge aufgelistet, wie sie abgearbeitet werden. Sollte gerade kein Job bearbeitet werden, fällt `(`*current_job*`)` weg. Das letzte Element der Liste ist die Anzahl der gequeueten Jobs. Alle Listeneinträge sind durch Leerzeichen getrennt.

  numjobs enthält die Anzahl der Jobs in der Queue. Alle Listeneinträge sind durch Leerzeichen getrennt.

- Auflistung aller abgearbeiteten Jobs

  Aufruf:

  `listcompletedjobs`
  Liefert eine Liste aller abgearbeiteten Jobs zurück, über die mit `status_i` noch Statusabfragen vorgenommen werden können.

  Rückgabe:

  `o:`*job1 ... jobn numjobs*
  Dabei hat *job** folgendes Format:
  jobid,type,completed_time
  Die einzelnen Jobs werden in derselben Reihenfolge aufgelistet, in der sie abgearbeitet wurden. Das heißt, der zuletzt abgearbeitete Job befindet sich am Ende der Auflistung. Alle Listeneinträge sind durch Leerzeichen getrennt.

  numjobs enthält die Anzahl der Jobs.

- Auflistung aller Accounts, die einen Job in der Queue haben

  Aufruf:

  `listqacc` *admin_passwort*
  Dieser Befehl entspricht `listjobs`. Hier werden statt den Job-IDs die Accounts zurückgeliefert. Das Administrator-Paßwort der WebGUI muß im Klartext übermittelt werden.

  Rückgabe:

[7] Die Positionen sind nur durch Kommata getrennt, die Leerzeichen dienen der besseren Übersicht

- `o:(`*current_acc*`)` *acc1 ... accn numaccs*
  Der Accountname des Jobs, der gerade gebrannt wird steht in *current_acc*. Danach folgt eine Auflistung aller anderen Accounts, von denen ein Job in die Queue gestellt wurde. Dabei hat *acc** folgendes Format[8]:
  account_name, pos, f_locked, f_prepared, estimated_time, percent, f_status, start_time, queue_time, jobtype

  account_name ist der Name des Accounts, die anderen Positionen haben dieselbe Bedeutung wie bei `status_a` (siehe Tabelle 4.2). Die einzelnen Jobs werden in derselben Reihenfolge aufgelistet, wie sie abgearbeitet werden. Sollte gerade kein Job bearbeitet werden, fällt `(`*current_acc*`)` weg. Das letzte Element der Liste ist die Anzahl der Accounts. Alle Listeneinträge sind durch Leerzeichen getrennt.
- `e:illegal password`
  Das angegebene Administrator-Paßwort ist ungültig.

• Auflistung aller dem System bekannten Accounts

  Aufruf:

  `listacc` *admin_passwort*
  Liefert alle dem System bekannten Accounts und deren Anzahl zurück. Das Administrator-Paßwort der WebGUI muß im Klartext übermittelt werden.

  Rückgabe:

  - `o:`*acc1 ...* `accn numaccs` Die Accounts werden als durch Leerzeichen getrennte Liste übermittelt. Das letzte Element der Liste ist die Anzahl der Accounts.
  - `e:illegal password`
    Das angegebene Administrator-Paßwort ist ungültig.

Datei-Befehle

Diese Befehle sind eigentlich nur für die WebGUI interessant, da der Webserver keine Zugriffsrechte auf die Home-Verzeichnisse der Benutzer hat.

Falls von dem angegebenen Account bereits ein Job in die Queue gestellt wurde, liefern die Befehle
`e:your job is currently queued`
zurück. Solange ein Job von einem Account in der Queue ist, dürfen keine Dateioperationen stattfinden.

• Dateien im Home-Verzeichnis des Benutzers erzeugen

  Aufruf:

  `create` *account filename content*
  Es wird die Datei *filename* im Home-Verzeichnis des Benutzers *account* angelegt. Der Inhalt der Datei (*content*) wird einfach als weiterer Parameter übergeben. Es ist jedoch nur möglich Dateien anzulegen, wenn *account* noch nicht gequeued ist.

  Rückgabe:

  - `o:`
    Die Datei wurde erstellt.
  - `e:can't open file for writing`
    Die angegebene Datei konnte nicht zum Schreiben geöffnet werden.

[8]Die Positionen sind nur durch Kommata getrennt, die Leerzeichen dienen der besseren Übersicht

- Es können Account-Locking-Fehler auftreten (siehe weiter oben).

- Datei verschieben

  Aufruf:

  `filemv` *`account source target`*
  Verschiebt die Datei *`source`* zu *`target`*. Die Dateinamen sind relativ zum Home-Verzeichnis des Nutzers *`account`*.

  Rückgabe:

  - `o:`
    Die Datei wurde umbenannt.
  - `e:target already exists`
    Es existiert bereits eine Datei oder ein Verzeichnis mit dem Namen *`target`*.
  - `e:`*`error`*
    Das Umbenennen ist aus dem Grund *`error`* fehlgeschlagen.
  - Es können Account-Locking-Fehler auftreten (siehe weiter oben).

- Datei kopieren

  Aufruf:

  `filecp` *`account source target`*
  Kopiert die Datei *`source`* zu *`target`*. Die Dateinamen sind relativ zum Home-Verzeichnis des Nutzers *`account`*.

  Rückgabe:

  - `o:`
    Die Datei wurde erfolgreich kopiert.
  - `e:error opening source:` *`error`*
    Das Öffnen der Quelldatei *`source`* ist aus dem Grund *`error`* fehlgeschlagen.
  - `e:error opening target:` *`error`*
    Das Öffnen der Zieldatei *`target`* ist aus dem Grund *`error`* fehlgeschlagen.
  - Es können Account-Locking-Fehler auftreten (siehe weiter oben).

Sonstige Befehle

- Versionsabfrage

  Aufruf:

  `version`
  Abfrage der Version des `netfired`.

  Rückgabe:

  `o:`*`version`*
  Liefert die Version *`version`* des `netfired` zurück.

4.3.1.3 Scripthooks des `netfired`

Der `netfired` ruft zu bestimmten Ereignissen Scripthooks auf, die dann die weitere Abarbeitung einer Aufgabe definieren. Für jeden Scripthook muß in der Konfigurationsdatei des `netfired` ein Skript eingetragen werden. Im folgenden werden die Schnittstellen zu den einzelnen Skripten aus Sicht des `netfired` beschrieben.

Rückgabewerte der Skripte

Alle Ausgaben der Skripte werden vom `netfired` ausgewertet. Sie müssen in der Notation `?:` erfolgen, wobei `?` für einen Buchstaben steht, alle anderen Ausgaben können vom `netfired` als Fehler interpretiert werden. Bevor das Skript beendet wird, muß in jedem Fall eine ok-Meldung (beginnend mit `o:`) oder eine Fehlermeldung (beginnend mit `e:`) ausgegeben werden. Zusätzlich liefern alle Skripte als Rückgabe einen Exitcode entsprechend dem Erfolg der Abarbeitung, ein Exitcode von 0 heißt erfolgreich. Alle anderen Exitcodes bedeuten einen Skriptspezifischen Fehler. Dieser Exitcode wird allerdings nicht vom `netfired` ausgewertet.

Scripthooks

Im folgenden werden die einzelnen Scripthooks erläutert. Hier sind nur die Forderungen an die Skripte beschrieben, die der `netfired` an sie stellt. Die konkreten Skripte, die an die Scripthooks gebunden sind, werden in Kapitel 4.3.2.2 beschrieben. In der Aufrufsyntax steht der Name des Scripthooks (`sh_...`) für das eigentliche Skript.

- `sh_nfinit`
  Wird bei Start des `netfired` aufgerufen.

- `sh_nfexit`
  Wird vor Beenden des `netfired` aufgerufen.

- `sh_nfcheck` *`userhome`*
  Dieses Skript überprüft, ob die Daten des Nutzers in *`userhome`* als Job gequeued werden können. Falls die Daten in Ordnung sind, wird `o:`*`estd_time`* zurückgegeben. *`estd_time`* ist die geschätzte Bearbeitungsdauer des Jobs. Im Fehlerfall werden eine oder mehrere Zeilen, die mit `e:` beginnen, zurückgegeben. Diese Zeilen beinhalten die Fehlerursache.

- `sh_nfprepare` *`userhome tmpdir exclusive jobid`*
  Dieses Skript bereitet den Job im Home-Verzeichnis *`homedir`* vor. Sämtliche Daten sollen nachher in *`tmpdir`* liegen.

  `exclusive` gibt an, ob dem Skript der CD-Brenner exklusiv zur Verfügung steht. Hierbei bedeutet `1` exklusiven Zugriff, bei `0` kann der CD-Brenner von anderen Jobs benutzt werden. Um eine CD-Kopie vorzubereiten, muß die Quell-CD eingelesen werden. Dies kann nur geschehen, wenn gerade keine CD eines anderen Jobs gebrannt wird.

  `jobid` enthält die Job-ID des Jobs, der vorbereitet werden soll. Dieses Skript besitzt neben den typischen Antwortcodes `o:` und `e:` einen zusätzlichen Code `r:`. Dieser wird zurückgeliefert, falls die Vorbereitung fehlgeschlagen ist, aber nochmal wiederholt werden soll. Zum Beispiel wird `r:` zurückgeliefert, wenn bei der Vorbereitung einer weiteren Session für eine Multisession-CD der CD-Brenner nicht exklusiv zur Verfügung stand.

- `sh_nfrun` *`userhome tmpdir jobid`*
  Dieses Skript sorgt für die Abarbeitung des Jobs im Home-Verzeichnis *`homedir`*. Bei nicht erfolgreicher Ausführung gibt das Skript einen Fehlergrund zurück. *`tmpdir`* ist dabei das Verzeichnis, in dem `sh_nfprepare` den Job vorbereitet hat. *`jobid`* enthält die Job-ID des Jobs, der vorbereitet werden soll.

- `sh_nfclean` *`userhome tmpdir account`*
  Dieses Skript löscht alle nicht mehr benötigten Daten. Die Verzeichnisse *`userhome`* und *`tmpdir`* werden gelöscht. Der Account *`tmpdir`* wird aus der Nutzerdatenbank entfernt.

- `sh_nfempty`
  Dieses Skript wird aufgerufen, sobald der letzte Job abgearbeitet wurde und die Queue

somit leer ist. Ist die Queue beim Starten des `netfired` leer, wird dieses Skript allerdings nicht aufgerufen.

- `sh_nfperiodic`
  Dieses Skript wird in bestimmten Zeitintervallen aufgerufen. Es könnte z.B. dafür sorgen, daß seit längerem ungenutzte Accounts auf dem System gelöscht werden.

4.3.1.4 `netfired`-Konfigurationsdatei

Der `netfired` wird über die Konfigurationsdatei `/etc/netfired.conf` konfiguriert. Diese Datei hat den INI-Style (siehe B.1). Im Folgenden sind die Schlüssel aufgelistet:

- `user=`*UserID*
  User-ID unter der `netfired` läuft. Möglich sind die numerische User-ID, oder ein in `/etc/passwd` eingetragener Benutzer.
- `group=`*GroupID*
  Group-ID unter der `netfired` läuft. Möglich sind die numerische Group-ID, oder eine in `/etc/group` eingetragene Gruppe.
- `socket=`*filename*
  Dateiname des UNIX-Domain-Sockets, über den `netfired` gesteuert wird.
- `queue=`*filename*
  Dateiname der Textdatei, in der die aktuelle Queue abgespeichert wird.
- `accountmaker=`*filename*
  Executable des `accountmaker`.
- `maxaccounts=`*number*
  Limit, wieviele Accounts maximal angelegt werden.
- `basedir=`*Directory*
  Gibt das Verzeichnis an, unterhalb dem die Home-Verzeichnisse der Benutzer angelegt werden.
- `pwdfile=`*filename*
  Paßwortdatenbank des FTP-Servers `proftpd`.
- `grpfile=`*filename*
  Gruppendatenbank des FTP-Servers `proftpd`.
- `accountdeleter=`*filename*
  Skript, das Accounts und deren Home-Verzeichnise löscht.
- `logfile=`*filename*
  Logdatei für `netfired`.
- `lockfile=`*filename*
  Lockfile, das mehrere parallele Instanzen des `netfired` verhindert.
- `htpasswd=`*filename*
  Paßwortdatei für die Administrator-Seiten der WebGUI.
- `sh_nfinit=`*filename*
  Skript, das bei dem Scripthook `sh_nfinit` ausgeführt wird.

- `sh_nfexit`=*filename*
  Skript, das bei dem Scripthook `sh_nfexit` ausgeführt wird.
- `sh_nfcheck`=*filename*
  Skript, das bei dem Scripthook `sh_nfcheck` ausgeführt wird.
- `sh_nfprepare`=*filename*
  Skript, das bei dem Scripthook `sh_nfprepare` ausgeführt wird.
- `sh_nfrun`=*filename*
  Skript, das bei dem Scripthook `sh_nfrun` ausgeführt wird.
- `sh_nfclean`=*filename*
  Skript, das bei dem Scripthook `sh_nfclean` ausgeführt wird.
- `sh_nfempty`=*filename*
  Skript, das bei dem Scripthook `sh_nfempty` ausgeführt wird.
- `sh_nfperiodic`=*filename*
  Skript, das bei dem Scripthook `sh_nfperiodic` ausgeführt wird.
- `nfperiodic`=*number*
  Zeit in Sekunden, die zwischen zwei `sh_nfperiodic`-Aufrufen liegen soll.

Es müssen alle Schlüssel in der Konfigurationsdatei vorhanden und ihnen ein Wert zugewiesen sein. Ansonsten bricht `netfired` mit einer Fehlermeldung ab.

4.3.2 Shellskripte

Um den `netfired` so modular und einfach wartbar wie möglich zu gestalten, wurde die eigentliche Funktionalitt zum Brennen einer CD und zum Vorbereiten der Daten in Shell-Skripte ausgelagert. Diesen Skripten kommt eine Vielzahl an Aufgaben zu:

- Ansteuerung des LCDs
- Ansteuerung des Brenners durch `cdrecord` und `cdrdao`
- Vorbereiten der Sitzung durch diverse Tools (`mkisofs`, `mpg321`, ...)
- Interaktion mit dem Benutzer (Einlegen der CD)

Es werden nur die Skripte aufgerufen, deren Dateiname mit "nf" beginnt. Alle anderen Skripte sind Unterskripte und werden von anderen Sktipten benötgt. Als interpretierendes Programm wird die Standardshell `/bin/sh` (`*.sh`) oder ein Perl-Interpreter `/usr/bin/perl` (`*.pl`) verwendet und benötigt.

4.3.2.1 Benutzer-Management

- `accountmaker`
  `accountmaker` ist ein Programm, das neue Benutzer anlegt. Diese werden in die Benutzerdatenbank des FTP-Servers eingetragen. Weiterhin wird für den neu erzeugten Benutzer ein Homeverzeichnis erzeugt. Tabelle 4.3 zeigt die Kommandozeilenparameter. Konnte der neue Benutzer angelegt werden gibt `accountmaker` die Zeile `o:`*`Account`* auf der Standardausgabe aus. Sollte es zu einem Fehler gekommen sein, wird `e:`*`Fehlermeldung`* ausgegeben.

Parameter		Beschreibung
`-p`, `--passwordfile` *file*	req.	Benutzerdatenbank des FTP-Servers
`-g`, `--groupfile` *file*	req.	Gruppendatenbank des FTP-Servers
`-b`, `--basedir` *directory*		Verzeichnis, in dem die Home-Verzeichnisse angelegt werden default: aktuelles Verzeichnis
`-G`, `--group` *group_id*	req.	Gruppen ID des zu erzeugenden Benutzers
`-P`, `--prefix` *string*		Präfix für den Benutzernamen default: `NF`
`-h`, `--help`		Hilfe
`-v`, `--version`		Version

Tabelle 4.3: Parameter für `accountmaker`

- `accountdeleter.sh`
  `accountdeleter.sh` ist ein Skript, das einen Benutzer aus der Benutzerdatenbank des FTP-Servers entfernt, und sein Home-Verzeichnis löscht. Als Parameter wird nur der Benutzername übergeben, der mit dem Home-Verzeichnis identisch ist. Das Skript erzeugt

Parameter		Beschreibung
account	req.	Benutzer, der gelöscht werden soll

Tabelle 4.4: Parameter für `accountdeleter.sh`

keine Ausgabe, die ausgewertet werden muß.

4.3.2.2 Skripte für den `netfired`

Der `netfired` ist auf folgende Skripte angewiesen, um einen Brennauftrag abzuarbeiten:

- `nfinit.sh`
  Dieses Skript teilt lediglich dem `lcdfired` mit, daß der `netfired` läuft.

- `nfcheck.sh`
  `nfcheck.sh` überprüft, ob das `jobcontrol` in Ordnung ist. Falls keine Fehler gefunden wurden, wird die erwartete Bearbeitungszeit berechnet und zurückgegeben.

- `nfprepare.sh`
  Im nächsten Schritt ruft der `netfired` `nfprepare.sh` auf. Dieses Skript prüft den Datentyp und verzweigt – je nach Datentyp – in die Unterskripte `audio.sh`, `dir.sh`, `iso.sh`, `zip.pl` oder `readraw.sh`. Dieses Skripte haben die Aufgabe, in einem temporär erzeugten Verzeichnis entweder ein fertiges ISO-Image zu erzeugen (`dir.sh`, `iso.sh` und `zip.pl`) oder fertige Audiotracks als `WAV`-Datei (`audio.sh`) oder ein Diskimage für `cdrdao` (`readraw.sh`). Das Skript bricht ggf. ab, wenn ungültige oder fehlerhafte Daten vorhanden sind (z.B. ein beschädigtes Archiv).

- `nfrun.sh`
  `nfrun.sh` brennt die von `nfprepare.sh` vorbereiteten Daten. Zunächst wird auf das Einlegen eines geeigneten Rohlings gewartet, der danach ggf. gelöscht wird. Dann wird je nach Datentyp in die Skripte `recdata.sh`, `recaudio.sh` oder `writeraw.sh` verzweigt, um die vorbereiteten Daten zu brennen. Dieser Vorgang läuft in einer Schleife ab, um das Brennen mehrerer Exemplare zu realisieren.

- `nfclean.sh`
  Dieses Skript sorgt dafür, daß die nach der Abarbeitung eines Jobs nicht mehr benötigten Daten gelöscht werden.

- `nfexit.sh`
  Dieses Skript teilt lediglich dem `lcdfired` mit, daß der `netfired` nicht mehr läuft.

4.3.3 FTP-Server Erweiterung

Mit dem FTP-Server sollte es, ebenso wie über die Weboberfläche, möglich sein, erst die zu brennenden Daten und dann eine sogenannte Job-Control-Datei, die die Parameter für den Brennvorgang enthält, zu transferieren. Zusätzliche Schritte sollten dann nicht mehr nötig sein, etwa um den Vorgang noch in die Queue einzureihen. Da es sich um ein System ohne feste Nutzerverwaltung handeln sollte, mußte auch darauf geachtet werden, daß automatisch ein temporärer Account angelegt wird, unter dem der Nutzer dann seine Daten ablegen kann, und mit dem es ihm möglich ist, sich im weiteren Verlauf nach dem Status zu erkundigen oder seinen Eintrag wieder aus der Queue zu entfernen.

Dazu wurde der FTP-Server über das Modul `mod_makeacct.c` erweitert. Die genaue Arbeitsweise der einzelnen darin enthaltenen Funktionen wird im folgenden beschrieben.

- `pre_cmd_user`
  Die Funktion `pre_cmd_user` wird vor der Ausführung des ProFTPd-internen USER-Commands abgearbeitet. Die Funktion USER wird immer dann aufgerufen, wenn sich ein Client einloggt. Das Modul vergleicht dann, ob sich ein User unter dem Namen 'new' anmelden will. Wenn dem so ist, wird der Befehl `new` an den `netfired` weitergeleitet. Dies geschieht über einen Unix-Socket, dessen Pfad in der Konfigurationsdatei des ProFTPd als neuer Eintrag hinzugefügt wurde, und der vom Modul zur Laufzeit ausgelesen wird. Der `netfired` legt daraufhin einen Account an und liefert den neu generierten Usernamen an den FTP-Server über den Socket zurück. Der FTP-Server ersetzt den vom Client gelieferten Usernamen 'new' durch den neuen Namen und gibt das USER-Command normal an den ProFTPd zur weiteren Abarbeitung zurück. Dort wird daraufhin dann der Login unter dem neuen temporären Usernamen vollzogen.

  Dieses Vorgehen hat den Vorteil, daß es deutlich komfortabler für den Anwender ist, als sich erst einen temporären Account anlegen zu lassen und sich dann mit dem doch recht kryptischen Namen nochmal manuell in den FTP-Server einzuloggen.

- `pre_cmd_stor`
  Damit der `netfired` zu jeder Zeit sicher sein kann, welche Dateien des Anwenders Nutzdaten sind, die auf CD gebrannt werden sollen und welche Daten die Informationen für den Brennvorgang enthalten, wird beim Upload kontrolliert, daß nur die Job-Control-Datei namens `jobcontrol` und eine `description.txt` in das Homeverzeichnis des Users gelegt werden dürfen. Die Datei `description.txt` wird dazu genutzt, um Daten, die nach dem Brennen ins Archiv kommen, mit einem kurzen Kommentar zu versehen, um sie so auch später noch eindeutig zuordnen zu können.

  Alle anderen Dateien müssen in ein entsprechendes Unterverzeichnis gelegt werden. Dazu wird in der Funktion `pre_cmd_stor` der Dateiname und Pfad der zu speichernden Datei daraufhin überprüft und dem Anwender eventuell mit einer Fehlermeldung mitgeteilt, daß im Homeverzeichnis selbst keine zu brennenden Daten liegen dürfen. Der Transfer selbst wird dann als abgearbeitet markiert, womit eine Weitergabe an die interne STOR-Funktion des ProFTPd entfällt.

- `post_cmd_stor`
  Damit nach einem Transfer der zu brennenden Dateien auf den FTP-Server der Queue-

und Brennvorgang automatisch gestartet werden kann, sobald der Benutzer seine Job-Control-Datei übertragen hat, gibt es die Funktion `post_cmd_stor`. Hier wird überprüft, ob gerade eine Job-Control-Datei übertragen wurde. Wenn dies der Fall war, so wird dem `netfired` der Befehl gegeben, diesen User in die Queue einzureihen. Dies geschieht wieder über den Unix-Socket mit dem Befehl `queue username`.

Da syntaktische oder logische Fehler in der Job-Control-Datei nicht erst nach einer längeren Wartezeit festgestellt werden sollen, wenn der Queueeintrag an der Reihe ist und der Brennvorgang ausgeführt werden soll, wird das Parsen der Datei vom Daemon vor der Einstellung des Auftrags in die Queue durchgeführt. Bei einer korrekten Job-Control-Datei wird dem Anwender der Upload der Job-Control-Datei normal bestätigt. Bei einem aufgetretenen Fehler leitet der FTP-Server die genaue Fehlerursache, die ihm der `netfired` übermittelt hat, an den FTP-Client weiter.

Wie schon im Kapitel 2.1.2 erwähnt, wurden dafür zwei Mechanismen kombiniert, um sicherstellen zu können, daß der Anwender auch in der Lage ist, in seinem FTP-Client den Fehlergrund zu sehen. Der Server schickt einerseits bei einer fehlerhaften Job-Control-Datei eine negative Bestätigung an den Client, die als Text den Grund des Parsingfehlers enthält. Da viele FTP-Clients den Text des FTP-Servers aber ignorieren und statt dessen eigene Fehlermeldungen für einen fehlgeschlagenen Upload anzeigen, wird zusätzlich noch eine Datei namens `.message` generiert. Sie enthält den gleichen Text, den der Anwender auch gesehen hätte, wenn sein FTP-Client ihm die Fehlermeldung des Servers unverfälscht gezeigt hätte, und wird vom ProFTPd bei einem Wechsel in das Home-Verzeichnis angezeigt. Für diese Meldungen gibt es in den FTP-Clients keine vordefinierten Texte, da es sich nicht um Fehler oder Bestätigungen sondern eben um Informationen von seiten des Servers handelt. Diese werden daher auch von den Clients dem Benutzer nicht vorenthalten oder verfälscht.

- `post_cmd_dele`
  Ist der Benutzer nach dem Upload einer gültigen Job-Control-Datei in der Queue, so kann er bei Bedarf auch seine Queueposition aufgeben und sich ganz aus der Queue entfernen lassen. Dies ist z.B. dann sinnvoll, wenn kurzfristig noch Änderungen an den hochgeladenen Daten vorgenommen werden sollen. Sobald der Anwender seine Job-Control-Datei löscht, wird vom FTP-Server in der Funktion `post_cmd_dele` ein `dequeue` an den `netfired` geschickt.

- `pre_cmd_lock und post_cmd_unlock`
  Damit solche Vorgänge wie das Ändern der hochgeladenen Daten vor dem Beginn des Brennvorganges nur solange möglich sind, wie der Brennvorgang bzw. die Vorbereitung der Daten für den Brennvorgang noch nicht begonnen hat, wird bei kritischen Operationen eine Lock-Datei angelegt, die dem `netfired` signalisiert, daß der Account noch nicht zur Abarbeitung bereit ist. Kritische Operationen sind dabei alle schreibenden Befehle. Diese sind in der Funktion `pre_cmd_lock` festgelegt als `STOR, STOU, APPE, REST, RNFR, RNTO, DELE, RMD, XRMD, MKD, XMKD` (siehe dazu [20]). Dabei wurde natürlich darauf geachtet, daß die Funktion `pre_cmd_lock` als erste vor allen anderen `pre_cmd`-Funktionen und die Funktion `post_cmd_unlock` als letzte nach allen anderen `post_cmd`-Funktionen aufgerufen wird.

- `pre_cmd_retr`
  Um auch über das FTP-Interface seinen aktuellen Status bezüglich der Postition in der Queue abfragen zu können, wurde ein Status-File benutzt. Die Datei mit Namen `status`, die sich im Home eines jeden Accounts befindet, enthält die vom `netfired` gelieferten Statusinformationen. Sie wird immer mit einer Größe von Null Bytes im Directory-Listing

des FTP-Clients angezeigt, was daran liegt, daß die Datei selbst nur einen Dummy darstellt. Die eigentlichen Statusinformationen werden erst beim Download der Datei über die Funktion `pre_cmd_retr` vom `netfired` angefordert und dann direkt gesendet. Damit ist der zurückgelieferte Status immer aktuell.

- `pre_cmd_dele`
  Das Löschen der Status-Datei ist dabei nicht erlaubt und wird in der Funktion `pre_cmd_dele` abgefangen. Grundsätzlich ist es für die Funktionsweise egal, ob diese Datei existiert oder nicht. Da aber einige FTP-Clients das Directory-Listing cachen, wäre es denkbar, daß sie gar nicht erst bereit sind, eine Download-Anfrage an den Server zu senden, wenn ihnen die Existens der Datei bei einem vorangegangenen Directory-Listing verborgen geblieben ist.

Wie im Kapitel 3.1.5 schon genannt, ließen sich einige wichtige Parameter durch die Ausnutzung der schon im Server vorhandenen Funktionen einstellen. So wird mit der Konfigurationsdirektive *DirFakeGroup* und *DirFakeUser* verhindert, daß andere Benutzer auf dem FTP-Server die Namen der Besitzer von Dateien oder Verzeichnissen anderer Benutzer sehen können. Dies ist vor allem deswegen wichtig, da die einzelnen temporären Accounts kein Passwort haben. Kennt jemand also den zufällig aber eindeutig generierten Namen eines anderen Benutzers, so könnte er sich ohne Probleme unter diesem Account einloggen.

Weiterhin wird die Sicherheit der abgelegten Daten der Benutzer durch die Konfigurationsdirektiven *HideNoAccess* und *IgnoreHidden* gesichert. Diese Optionen veranlassen den FTP-Server, Dateien oder Verzeichnisse, auf die man keine Zugriffsrechte hat, gar nicht erst anzuzeigen.

4.4 Software-Entwicklungen zur User-Interaktion

Das NetFire-System soll es dem Benutzer ermöglichen, bequem und ohne Spezialsoftware das System zu bedienen. Weiterhin soll das System platzsparend und kostengünstig sein, weswegen auf eine Bedienung per Tastatur und Monitor verzichtet wurde. So entstand die Idee, eine HTML-Oberfläche zu gestalten, welche Benutzung und Administration des Systems zuläßt. Zur Ausgabe von einfachen Meldungen und zur Grundkonfiguration soll ein kostengünstiges LC-Display mit zwei Tastern dienen.

4.4.1 WebGUI

4.4.1.1 Motivation und Ziele

Im Rahmen unseres Projektes wird eine WebGUI bereitgestellt. Diese ermöglicht dem Benutzer einen komfortablen Zugriff auf das Gesamtsystem.

Die Aufgabe besteht nun darin, die Steuerung des Brennvorgangs auf mehreren HTML-Seiten darzustellen. Des weiteren soll dem Nutzer eine Unterstützung bei der Anfrage an das Brennsystem ermöglicht werden.

4.4.1.2 Qualitätsanforderungen

Bei der Erstellung der GUI muß sichergestellt werden, daß nur Funktionalitäten ausgewählt werden können, die auch im aktuellen Kontext verfügbar sind. Fehlangaben, die zum Abstürzen führen, könnten so vermieden werden. Die Software sollte ein gewisses Maß an Robustheit und Stabilität mit sich bringen. Ein wichtiges Kriterium bei der Erstellung der GUI ist Benutzbarkeit. Für den Benutzer soll das Konzept der graphischen bzw. textuellen Darstellung schnell durchschaubar sein. Mögliche Funktionen sind durch eindeutige Menüpunkte und Beschriftungen zu erklären. Für den Anwender soll der Lern-Aufwand möglichst gering gehalten werden.

4.4.1.3 Entwurf und Grobstruktur

In der ersten Entwicklungsphase wurde die Grobstruktur der Seiten auf Papier skizziert. Um den Anforderungen gerecht zu werden, wurden die einzelnen Seitenkomponenten nach den oben genannten Kriterien erstellt und platziert. In dieser Phase mußten bereits die ersten Abläufe der Benutzerführung bedacht werden. Darüberhinaus mußte zu diesem Zeitpunkt grob festgelegt werden, welche Möglichkeiten dem Benutzer zur Verfügung gestellt werden sollten. In der momentanen Version sind folgende Möglichkeiten enthalten:

- Upload von eigenen Daten, die auf einen CD-Rohling gebrannt werden sollen, dazu zählen:
  - Upload von einem eigenen ISO-Image
  - Upload von Daten, aus denen ein Image erstellt wird
  - Upload von komprimierten Daten, die noch entpackt werden müssen, bevor das Image erstellt wird
  - Upload von Audio-Dateien in einem .wav- oder .mp3-Format
- Auswahl eines ISO-Image aus einem Archiv
- Erstellung einer CD-Kopie

Der grobe Ablauf besteht aus drei Schritten:

1. Auswahl einer der oben erwähnten Möglichkeiten
2. Upload der Daten bzw. Wahl eines ISO-Image im Archiv oder Einlegen einer zu kopierenden CD, abhängig vom letzten Punkt
3. Festlegen der Einstellungen, die den Brennvorgang betreffen

4.4.1.4 HTML-Seiten

Nach dem Abschluß der ersten Entwicklungsphase, in der die Papier-Prototypen und die Ablauf- bzw. Sequenzdiagramme des Brennvorgangs erstellt wurden, sind sukzessive die eigentlichen HTML-Seiten implementiert worden. Unter Berücksichtigung der Sequenzdiagramme wurde die logische Reihenfolge und Verknüpfung der einzelnen Seiten festgelegt und im Zusammenspiel mit PHP vervollständigt. In dieser Arbeitsphase wird zuerst auf die Funktionalität geachtet und das Design nur eigeschränkt überdacht und verwirklicht. In dieser Hinsicht sind aber schon grundlegende Entscheidungen getroffen worden. Die Web-Seiten sollen vor allem dem Benutzer ein einheitliches Erscheinungsbild vermitteln und ein gewisses Maß an Übersichtlichkeit bieten.

Das Grundgerüst der Seiten, das im Verlauf der Interaktivität mit dem User beibehalten wird, wird mit Hilfe von insgesamt vier Frames, die voneinander mit unsichbaren Rändern abgegrenzt sind, implementiert (siehe Abbildung 2.1). In dem oberen Frame auf einem schwarzen Hintergrund ist ein Schriftzug unserer Projektgruppe sichtbar. Der untere Frame ist in einem gleichen Stil wie der obere Frame und der gleichen Größe fertiggestellt, um eine gewisse Symmetrie und Kontrast zu dem mittleren Teil der Seiten, der in weiß gehalten wird, zu erreichen. Der linke vertikale Frame beinhaltet das Logo des Projektes sowie drei Links, die im Kapitel 2.1.1 näher beschrieben sind. Der wichtigste Bereich einer Seite befindet sich in dem mittleren Frame, der die meiste Fläche des Entwurfs in Anspruch nimmt. Hier werden die benötigten Abfragen und Einstellungen, die zu einem Brennvorgang nötig sind, vorgenommen. Alle weiteren HTML-Seiten, aufbauend auf der Willkommen-Seite, werden an diese Stelle von dem Server geladen. In diesem Zusammenhang sind die CGI-Skripte zu erwähnen, auf die noch weiter unten genauer eingegangen wird.

Das Herzstück des HTML-Codes bilden die Formularfelder, die die notwendige Interaktion zwischen Client und Server überhaupt möglich machen. Ein gutes Beispiel für die Vielfalt und die Anwendbarkeit der Formular-Tags wird vor allem in der Brenneinstellungsseite sichtbar. Die Seite selbst ist in der Abbildung 4.15 dargestellt. Als Elemente für die Formulargestaltung werden im Rahmen der WebGUI viele Feldarten benutzt. Die Liste beginnt mit den einfachen Feldern, in denen Text eingegeben werden kann. Sie reicht über Auswahllisten, z.B. bei der Auswahl eines ISO-Images aus dem Archiv, bis hin zu Schaltern und Ankreuzfeldern. Mit Hilfe der einzelnen Komponenten lassen sich sehr schnell und einfach alle signifikanten Informationen sammeln und gegebenfalls mit Hilfe der hidden-Tags dynamisch auf die folgenden Seiten übertragen. Die somit gesammelten Werte gelangen als Stream mittels der Post-Methode, bei der die Informationen nicht sichtbar an URL angehängt werden, auf den Server. Dort werden diese weiterverarbeitet, abhängig von den Absichten des Users.

Desweiteren ist dem Benutzer die Möglichkeit gegeben worden, mit Hilfe der WebGUI einige administrative Einstellungen und Veränderungen, wie z.B. Löschen der gesamten Queue oder Änderung der IP-Adresse, zu betätigen. Dazu ist ein separater Ablauf neuer Seitenfolgen implementiert worden (siehe auch Kapitel 2.1.1).

Abbildung 4.15: Brenneinstellungen mittels WebGUI vornehmen

4.4.1.5 PHP-Skripte

Um die Interaktivität zwischen System und Benutzer überhaupt sicherzustellen, werden meistens CGI-Skripte eingesetzt [21]. Nach dem CGI-Standard, der festlegt, wie ein Client ein Programm auf einem WWW-Server starten und an dieses Informationen übergeben kann, kann ein CGI-Skript in einer beliebigen Programmiersprache geschrieben werden. Nach einer gründlichen Überlegung und weiteren Recherchen fiel die Entscheidung auf PHP [22]. PHP basiert auf der Idee von CGI. Es wird aber nicht als eigenständiges CGI-Programm ausgeführt. Diese Skriptsprache zur Erstellung dynamischer Websites wird nämlich direkt in die HTML-Seiten eingebunden. Dabei benötigt PHP keine bestimmte browserseitige Technik, kann also mit jedem Standardbrowser arbeiten. Es handelt sich bei PHP um eine sehr mächtige Skriptsprache mit einer Vielzahl von nützlichen Funktionen, die die Entwicklungsarbeit vereinfachen und beschleunigen. Auch die komfortable Unterstützung der Sockets, die zur Kommunikation mit dem `netfired` Deamon benötigt werden, war einer der Gründe, sich für PHP zu entscheiden.

Der Schlüssel zu interaktiven Webseiten sind die HTML-Formulare. Der Nutzer wird in die Lage versetzt, Daten einzugeben, und der Server gibt diese Daten an den PHP-Prozessor zur Auswertung weiter. Jedes Formularelement ist durch das Attribut `name` gekennzeichnet. Es erscheint in PHP als normale Variable mit dem Namen des Elements, gefüllt mit den eingegebenen Werten. Der Nutzer wird abhängig von den von ihm mit Formularen angegebenen Daten von Skript zu Skript weitergeleitet. Wichtige Angaben werden dabei mitweitergereicht. Dazu werden versteckte Felder innerhalb eines Formulars benutzt. Zu den wichtigen Angaben gehört beispielsweise die Nutzerkennung. Man braucht deswegen keine aufwendigeren Sessionverwaltungen einzuführen, da keine durch den Nutzer vorgenommenen Einstellungen zwischengespeichert werden müssen.

4.4.1.6 HTTP-Upload

Eine der Techniken, eine Datei auf unseren `netfired`-Server hochzuladen, ist der HTTP-Upload. Dabei wird die Datei als ein Stream dem entsprechendem Skript übergeben. Bei der Durchführung eines Tests hat sich aber herausgestellt, daß die Übertragung von großen Dateien Schwierigkeiten bereitet. Auf der Serverseite werden sehr hohe Speicherressourcen vorausgesetzt. Der Grund liegt in der Tatsache, daß das jeweilige PHP-Skript die hochgeladene Datei nicht direkt auf die Festplatte speichert, sondern versucht, diese in dem Arbeitsspeicher bzw. Swap-Bereich zwischenzuspeichern. Mit dieser Einschränkung ist auch das gleichzeitige Uploaden mehrerer Dateien von verschiedenen Nutzern nicht realisierbar, da der dazu benötigte Arbeitsspeicher immense Kosten verursachen würde. Es wurde versucht, dieses Problem zu bewältigen, indem ein CGI-Programm in C implementiert wurde, dessen Aufgabe daran bestand, die Daten vom Benutzer zu empfangen und diese direkt auf die Festplatte zu schreiben. Dies schien die Lösung des Problems zu sein, was einige Tests belegten. Es hat sich aber noch ein größeres Problem harauskristallisiert. Nach genaueren Recherchen und ausgiebigen Tests mit verschiedenen Browsern hat sich gezeigt, daß der Großteil der allgemein benutzten Browser grundsätzlich mit dem HTTP-Upload nicht zurechtkommt. Der Grund dafür ist vermutlich die Größe der Dateien, die hochgeladen werden, die bei Images bekanntlich bis zu 700 MB betragen kann.

Unter Berücksichtigung der oben genannten Probleme kann die Entscheidung nicht fern liegen, die Weiterentwicklung dieser Technik aufzugeben oder bis auf weiteres zu verschieben. Der Einsatz dieser Upload-Möglichkeit wird auf den Fall begrenzt, in dem der Nutzer eine bootfähige CD erstellen will. Durch die beschränkte Größe eines Boot-Images ist es möglich dieses via HTTP problemlos hochzuladen.

4.4.1.7 FTP-Upload

Aufgrund der oben genannten Umstände scheint die Upload-Möglichkeit per FTP die einzige, die in Frage kommt, um große ISO-Images auf den Server hochzuladen. FTP ist für diese Funktionalität ohnehin prädestiniert und bietet in dieser Hinsicht weitere Vorteile. Einer davon ist offensichtlich die Fähigkeit, ein unterbrochenes Upload an der richtigen Stelle fortzusetzen (Resume-Funktion), ohne die ganze Datei noch einmal zu übertragen. Die Unterbrechung kann leicht beispielsweise durch eine Leitungsstörung oder einen Netzwerkfehler verursacht werden. Wie in dem Kapitel 3.1.3 gezeigt ist, bietet so gut wie jeder Browser die Möglichkeit, einen FTP-Upload durchzuführen. Die konkrete Implementierung sieht folgendermaßen aus: An der entsprechenden Stelle wird der Benutzer aufgefordert, mittels eines Links ein neues Browser-Fenster zu öffnen. Dieses Fenster stellt das für ihn bestimmte Zielverzeichnis dar. Je nach Browsertyp wird ein Upload entweder per *Drag&Drop* oder per Menüpunkt in der Menüleiste

(Netscape) initialisiert. Dabei wird dem Benutzer die Größe der übertragenen Dateien auf dem Server angezeigt. Auf diese Weise kann der Nutzer kontrollieren, ob die ganze Datei auf den Server übertragen wurde, oder ob wegen einer Störung die Wiederaufnahme der Übertragung nötig ist.

4.4.1.8 Job-Control-Datei

Nachdem der Benutzer alle Eingaben, die zum Starten eines Brennvorgangs nötig sind, gemacht hat, werden diese an den `netfired` Deamon weitergegeben. Dies passiert in zwei Schritten. Zuerst wird aus den Benutzereingaben, die mit Hilfe von HTML-Formularen an PHP weitergegeben wurden und in PHP-Variablen gespeichert sind, eine Job-Control-Datei erzeugt. Das File wird ins User-Home des gerade aktiven NetFire-Benutzers geschrieben. Es beinhaltet alle Einstellungen, die für das Brennen einer CD relevant sind. Im zweiten Schritt wird der NetFire-Deamon über das Socket-Interface mit entsprechendem Befehl benachrichtigt, indem der Job in die Queue gestellt wird.

Ein Beispiel für eine Job-Control-Datei mit den dazugehörigen Erklärungen wird in der Abbildung 4.16 dargestellt. Diese Datei hat den shellkompatiblen Stil, wie er im Anhang B.2 beschrieben wird.

4.4.1.9 Wichtige Verzeichnisse und Dateien der WebGUI

Konfigurationsverzeichnis	`/etc/httpd`
Konfigurationsdatei für PHP	`/etc/php.ini`
Verzeichnis für Webseiten und PHP-Scripts	`/var/www`
Konfigurationsdatei für die WebGUI	`/etc/webgui.config`

4.4.1.10 WebGUI-Konfigurationsdatei

Die PHP-Skripte der WebGUI brauchen die Position einiger Pfade und Dateien im System. Diese werden über die Konfigurationsdatei `/etc/webgui.config` konfiguriert. Diese Datei hat den shellkompatiblen Stil, wie er im Anhang B.2 beschrieben wird. Im folgenden werden die einzelnen Schlüssel beschrieben.

- `socket`
  Dateiname des Unix-Domain-Sockets, über den Die WebGUI mit dem `netfired` kommuniziert.

- `archive_path`
  Pfad der angibt, wo das Archiv mit schon gebrannten archivierten ISOs zu finden ist.

- `netfire_config`
  Position der Konfigurationsdatei `netfire.config`, in der Netzwerkeinstellung und SCSI-Einstellungen eingetragen wird. Diese Datei ist von der WebGUI nicht schreibbar, sie dient nur zum Auslesen der Werte.

- `netfire_config_tmp`
  Position der temporären Konfigurationsdatei `netfire.config`, in der Netzwerkeinstellung und SCSI-Einstellungen eingetragen wird. Diese Datei sieht genau aus wie die `netfire.config` und ist von der WebGUI schreibbar. Diese Datei wird anschließend durch die beiden Skripte `update_config.sh` und `configurator` in die richtige Konfigurationsdatei `netfire.config` geschrieben.

```
# Type of job
# [Normal, Zip, Dir, Copy, Audio, Toc, Cue, Bin, ...]
# set "Normal" for Iso-Image
Type=Normal #string

# Preferences: Speed (max=24) and number of copies (min=0,max=3)
# if Speed=100 then maximum available Speed is used
Speed=24 #int
Copies=0 #int

# CD-RW-Deletion
# No:   don't delete
# All:  delete whole CD-RW
# Fast: delete TOC
DeleteMode=No #string

# Write single image: MultiSession=no
# Create multisession CD: MultiSession=yes
MultiSession=no #bool

# available write-modes:
# DAO/SAO: Disc At Once / Session At Once :DAOMode=Yes
# TAO: Track At Once:DAOMode=No
DAOMode=Yes #bool

# CD-Modes
# 1 / mode1:    CD-ROM Mode 1 (Yellow-Book) (2048 bytes/sector, ec included)
# 2 / mode2xa1: CD-ROM Mode 2 XA1
# 3 / mode2xa2: CD-ROM Mode 2 XA2
# 4 / Audio :   CDDA mode (Red-Book) (2352 bytes/sector, ec included)
# 5 / Mode1Raw: CD-ROM Mode 1 RAW : (2352 bytes/sector)
Mode=1 #string: also valid: Mode=Audio

# Enter the name of your boot-image here to create a bootable cd
# the boot-image must be present in the user-home folder ~/boot
# if no BootFileName is given a non-bootable CD will be created.
BootFileName=boot.img #string

# direct path and name of your iso-image
FileName=/home/archive/test.iso #string

# move iso-image to archive? "yes" or "no"
MoveToArchive=no #bool

# Insert user's email-address.
# If this option is left blank the user will only be notified via HTML-page
EMail=bla@bla.bla #string
```

Abbildung 4.16: Beispiel für eine Job-Control-Datei

- `passwd_file`
  Position der `passwd`-Datei, die in der `httpd.conf` eingetragen ist. Diese Datei enthält das Passwort für den Admin-Benutzer des Systems bzw. des Apache.

4.4.2 Display- und Tastersteuerung

Das Konzept zur Display- und Tasteransteuerung wurde durch eine Reihe aufeinander aufbauender Tools und Bibliotheken umgesetzt. Als Basis für die Ansteuerung dient das LCD Kernel-Modul (siehe 4.2.3.1). Als allgemeine Grundlage zur Implementierung eigener Software dient die `liblcd`, welche Funktionen zur Text- und Grafikausgabe bereithält.

Hierauf setzen die NetFire-spezifischen Applikationen `lcdsetup` und `lcdfired` auf. `lcdsetup` gibt dem Anwender die Möglichkeit, grundlegende Netzwerkeinstellungen über das LC-Display zu treffen, während `lcdfired` Statusmeldungen der Brennvorgänge anzeigt.

4.4.2.1 Die `liblcd` zur Ansteuerung des Displays

Konzept

Aufsetzend auf das LCD Kernel-Module wurde eine statische Bibliothek entwickelt, die eine weitergehende Ansteuerung des Displays ermöglicht. Diese Bibliothek enthält Funktionen zur gesteuerten Ausgabe von Text und Grafik, sowie zur Beeinflussung automatischer Verhaltensweisen wie Scrolling oder überdeckende Grafiken (Sprites).

Kompatibilität

Unter SuSE Linux 7.3 mit Kernel 2.4.10GB zeigten sich Probleme bei der Positionierung von Text und Grafik auf dem Display. Grund hierfür ist eine fehlerhafte Implementierung der Funktion `fileno(...)` in diesem Kernel. Diese Funktion wird jedoch von der `liblcd` benötigt, um das Filehandle des geöffneten Devicefiles herauszufinden. Abhilfe schafft hier nur die Verwendung eines anderen Kernels, z.B. Version 2.4.18.

Funktionen

Die C-Bibliothek kann statisch im eigenen Programmcode eingebunden werden, indem man die Datei `liblcd.a` als Input-File für den Linker angibt. Die Funktionsdeklarationen befinden sich in `liblcd.h`. Soweit nicht anders angegeben, liefern die Funkionen einen Wert <0, falls sie fehlgeschlagen sind.

Viele der Grafikfunktionen arbeiten mit einem internen Puffer. Wird bei solchen Funktionen das Flag *`buffered`* gesetzt, so findet die Ausgabe nicht direkt auf dem Display statt. Dies ermöglicht den Aufbau von Grafiken, ohne daß dabei das Display flackert. Alle gepufferten Grafikoperationen werden entweder durch `dump_buffer()`, oder durch die erste ungepufferte Grafikoperation sichtbar. Folgende Funktionen sind deklariert:

- `int lcd_open(char *device)`

  Öffnet die Kommunikation mit dem angegebenen *`device`* (`/dev/lcd`). Ist das Device nicht ein Devicefile zum LCD-Treiber, so wird in den Emulationsmodus geschaltet, in dem alle Displayausgaben durch ASCII-Grafik auf die Standardausgabe gegeben werden. Die angegebene Datei muß dazu jedoch als leere Datei existieren.

- `int lcd_close()`

  Schließt die Kommunikation mit dem LCD-Treiber

- `int lcd_setfont(const char *fontname)`

  Lädt ein PSF-Fontfile zur Textausgabe. Textausgaben sind erst gültig, wenn eine Schriftart geladen ist.

- `void lcd_setscollmode(lcd_scrollmode mode)`

  Beeinflußt das Scrollverhalten bei Textausgabe. Die Scrollmodes sind in Tabelle 4.5 dokumentiert.

- `int lcd_putchar(char c)`

  Gibt das Zeichen *c* an der aktuellen Cursorposition aus.

- `void lcd_dump_buffer()`

  Gibt alle gepufferten Grafikdaten auf das Display aus.

- `int lcd_clear()`

  Löscht Display und Puffer umgehend.

- `int lcd_setcurpos(int x, int y)`

  Setzt die Cursorposition für Textausgabe. x und y sind Angaben in Pixel.

- `int lcd_getcurpos(int *x, int *y)`

  Liefert die Cursorposition für Textausgaben in Pixel.

- `int lcd_printf(const char *format, ...)`

  Gibt einen Text auf das Display aus. Die Parameter entsprechen der C-Funktion `printf`.

- `int lcd_setpixel(int x, int y, char pixel)`

  Setzt das Pixel an der Position x,y, wenn der Wert *pixel* ungleich 0 ist. Ansonsten wird das Pixel gelöscht.

- `int lcd_getpixel(int x, int y)`

  Liefert den Wert (0 oder 1) des angegebenen Pixel.

- `char lcd_getkeystate(int wait)`

  Liefert den Zustand der Display-Taster, wenn `/dev/lcduser` existiert, ansonsten werden Eingaben von der Tastatur entgegengenommen (wartend). Das Ergebnis der Abfrage ist eine Zahl von 0 (kein Taster) bis 3 (beide Taster), jedoch als ASCII-Zeichen (also die Werte 0x30 bis 0x33). Ist das Flag *wait* gesetzt, wird gewartet, bis ein Taster gedrückt wurde. Im Emulationsmodus wartet die Bibliothek auf die Eingabe einer Zahl (1 bis 3), gefolgt von Return.

- `void *lcd_loadsprite(const char *fname, int x, int y)`

  Lädt eine PCX-Datei und generiert daraus Sprite-Daten mit den Koordinaten x,y. Der Rückgabewert ist ein Handle auf dieses Sprite für den weiteren Gebrauch und darf erst nach Freigabe des Sprites vernichtet werden.

- `void lcd_unloadsprite(void *sprite, int buffered)`

  Entfernt das Sprite aus dem Speicher. Danach wird *sprite* ungültig.

- `void lcd_movesprite(void *sprite, int x, int y, int buffered)`

  Setzt das Sprite *sprite* auf die Koordinaten x,y.

lcd_scrollmode	Bedeutung
lcd_none	Der Cursor bleibt an der letzten Position stehen
lcd_rewrite	Der Cursor wird in die linke, obere Ecke gesetzt, wenn er den unteren Rand überschreitet
lcd_hard	Es wird, wenn nötig, eine Zeile weiter gescrollt
lcd_soft	wie *lcd_hard*, jedoch wird pixelweise gescrollt

Tabelle 4.5: Die Scrollmodes der `liblcd`

- `void lcd_hidesprite(void *sprite, int buffered)`

  Entfernt das Sprite *sprite* vom Display, belässt es aber im Speicher.

- `void lcd_showsprite(void *sprite, int buffered)`

  Zeigt das Sprite *sprite* an.

- `void lcd_settextwindow(int x1, int y1, int x2, int y2)`

  Setzt den sichtbaren Bereich für Text. Textzeichen werden nicht außerhalb dieses Bereichs angezeigt. Somit ist es möglich, textfreie Regionen für die Grafikausgabe (z.B. Sprites) zu schaffen. Wird durch die Textausgabe der untere Rand des Textfensters überschritten, so wird ein Scrolling eingeleitet. Das Textfenster entspricht zu Anfang der gesamten Displaygröße (entsprechend `lcd_settextwindow(0,0,121,31)`).

4.4.2.2 Das `lcdsetup` - Einstellung der Netzwerkparameter

Das `lcdsetup` ist ein Tool zur Einstellung der Netzwerkkonfiguration über das LC-Display. `lcdsetup` wird nach Systemstart – zu einem Zeitpunkt, an dem das Root-Filesystem Schreibzugriffe erlaubt – von einem Initskript aufgerufen. Es gibt dann eine Meldung auf dem Display aus, daß durch Drücken beider Taster am LCD das Setup-Programm gestartet wird. Sollte dies nicht innerhalb von 5 Sekunden geschehen, fährt das System mit der Bootsequenz fort.

Sollte bei Systemstart die Datei `/etc/netfire.config` (bzw. die in der `lcdsetup`-Konfiguration angegebene Datei) noch nicht existieren, wird das Setup-Programm sofort durchlaufen.

Bedienung

Bei der Bedienung des Setup-Programms erhalten die beiden Taster je nach Situation unterschiedliche Bedeutungen. Die Funktion der Taster wird durch Graphiken, die direkt neben den Tastern angezeigt werden, verdeutlicht – z.B. zeigt ein „yes“-Symbol, daß der betreffende Taster zur Bestätigung dient.

Das Setup-Programm bietet die Möglichkeit, das DHCP-Protokoll zur automatischen Netzwerkkonfiguration auszuwählen. Ist dies erwünscht – was natürlich einen DHCP-Server im lokalen Netzwerk voraussetzt – bestätigt man dies durch Drücken des „yes“ Tasters (oben). Das Setup-Programm beendet nun und gibt die HW-Adresse (MAC) der Ethernetkarte aus. Diese wird benötigt, um Einstellungen im DHCP-Server zu treffen, wie die Vergabe einer festen IP.

Wird DHCP nicht aktiviert („no“-Taster, unten), so kann man nun die lokale IP, die Netzmaske und die Gateway-Adresse einstellen. Durch den oberen Taster kann man eine Ziffer um eins erhöhen, durch den unteren Taster bewegt sich der Auswahlcursor um eine Ziffer nach rechts. Bewegt man den Auswahlcursor über die letzte Ziffer hinaus, so erfolgt eine Bestätigungsabfrage für die jeweilige Einstellung. Bestätigt man nicht, so beginnt der Einstellvorgang der jeweiligen Option von der ersten Ziffer an. Bestätigt man durch „yes“, so wird die jeweils nächste Option eingeblendet. Nach Bestätigung der eingestellten Gateway-Adresse wird das Setup-Tool beendet.

Funktionsweise

Das `lcdsetup` verwendet `liblcd` zur Ansteuerung des Displays. Es liest zu Anfang die Konfigurationsdatei `/etc/lcdsetup.conf` . Dort wird der Dateiname der Netzwerkkonfiguration hinterlegt, üblicherweise `/etc/netfire.config`, welche ebenfalls eingelesen wird.

Dem Benutzer werden die Informationen aus dieser Datei als entsprechende Vorgaben angezeigt. Neue Einstellungen werden wieder in der Netzwerkkonfiguration gespeichert. Sollte beim Start noch keine Netzwerkkonfiguration existieren, so werden Standardeinstellungen verwendet und gespeichert. Diese kann man Tabelle 4.6 entnehmen.

`lcdsetup` liefert einen Exit-Code von 0, falls Einstellungen verändert wurden. Das aufrufende Init-Skript kann so ggf. das Konfigurationsskript `/sbin/configurator` aufrufen, welches die neugeschriebene Netzwerkkonfiguration in die Systemeinstellungen übernimmt.

`lcdsetup` unterstützt die Kommandozeilen-Option -c zur Angabe einer anderen Konfigurationsdatei.

Zur Ausgabe der MAC-Adresse wird ein externes Tool benötigt, welches die MAC-Adresse der Netzwerkkarte ausliest und auf die Standardausgabe ausgibt. Der Name dieses Tools kann in der Konfigurationsdatei angegeben werden. Dem `lcdsetup` liegt ein solches Tool mit dem Namen `/bin/macread` bei.

Konfiguration

Die Konfigurationsdatei des `lcdsetup` entspricht im allgemeinen dem INI-File Format (siehe B.1). Folgende Schlüssel/Wertepaare müssen gesetzt werden:

- `font`=*filename*
  Ein Fontfile im psf-Format als Displayfont
- `device`=*filename*
  Das LCD-Treiber Devicefile
- `macread`=*filename*
  Tool zum Auslesen der MAC-Adresse. Das Tool liefert die MAC-Adresse auf der Standardausgabe.
- `nfconfig`=*filename*
  Die NetFire Netzwerkkonfigurationsdatei
- `arr_up`=*filename*
  PCX-File mit der Sprite-Grafik „Pfeil hoch“
- `arr_r`=*filename*
  PCX-File mit der Sprite-Grafik „Pfeil rechts“
- `yes`=*filename*
  PCX-File mit der Sprite-Grafik „Yes“
- `no`=*filename*
  PCX-File mit der Sprite-Grafik „No“

```
# Beispieldatei /etc/lcdfire.conf
font="/bin/lat9-08.psf"
device="/dev/lcd"
nfconfig="/etc/netfire.config"
arr_up="/bin/arr_up.pcx"
```

```
arr_r="/bin/arr_r.pcx"
yes="/bin/yes.pcx"
no="/bin/no.pcx"
macread="/bin/macread"
```

Einstellung	Wert
HOSTIP	192.168.1.1
GATEWAY	192.168.1.99
NETMASK	255.255.255.0
NAMESERVER	0.0.0.0
DOMAIN	intern.net
USE_DHCP	1
SMARTHOST	0.0.0.0
HOSTNAME	NetFire
SCSI_ADDRESS	0,0,0

Tabelle 4.6: Default-Werte für `/etc/netfire.config`

4.4.2.3 Der `lcdfired` zur Anzeige von Statusmeldungen

Der `lcdfired` ist ein Daemon-Prozeß, der Statusmeldungen für das Display entgegennimmt und verwaltet. Er verwendet dazu eine Socket-Schnittstelle, über die ein Client Statusmeldungen an den Daemon senden kann.

Die Statusmeldungen entsprechen dem aktuellen Zustand der Brennvorgänge. Der `lcdfired` berücksichtigt, daß gleichzeitig mehrere Zustände eintreten können. Z.B. kann die Vorbereitung von CD-Daten zeitgleich mit einem laufenden Brennjob stattfinden. `lcdfired` reagiert auf solche Situationen, indem es entsprechende Statusmeldungen abwechselnd anzeigt.

`lcdfired` wird über die Datei `/etc/lcdfired.config` konfiguriert. Durch die Kommandozeilen-Option -c kann eine andere Konfigurationsdatei angegeben werden. Die Option -d startet `lcdfired` nicht als Daemon. Dies ist insbesondere für den Emulationsmodus notwendig (siehe 4.4.2.1 unter `lcd_open(...)`).

In der `lcdfired`-Konfiguration kann ein Shutdown-Befehl eingestellt werden. Dieser wird ausgeführt, wenn das Shutdown-Kommando über den Socket gesendet wird, oder wenn beide Display-Taster für mehr als 4 Sekunden gedrückt werden.

Befehle des `lcdfired`

Über den Socket werden folgende Befehle von `lcdfired` entgegengenommen:

- `status empty`
  Signalisiert dem Daemon, daß die Queue leer ist. Dieser Status sollte auch einmalig beim Initialisieren des `netfired` aufgerufen werden, um den Status *nodaemon* zu verlassen. Das Display zeigt bei Empty eine Versionsmeldung.

- `status prepare` *`jobid`*
  Signalisiert den Beginn der Jobvorbereitung für *`jobid`*

- `status endprepare`
  Signalisiert das Ende der Jobvorbereitung

- `status announce` *`cdtyp`*
  Meldet einen gewünschten Rohling-Typen.

- `status reading` *[jobid]*
  Meldet, daß ein eingelegter Rohling geprüft/gelesen wird. Ist eine *jobid* angegeben, so wird zudem noch die Job-ID und der Fortschritt des Lesevorgangs angezeigt. Diese Anzeige kann durch `status progress` beeinflußt werden.

- `status burning` *jobid*
  Zeigt, daß der Brennvorgang *jobid* läuft.

- `status progress` *vh*
  Meldet den Fortschritt eines Brenn- oder Lesevorgangs in Prozent.
  Folgende Sonderfälle sind möglich:

Wert	Bedeutung
104	flushing cache
103	CD wird gelöscht
102	schreiben lead in
101	schreiben lead out

- `status remove` *jobid*
  Meldet den Brennvorgang *jobid* als erfolgreich.

- `status error` *jobid*
  Der Brennvorgang ist fehlgeschlagen.

- `status nodaemon`
  `netfired` steht nicht zur Verfügung. Dies ist der initiale Status des `lcdfired`. Dieser Status sollte auch gesetzt werden, wenn `netfired` terminiert.

- `version`
  Liefert die Versionsnummer des `lcdfired` über den Socket (mit vorangestelltem o:)

- `shutdown`
  Ruft den Shutdown-Befehl auf.

Jeder Statusmeldung kann zusätzlich ein Text angehängt werden. Ist ein solcher Text vorhanden, so wird dieser Anstelle des Standardtextes auf dem Display ausgegeben. Der Text muß durch ein ; vom status-Kommando getrennt werden.

Beispiel: `status burning Job1;Job1 wird gebrannt.`

Weiterhin darf die Zeichenkette `\n` benutzt werden, um Zeilenumbrüche zu erzwingen - hier wird nicht das Newline-Zeichen verwendet, sondern die beiden ASCII-Zeichen `\` und `n` !

Rückmeldung

Der `lcdfired` liefert zu jedem Befehl einen Antwortstring. Dieser beginnt mit `o:` , falls der Befehl korrekt ausgeführt wurde, oder mit `e:` , falls es bei der Abarbeitung zu einem Fehler kam. Im Anschluß an dieses Kürzel kann eine freie, textuelle Rückmeldung folgen.

Client

Der lcdfire-Client (`/bin/lcdfire`) nimmt Befehle über die Kommandozeile entgegen und sendet diese an den `lcdfired`-Socket. Dazu liest es die Konfigurationsdatei des `lcdfired`. Auch beim Client kann durch die Kommandozeilenoption -c eine andere Konfigurationsdatei ausgewählt werden.

Beispiel: `lcdfire status prepare job1`

Konfiguration

Die Konfigurationsdatei des `lcdfired` entspricht im allgemeinen dem INI-File Format (siehe B.1). Folgende Schlüssel/Wertepaare müssen gesetzt werden:

- `user`=*UID*
  Der User, unter dem der Daemon laufen soll
- `group`=*GID*
  Die Gruppe, unter der der Daemon laufen soll
- `logfile`=*filename*
  Das Logfile für Daemon-Zugriffe
- `socket`=*filename*
  Der Socket für die Clientkommunikation
- `font`=*filename*
  Ein Fontfile im PSF-Format als Displayfont
- `device`=*filename*
  Das LCD-Treiber Devicefile (ein echtes File im Emulationsmodus)
- `shutdown`=*filename*
  Shutdown-Befehl

```
# Beispieldatei /etc/lcdfired.conf
USER=netfire
GROUP=netfire
LOGFILE=/var/log/lcdfired.log
SOCKET=/var/run/.lcdfired.sock
FONT=/bin/lat9-08.psf
DEVICE=/dev/lcd
SHUTDOWN="/sbin/halt"
```

Kapitel 5

Hardware

Für das NetFire-System waren neben der Software auch mehrere Hardwarekomponenten notwendig: ein Basissystem (Chassis), ein CD-Brenner sowie ein Display. Das Basissystem und der Brenner wurden dabei fertig gekauft, das Display hingegen wurde selbst entwickelt. Nachfolgend sind diese Komponenten näher beschrieben.

5.1 Die Komponenten des NetFire-Systems

Einleitung

Damit das System den Anforderungen entspricht, die die Spezifikation stellt, ist ein modernes und leistungsfähiges System notwendig. Gleichzeitig soll es preisgünstig und klein sein. Um auch moderne 24x CD-Brenner und zukünftige DVD-Brenner zu unterstützen, benötigt es außerdem ein 100Mbit/sec Ethernetinterface und eine IDE-Schnittstelle.

Um die Entwicklungszeit des Gesamtsystems möglichst kurz zu halten, wurden möglichst viele fertige Komponenten verwendet. Auf der Hardwareseite bedeutet dies, daß ein möglichst komplettes Board mit Peripherie (Ethernet, IDE-Interface) bevorzugt wurde. Auf der Softwareseite ist die Konsequenz, auf ein System zu setzen, für das möglichst viel Software frei verfügbar ist, und das von einer möglichst großen Zahl von PG-Teilnehmern beherrscht wird. Zusätzlich spielen Lizenzkosten für das verwendete System eine Rolle. Das Vorhandensein von Quellcodes ist ein Plus.

Kleine ATX PC-Systeme

Im Bereich von ATX gibt es eine Reihe von Standards, die kleine Formfaktoren ermöglichen:

Form Faktor	Maximale Breite	Maximale Tiefe
FlexATX	229mm	191mm
microATX	244mm	244mm
ATX (normal)	305mm	244mm
Mini-ATX	284mm	208mm

Tabelle 5.1: Formfaktoren der ATX Standards

Für unsere Zwecke scheint insbesondere FlexATX[1] interessant. Der Standard wurde 1999 von Intel vorgestellt und soll den Bau von kleinen low-cost PC-Systemen erlauben. Da es dieselben Einbaulöcher auf dem Mainboard vorsieht wie der ältere microATX[2] Standard, sind keine

[1] `http://www.formfactors.org/developer/specs/flexatx/flexatxspecs.htm`
[2] `http://www.formfactors.org/developer/specs/microatx/microatx.htm`

neuen Gehäuse nötig. Ein ATX oder SFX[3] Netzteil wird benötigt, um die Versorgungspannungen zur Verfügung zu stellen.

Verfügbare FlexATX Systeme

FlexATX Systeme gab es von der Firma Spacewalker[4]. Vertrieben wurden sie z.B. von Reichelt Elektronik[5]. Ein passendes Gehäuse „FLEX ATX GEHÄUSE“ war für 159DM verfügbar. An Mainboards war das FV24[6] mit Sockel370 und VIA PL133 Chipsatz (mit integriertem Ethernet, IDE, VGA, Sound) zum Preis von 269DM verfügbar[7].

Zusätzlich wurden eine CPU, Speicher und ein Bootmedium benötigt. Da es sich um ein Standard PC-Mainboard handelt, konnte jeder Socket370 kompatible Prozessor (z.B. Intel Celeron) verwendet werden. Als Speicher wurde handelsüblicher PC100 oder PC133 DIMM Speicher benötigt. Ein komplettes System auf Basis des FV24 Mainboards war als SV24[8] ebenfalls von Spacewalker verfügbar. Bei diesem „bare-bone System“ wurde ein Gehäuse von der Größe eines externen SCSI Gehäuses mit vormontiertem FV24 Mainboard geliefert. Der Preis für das Gehäuse mit Board lag bei 519 DM. Zudem wurde eine Festplatte von 60 GB eingebaut, um die Archiv-Funktionalität zu ermöglichen.

Zusammenfassung

Es wurde das vollständige PC-kompatible System SV24 der Firma Spacewalker vom Formfaktor FlexATX verwendet. Die Vorteile lagen in den geringen Kosten von Standardkomponenten für PCs, in deren hoher Leistung und in der guten Kompatibilität zu dem Betriebssystemen Linux. Die Gesamtkosten eines solchen Systems lagen bei etwa 800DM.

Nach der Entscheidung für ein PC-kompatibles System stellte sich noch die Frage, welche CPU zum Einsatz kommen sollte. Das Mainboard des SV24 Systems kann mit CPUs des Typs Sockel370 bestückt werden. Damit hatet man die Wahl zwischen den CPUs Celeron der Firma Intel und den CPUs vom Typ C3 der Firma VIA. Eine C3 CPU ist sinnvoller, da sie weniger Leistung benötigt als eine vergleichbare Intel CPU. Da die Abwärme in einem kleinen System wie dem SV24 eine Rolle spielen kann, ist die Entscheidung für die C3 CPU gefallen. Zudem wurde eine Festplatte der Größe 60 GB ausgewählt, um die Archiv-Funktionalität zu ermöglichen.

5.2 Bezugsquellen und Preise

Im Rahmen der Projektgruppe wurde eine Reihe von Hardwarekomponenten, die nach gründlichen Überlegungen ausgewählt wurden, verwendet. Darüberhinaus wurden einige Bauelemente für die Entwicklung eines an das Gesamtsystem angepassten LC-Displays, sowie das LCD selbst, benötigt. In der Tabelle 5.2 werden die einzelnen Bestandteile aufgeführt. Aufgrund der häufigen Preisschwankungen sind alle Preise nur Richtwerte. Außerdem sind einige Komponenten nicht mehr lieferbar (z.B. ist der C3-Prozessor nur noch mit höherer Taktfrequenz erhältlich). Der Stand der Preise und der Verfügbarket ist Juli 2002.

[3] `http://www.formfactors.org/developer/specs/sfx/sfx12v.pdf`
[4] `http://www.spacewalker.com/german`
[5] `http://www.reichelt.de`
[6] `http://www.spacewalker.com/german/faq/fv24.htm`
[7] Preise Stand 25. September 2001, „Sievers Computer & Software GmbH“, Dresden, `http://www.siecom.de/`
[8] `http://www.spacewalker.com/german/p_sv24.htm`

Menge	Bezeichnung	Art.Nr.	Einzel-preis/€	Anbieter
1	PC-133 INFINEON 512MB PC-133-CL3	-	140,00	diverse
1	SAMSUNG SV6004H 60.0GB	-	100,00	diverse
1	RICOH RW9200A (20xS/10xWS/40xL)	-	140,00	diverse
1	VIA C3 800MHz (Socket370 Prozessor)	-	49,00	ALTERNATE
1	SHUTTLE Barebone SV24 (Spacewalker)	-	299,00	SIEVERS
4	Elko 1uF/50V	426946	0,33	CONRAD
1	R/Trimm 50k	426865	0,64	CONRAD
1	IC 74LS00D	142867	0,49	CONRAD
1	IC MAX232DW TIS	152295	1,43	CONRAD
1	IC 74LS245DW	144398	0,84	CONRAD
1	Display DIP-EA-122-5Nled	181769	31,43	CONRAD
1	Buchsenleiste 20P/2mm	739243	2,20	CONRAD
2	R27 SMD 1206	402354	0,10	CONRAD
2	R0 SMD 1206	402222	0,10	CONRAD
2	Taster Miniprint prellfrei, schwarz	707600	0,41	CONRAD
1	Stiftleiste 5P/2, 54mm, gewinkelt	739464	0,23	CONRAD
1	Stiftleiste 2x8, gewinkelt	742171	0,51	CONRAD
1	Pfostensteckvb. 2x8, Kabelmont.	742198	1,00	CONRAD
1	Flachbandkabel 16P, 1m	609412	1,35	CONRAD
1	25P-SubD-Stiftl.	741671	0,59	CONRAD
1	Gehäuse Sub-D	711780	1,18	CONRAD
1	Satz Verriegelungsschrauben, kurz	711136	1,10	CONRAD

Tabelle 5.2: Hardware-Bestandteile und Bezugsquellen

5.2.1 Genaue Angaben zu den Bezugsquellen/Anbietern

K&M Elektronik
Blumenstrasse 21, 71106 Magstadt
Fax-Nr. 07159 / 943-222

SIEVERS COMPUTER UND SOFTWARE GMBH
Freiberger Str. 69-71, 01159 Dresden
Fax-Nr. 0351 / 86675-33

ALTERNATE Computerversand GmbH
Philipp-Reis-Str. 9, 35440 Linden
Fax-Nr. 06403 / 905020

CONRAD Electronic
Klaus Conrad Str. 1, 92240 Hirschau
Fax-Nr. 0180 / 5312110

5.3 Display und Taster

Das NetFire-System ist als Stand-Alone Gerät konzipiert, d.H. es soll ohne externe Geräte wie Tastatur, Maus und Monitor auskommen. Es ist aber notwendig, dem Benutzer Informationen mitzuteilen, ohne das Webinterface zu verwenden.

Soll z.B. eine Grund-Konfiguration erfolgen, wie das Einstellen der IP-Adresse des NetFire-Rechners, oder sollen „vor Ort“ Informationen angezeigt werden, z.B. welche CD gerade fertig gestellt wurde oder welcher Rohlingtyp als nächstes eingelegt werden muß, so wird ein Display am NetFire-Rechner unerläßlich.

Dieses benötigt natürlich auch Eingabemöglichkeiten, um Aktionen zu bestätigen oder zu unterbinden, aber auch um einfache Einstellungen zu treffen. Hierzu dienen zwei Taster, welche neben dem Display senkrecht angeordnet sind und eine situationsbedingte Eingabemöglichkeit bieten.

Da solche Benutzerschnittstellen nicht in einbaufertiger Form zu erwerben sind, müssen Display und Taster selbst gebaut werden.

5.3.1 Displayauswahl

Für die Auswahl des Displays wurden folgende Kriterien festgelegt:

- Grafikfähigkeit
- Größe entsprechend 3,5-Zoll Schacht
- Einfache Kontaktierung
- Ansteuerbarkeit per Parallelport (Anzahl der benötigten Leitungen)
- Hintergrundbeleuchtung
- Gute Beschaffungsmöglichkeiten

Zum Vergleich wurden mehrere Datenblätter und bereits vorliegende Displays herangezogen. Hierbei hob sich das Display 122-NLED von Electronic Assembly[9] hervor.

[9] http://www.electronic-assembly.de

Das Display zeichnet sich vor allem durch die Kontaktierung aus: Auf der Rückseite befinden sich zwei 9-Pol Stiftleisten im 2mm Rastermaß. So ist es möglich, das Display durch entsprechende Buchsenleisten auf eine Platine aufzustecken. Andere Displays werden üblicherweise durch Kontaktleisten am Rand der Displayplatine oder durch Flachleiter-Kabel kontaktiert. Der Aufbau einer Trägerplatine mit Buchsenleisten ist deutlich einfacher.

Das 122-NLED erfüllt auch alle weiteren oben genannten Kriterien: Es ist grafikfähig mit einer Auflösung von 122x32 Pixel, klein genug für einen Laufwerksschacht, läßt sich mit geringem Schaltaufwand am Parallelport betreiben, hat eine integrierte Hintergrundbeleuchtung und läßt sich über Conrad-Elektronik[10] unter der Bezeichnung DIP-EA-122-NLED beziehen.

5.3.2 Platinenentwurf und Bestückung

5.3.2.1 Problemstellung

Das EA-122-NLED verfügt über einige Eigenschaften, die ein direktes Anschließen an den Parallelport und die Spannungsversorgung eines PCs nicht erlauben. Zu lösende Probleme waren im einzelnen:

- Eine Kontrastspannung von -4V wird benötigt
- Das Display arbeitet bidirektional, der Parallelport soll unidirektional betrieben werden, um Kompatibilität zu älteren Systemen und einfache Treiberimplementierungen zu gewährleisten
- Um das Display vollständig ansteuern zu können, benötigt man eine Steuerleitung mehr als der Parallelport zur Verfügung stellt
- Stromversorgung für Hintergrundbeleuchtung
- Das Gesamtkonzept sieht zwei Taster für Benutzereingaben vor

5.3.2.2 Erzeugung einer negativen Kontrastspannung

Das Display benötigt eine Kontrastspannung von ca. -4 Volt. Da das Display über einen Laufwerksstecker mit Spannung versorgt werden soll, an dem nur +5V und +12V zur Verfügung stehen, muß die Kontrastspannung durch eine Schaltung auf der Displayplatine generiert werden.

Erzeugt wird die negative Spannung durch einen IC MAX232 (Abb. 5.1, IC2). Dieser IC ist eine Treiberstufe für die RS232-Schnittstelle und wird hier aufgrund seiner geringen Kosten verwendet. Er beinhaltet zwei „Ladungspumpen“, welche durch das parallele Laden und anschließendes „in-Reihe-Schalten“ von Kondensatoren Spannungen verdoppeln und auch umpolen kann. Der MAX232 erzeugt auf diese Weise aus der Betriebsspannung von 5 Volt die Ausgangspotenziale +10 Volt (am Pin V+) und -10 Volt (am Pin V-).

Die Ausgangsleitung „V-“ wird nun über ein Potentiometer (R1, 50kΩ) an das Display geführt, wodurch eine Kontrastregelung möglich ist. Dies ist zulässig, da das Display hierfür nur einen Strom von wenigen Microampere benötigt und so die Ladungspumpe nicht überlastet wird.

5.3.2.3 Bidirektionaler Zugriff

Das EA-122-NLED läßt sowohl schreibende wie auch lesende Zugriffe auf den 8-Bit breiten Bus zu. Kontrolliert wird der Zugriffsmodus durch ein Signal am Eingang „R/W“ des Displays.

[10] `http://www.conrad.de`

Lesende Zugriffe kann man weitgehend vermeiden. Diese bieten zwar die Möglichkeit, detaillierte Informationen über den Status des Displays zu erhalten, notwendig ist aber lediglich ein Auslesen des Busy-Signals. Dieses wird nach einer Statusanfrage auf einer Datenleitung (D7) übertragen. Das Display zeigt über dieses Signal an, daß die interne Abarbeitung des vorhergehenden Kommandos abläuft. Die Verwendung dieses Signals kann Zugriffe auf das Display stark beschleunigen, da nicht nach jedem Zugriff die maximale Verarbeitungszeit des Displays abgewartet werden muß.

Es war also nötig, im Falle von lesenden Zugriffen (das Signal R/W ist „low“, das Display befindet sich im Schreibmodus), den Datenbus zwischen Display und Parallelport zu trennen, um eine Datenkollision mit dem unidirektional betriebenen Parallelport zu vermeiden. Gleichzeitig muß aber das Busy-Signal des Displays am Parallelport anliegen.

Das Problem wurde durch Verwendung eines IC 74LS245DW (IC1) gelöst. Beim 74LS245DW handelt es sich um eine 8-Bit bidirektionale und „trennende“ Treiberstufe. Der IC ist so verschaltet, daß er im Falle von schreibenden Zugriffen auf das Display als Treiber zwischen Parallelport und Display fungiert, bei Lesezugriffen diese aber voneinander trennt.

Die Datenleitung D7 wurde zudem dauerhaft mit dem Busy-Eingang des Parallelports verbunden, so daß das Busy-Signal des Displays nach einer Status-Anfrage von dort gelesen werden kann.

5.3.2.4 Fehlende Steuerleitung

Da der Parallelport nicht mit genügend Steuerleitungen versehen ist – es fehlt eine Signalleitung für die Ansteuerung des Displays – wurden Signale folgendermaßen zusammengeführt: Intern arbeitet das 122NLED mit zwei Controllern vom Typ SED1520. Die Display-Signale E1 und E2 (INIT und AUTOFEED am Parallelport) dienen der Ansteuerung dieser Controller. Da es nicht zwingend erforderlich ist, beide Controller gleichzeitig anzusprechen, wurden die entsprechenden Datenleitungen mit einem NAND-Gatter (IC3, 74LS00D) wie folgt verschaltet: Immer wenn beide Controller gleichzeitig angesprochen werden (INIT/E1 und AUTOFEED/E2 high), dann schaltet das NAND seinen Ausgang auf low. Dieser Ausgang ist mit der Reset-Leitung verbunden (siehe Tabelle 5.3).

INIT	AUTOFEED	Funktion
low	low	inaktiv
low	high	E1 aktiv
high	low	E2 aktiv
high	high	Reset an E1 und E2

Tabelle 5.3: Displayfunktionen über INIT und AUTOFEED des Parallelports

5.3.2.5 Stromversorgung für die Hintergrundbeleuchtung

Die Hintergrundbeleuchtung des Displays kann problemlos an +5V vom PC betrieben werden, jedoch ist dabei zu bedenken, daß der Strom durch Widerstände (ca. 12 Ω) begrenzt werden sollte. Hierbei fließt ein Strom von ca. 200mA. Die Verlustleistung an den Widerständen ist entsprechend hoch, so daß diese sich stark erwärmen. Um die Erwärmung zu verringern, wurden deshalb 2x27 Ω parallel beschaltet. Dies Entspricht einem Gesamtwiderstand von 13,5 Ω.

5.3.2.6 Taster

Da am Parallelport noch Eingangsleitungen frei sind, können die Schalter ohne weiteren Schaltungsaufwand angeschlossen werden. Die Eingänge des Parallelports werden intern durch „Pull-Up-Widerstände“ auf High-Pegel gebracht. Die Taster müssen also lediglich den Eingang des

Parallelports bei Betätigung auf Low-Pegel bringen. Low-Pegel entspricht in der TTL-Logik des Parallelports einer Spannung von 0V, welche von der Stromversorgung bereitgestellt wird. Die verwendeten Eingänge sind „ERROR“ und „SELECT“.

Bei der Auswahl der Taster wurde darauf geachtet, daß diese prellfrei sind. Somit ist es unnötig, Entprellungsmechanismen in die Elektronik oder die Software einfließen zu lassen.

5.3.2.7 Platinenentwurf

Folgende Punkte wurden beachtet:

- Einseitiges Platinenlayout (aus Kostengründen)
- Senkrechte Montage im 3,5-Zoll Schacht
- Vorhandensein von Befestigungsmöglichkeiten im 3,5-Zoll Schacht

Es wurde ein Layout mit der CAD-Software Eagle[11] entworfen (siehe Abb. 5.2), welches diese Grundvorraussetzungen erfüllt. An der Vorderseite der Platine wird über zwei Buchsenleisten das Display befestigt. Rechts neben dem Display befinden sich in senkrechter Anordnung die Taster. Um das Layout einseitig zu halten, sind alle Bauteile für die frontseitige Bestückung verdrahtet, so daß sie auf der Rückseite verlötet werden. Alle weiteren Bauteile befinden sich in SMD-Gehäuseform auf der Rückseite der Platine. Dies bietet zusätzliche Vorteile: Der Platzbedarf sinkt und auf der Frontseite befinden sich ausschließlich die Bedienelemente.

5.3.3 Mechanische Befestigung am Gehäuse

Bereits beim Platinenlayout wurde die spätere Montage der fertig bestückten Platine im Floppyschacht berücksichtigt.

Um eine Befestigungsmöglichkeit für die Displayplatine zu haben, wurde am linken und rechten Rand Platz für große Lötkontakte vorgesehen (siehe Abb. 5.2). Auf diese Kontaktflächen wurden dann zwei passende Platinenstücke gelötet (siehe Abb. 5.7 und 5.8, Maßangaben in mm). Diese Befestigungsmethode kann vor allem Druck nach innen gut abfangen.

Um den Floppyschacht wieder nach vorne abzuschließen, wurde die ursprüngliche Aluminiumblende passend zersägt (siehe Abb. 5.5) und kann nun zwischen Halterung und Gehäuse festgeschraubt werden (siehe Abb. 5.6). In ein Blendenstück wurden noch zwei Löcher gebohrt, damit die Taster auch von außen bedienbar sind. Der Durchmesser der Bohrungen ist von den verwendeten Tastköpfen abhängig.

Verbindung mit dem Parallelport

Die Displayplatine wird durch ein Flachbandkabel mit dem Parallelport verbunden. Im Inneren des Gehäuses wurde ein Pfostenstecker, außen ein normaler SUB-D Stecker verwendet. Um das Flachbandkabel zu verstärken, wurde es mit Schrumpfschlauch ummantelt. Das Kabel wurde dann durch den Steckkarten-Schacht nach außen zum Parallelport verlegt. Dazu mußte das Blechstück aus dem Gehäuse herausgebrochen werden, das den PCI-Slot normalerweise verdeckt. Dieses Blechstück wurde dann durch ein Slotblech ersetzt, in das eine Einkerbung für das Kabel gesägt wurde.

[11] `http://www.cadsoft.com`

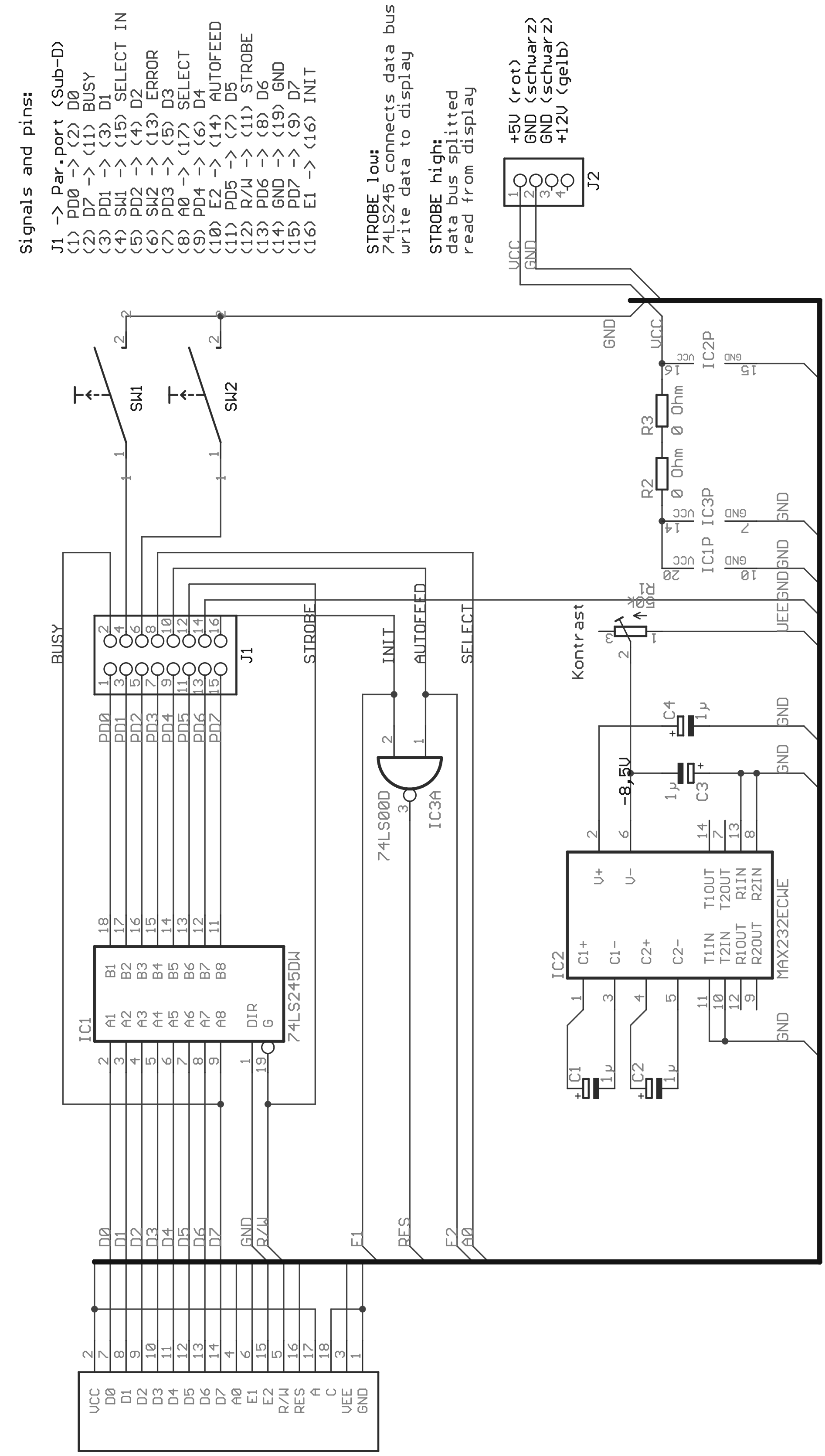

Abbildung 5.1: Schaltplan der LCD-Ansteuerung

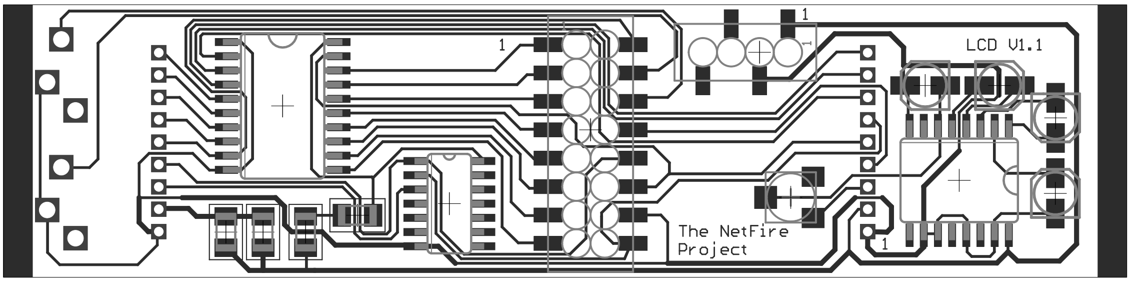

Abbildung 5.2: Platinenlayout der LCD-Ansteuerung

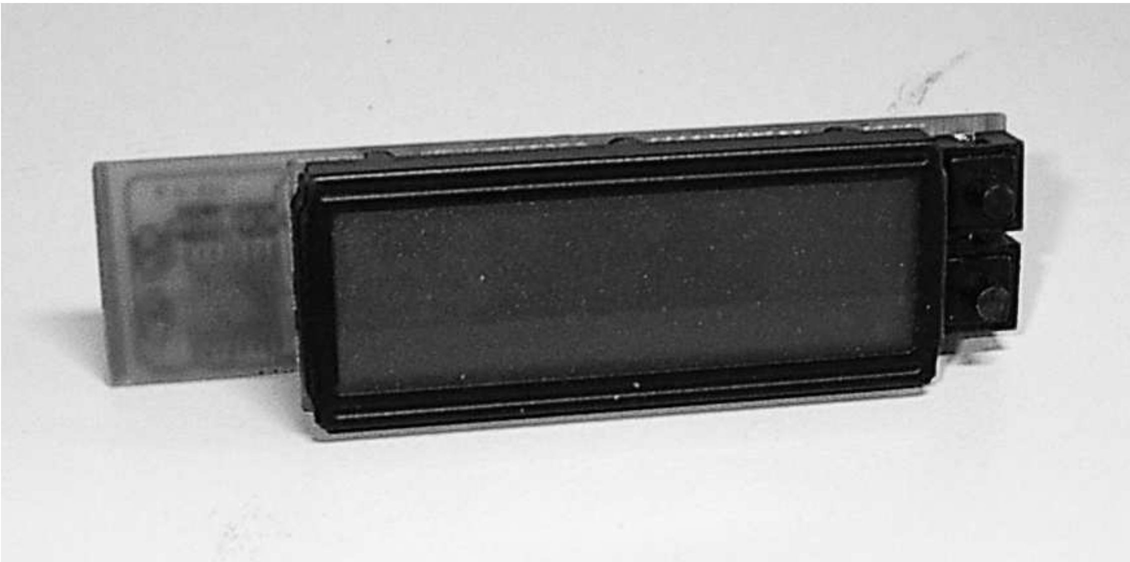

Abbildung 5.3: Bestückte Display-Platine, Vorderansicht

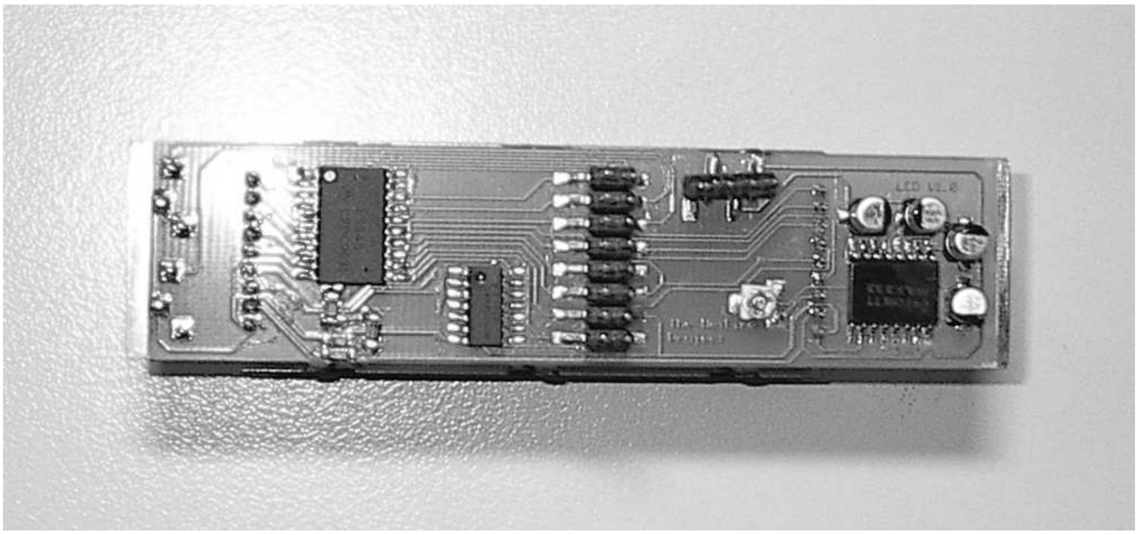

Abbildung 5.4: Bestückte Display-Platine, Rückseite

Abbildung 5.5: Zersägen und Durchbohren der Blende

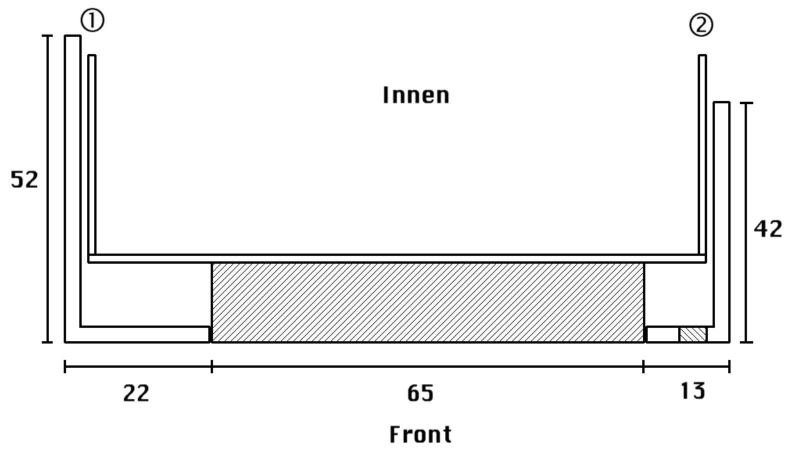

Abbildung 5.6: Einbauplan

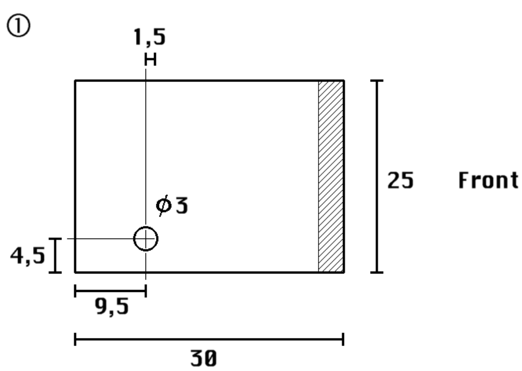

Abbildung 5.7: Linke Halterung

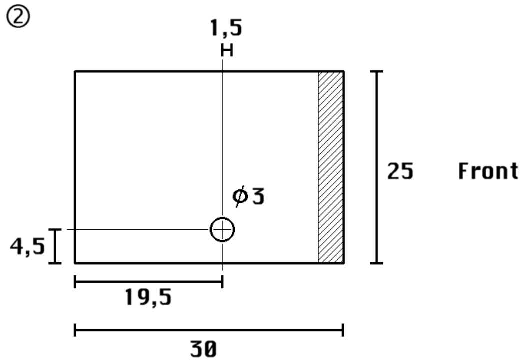

Abbildung 5.8: Rechte Halterung

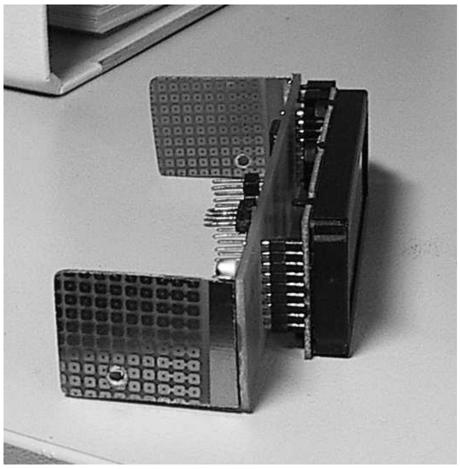

Abbildung 5.9: Platine mit angelöteten Halterungen

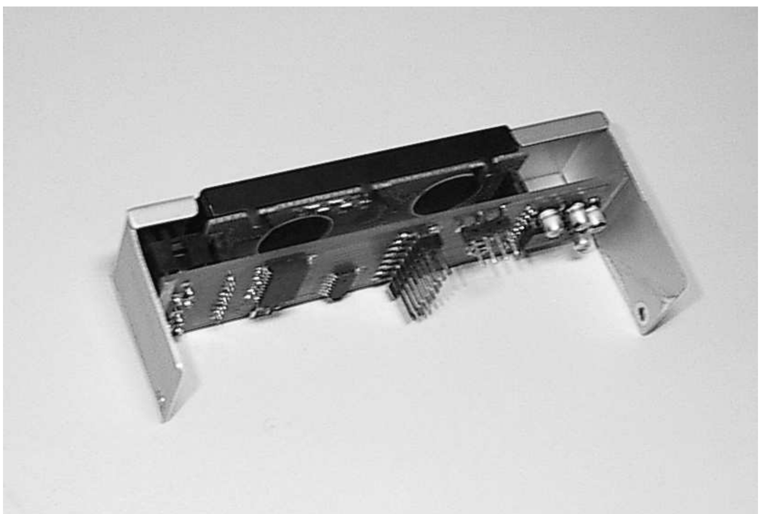

Abbildung 5.10: Blende und Platine

Abbildung 5.11: Das eingebaute Display

Kapitel 6

Systemtests

Zu jeder Entwicklungsarbeit gehört als qualitätssichernde Maßnahme sowohl entwicklungsbegleitend als auch abschließend das Testen dieser Arbeit. In dieser Projektgruppe wurde in einem relativ späten Stadium der Entwicklung, ca. 2 Monate vor Projektabschluss, eine zweiköpfige Test-Taskforce benannt, um diesem Aspekt genügend Rechnung zu tragen. Die vorgenommenen Tests können grob in die drei Kategorien Modultests, Integrationstests und Gesamtsystem-Tests unterteilt werden, welche im folgenden näher beschrieben werden.

6.1 Modul Tests

Bei Modultests wird nicht das gesamte Programm als Ganzes getestet, sondern nur möglichst kleine Teile. Diese Vorgehensweise bietet folgende drei Vorteile gegenüber anderen Testverfahren.

1. Fehler können schneller und besser lokalisiert werden, da der Ort des Fehlers direkt auf das getestete Modul beschränkt ist.

2. Einzeltests fallen weniger Komplex aus, da nicht alle Eingangsbedingungen kombiniert werden müssen.

3. Es Kann parallel getestet werden, was einen deutlichen Zeitvorteil bietet.

Die Aufgabe der Modultests ist der Vergleich der Modulfunktion mit deren Schnittstellendefinition, mit dem Ziel, vorhandene Widersprüche aufzufinden.

Unsere ersten Modultests fanden in Form von Whitebox Tests statt und lagen in der Verantwortung der jeweiligen Autoren dieser Module und wurden nicht dokumentiert. Erst nach erfolgreichem Abschluss der Whitebox Tests wurden die Module der Test Taskforce zu erneuten, diesmal von den Autoren unabhängigen, Tests übergeben. Diese fanden zum großen Teil in Form von Blackbox Tests, ohne Kenntnis des Quellcodes, statt.

6.1.1 `netfired`

Der Tatsache Rechnung tragend, daß der `netfired` die zentrale Softwarekomponente ist, wurde dem Testen desselbigen besondere Aufmerksamkeit geschenkt. Aufgrund seiner Struktur als permanent laufender Prozeß und unter Berücksichtigung seiner Schnittstellendefinition, in welcher sämtliche Kommunikation über Sockets festgelegt wurde, war es besonders gut möglich, ein automatisches Testverfahren zu generieren. Es wurde ein Programm in Perl geschrieben, welches in der Lage ist, beliebige seiner vordefinierten Befehle an den `netfired` zu schicken und automatisch zu Überprüfen, ob die Nachbedingung dieses Befehls eingehalten wird. Aus diesen einzelnen Befehlen und deren automatischer Überprüfung konnten dann kompliziertere Testfälle bestehend aus ganzen Befehlssequenzen konstruiert werden. Eine detaillierte Testdokumentation würde den Rahmen dieses Berichts sprengen, deshalb sind im folgenden auszugshaft und stichpunktartig einige Testfälle aufgelistet:

- Anlegen eines Accounts, Queuen des Accounts, Dequeuen des Accounts, Löschen des Accounts. Diese Befehlssequenz wurde nacheinander 1000 mal wiederholt.
- Anlegen von 100 Accounts.

  Dieser Fall führte erwartungsgemäß zu einem Fehler, da der Accountmaker per Spezifikation auf maximal 50 Accounts beschränkt sein soll.
- Löschen nicht vorhandener Accounts
- Übergabe eines \n Zeichens am Ende des Befehls

6.1.2 `lcdfired`

Für das Testen des LC-Displays, des dazugehörigen Linuxmoduletreibers, den `lcdfired` und des LCDSetup (Konfiguration des NetFire-Systems über Tasten am Display) mit der Whitebox Methode war der Autor verantwortlich. Die Blackbox Tests wurden von der Test-Taskforce mit einem Perlskript bzw. per Hand durchgeführt. Dabei wurden einige Fehler entdeckt, wie z.B. das die Position des ersten Zeichens auf dem LC-Display mittig war, die anschließend vom Programmierer behoben wurden. Einige Hardwareprobleme sind auch aufgetreten, wie z.B. das unterschiedliche Datenkabel mit gleicher Pinbelegung nicht funktionierten.

6.1.3 `ProFTPd`

Die `ProFTPd`-Komponenten (u.a. Accoutmaker und Accountdelter) wurden nur vom Programmierer selbst getestet. Eine Testdokumentation des `ProFTPd`'s durch die Test-Taskforce wurde im Zusammenhang mit dem Testen vom `netfired` durchgeführt.Im Zusammenspiel mit der WebGUI sind Probleme mit der Angabe des Upload-Verzeichnis aufgetreten, die beseitigt werden konnten.

6.1.4 Brenn-Skripte

Die Brennskripte sind bereits durch ihren Autor sehr modular gehalten. Dieser Umstand ermöglicht besonders einfache Modultests, da die Menge der Eingangsbedingungen besonders klein ist.

Im einzelnen übernehmen sie als erstes die Aufgabe, die Job-Control-Dateizu Parsen, welches anhand zahlreicher Files getestet wurde. Des weiteren werden durch diese Skripte die Daten in das zu brennende Ausgangsformat konvertiert und an der dafür spezifizierten Stelle abgelegt. Dies konnte Dank dedizierter Aufteilung der Skripte für die verschiedenen Formate sehr einfach anhand zahlreicher Testdaten auf Korrektheit überprüft werden. Weiterhin übergeben die Skripte ihre konvertierten Daten mit den aus der Job-Control-Dateigeparsten Parametern an die Programme `cdrecord` und `cdrdao`. Hier wurde getestet, ob die Übergabe der Skripte in einer in den man-pages der genannten Programmen definierten Weise stattfindet. Dies wurde durch die Tester zum einen mittels einer Durchsicht des Quellcodes, und zum anderen anhand von Testaufrufen von `cdrecord` und `cdrdao` mit den durch die Skripte generierten Parametern überprüft. Abschließend wurde die komplette Ablaufsteuerung des Brennvorgangs mit der dazugehörigen Schubladen-Logik und den Ausgaben, welche an das Display und die WebGUIgeleitet werden, getestet. Dazu wurde auch hier in eine Durchsicht des Quellcodes die Umsetzung der Skripte mit der vorhandenen Ablaufsteuerung verglichen. Des weiteren wurden die verschiedenen Möglichkeiten anhand von Testfällen mit entsprechenden Eingabedaten und Job-Control-Dateien simuliert durchlaufen und auf Korrektheit überprüft.

Insgesamt kann man sagen, daß in dem gesamten Bereich der Brennskripte besonders viele Fehler auftraten, was ein vergleichsweise intensives entwicklungsbegleitendes Testen zur Folge hatte. Die Ursache lag im Besonderen an mangelhaften Vortets des Autors. Durch eine sorgfältigere Durchführung dieser Tests hätte sich der zeitliche Aufwand für Entwicklung und Verifikation wohl entscheidend verringern lassen.

6.1.5 WebGUI

Ein Whitebox Test wurde von der Test-Taskforce nicht durchgeführt, sondern dies übernahm die WebGUI-Gruppe. Die Webseitenfunktionalitäten wurden nur im Zusammenhang mit dem Integration- und Gesamtsystemtest durchgeführt und dokumentiert.

6.2 Integrations Tests

Den Integrationstest kann man auch als Whitebox Test bezeichnen, da alle Systemkomponenten und deren Beziehungen untereinander während des Test sichtbar sind. Im Gegensatz dazu steht der Blackbox Test, bei dem das System nur über die nach außen geführten Schnittstellen, z.B. das Userinterface, betrachtet wird. Aufgabe des Integrationstests ist es, das fehlerfreie Zusammenwirken von Systemkomponenten und Teilmodulen zu überprüfen. Voraussetzungen für den Integrationstest ist, daß jede einzelne Systemkomponente in einem Komponenten- bzw. Modultest für sich geprüft ist. Stück für Stück werden alle Komponenten zusammengeführt (integriert) und das Zusammenwirken getestet. Komponenten, die im Test noch nicht integriert sind, werden durch Testtreiber bzw. Platzhalter ersetzt.

Die Test-Taskforce hat für einzelne größere Integrationstest kleine Testtreiber geschrieben und hauptsächlich per Eingabe der Parameter über die Konsole die Integrationstest durchgeführt. Testtreiber wurden für die LC-Display Ansteuerung und das Testen vom `netfired` geschrieben. Teilweise sind die aufgetretenen Fehler in den Bugreport geflossen oder direkt an den Programmierer gemail worden.

6.3 Tests des Gesamtsystems

Der Gesamtsystemtest wurde von der Test-Taskforce durchgeführt und in einem Bugreport dokumentiert. Hier ein kleiner Auszug aus dem Bugreport:

```
#########################
BUG 015
Datum:24.06.2002
Status: verified Bug - beckmann 24.06.2002
Fehlerquelle: Statusseiten
Verantwortlich: WebGUI
Testsystem: Embdebian
Fehlerbeschreibung: Wenn ein Job gebrannt wird, liefern die Statusseiten eine
"falsche JobID"
##########################
BUG 016
Datum:24.06.2002
Status: verified Bug - beckmann 24.06.2002
Fehlerquelle: vermutlich Brennskripte
Verantwortlich: Konstantin
```

```
Testsystem: High-level pc18 -> würfel embdebian
Fehlerbeschreibung: nach dem Brennen eines Jobs (mehrere Dateien) und der
Entnahme der CD steht auf dem Display:"CD wird gelesen"

Koll: Fehler ist nicht (mehr) rekonstruierbar. Bitte nochmal testen.
##########################
BUG 017
Datum:24.06.2002
Status: verified Bug - huesken 24.06.2002
Fehlerquelle: vermutlich Brennskripte
Testsystem: High-Level pc18 -> würfel embdebian
Fehlerbeschreibung: Es erfolgt ein Umsprung der LCD-Anzeige von "CD xxx
einlegen" auf "CD wird gelesen", ohne das eine CD im Brenner eingelegt wurde
bzw. die Lade geschlossen wurde

Koll: geht
##########################
BUG 018
Datum:24.06.2002
Status: FIXED           - Beckmann      03.07.2002
        verified Bug    - huesken       24.06.2002
Fehlerquelle: vermutlich Webseiten
Testsystem: High-Level pc18 -> würfel embdebian
Fehlerbeschreibung: In der Jobcontrol-Datei wird Multisession nicht auf "Yes"
gesetzt wenn man den Schalter in den Webseiten auswählt
##########################
BUG 019
Datum: 24.06.2002
Status: verified Bug - huesken 24.06.2002
Fehlerquelle: vermutlich Brennskripte
Testsystem: High-Level pc18 -> würfel embdebian
Fehlerbeschreibung: Die Option mehrere CD-Exemplare ohne Funktion

Koll: ist implementiert und sollte gehen
##########################
```

Kapitel 7

Die Archiv-CD

Die Archiv-CD der PG-NetFire enthält alle Daten, die die PG-Teilnehmer im Rahmen der zweisemestrigen PG-Phase verfaßt haben. Die CD enthält alle Informationen zum Projekt NetFire, um die Entwicklung verifizieren und verfolgen zu können. Die Archiv-CD wurde im ISO9660/Joliet-Format erstellt und stellt sicher, daß die gesamten Informationen und Daten der PG-NetFire am Lehrstuhl Informatik 12 archiviert sind.

7.1 Inhalt

Auf der Archiv-CD der PG-NetFire befindet sich unter anderem das gesamte „Concurrent Versions System“ (CVS) mit dem entwickelten Sourcecode, wie z.B. das weiterentwickelte Embedded Debian, die eigenentwickelten Daemonen LCDFired und NetFired, die Skripten und Webseiten. Im CVS sind desweiteren die Platinenlayouts und Schaltpläne für das LC-Display, der Zwischen- und Endbericht, sowie der Bugreport und ein wenig Artwork enthalten. Sonst beinhaltet die CD noch alle Protokolle der PG-Sitzungen, die Seminararbeiten der PG-Teilnehmer, Bilder der Hardware, Dokumentation der Hard- und Software und ein Verzeichnis `Sonstiges` mit Testarchiven, Sequenzdiagramme des NetFire-Systems und Fotos von den PG-Teilnehmern. Ein kleiner Auszug aus dem Verzeichnisbaum der Archiv-CD wird in Tabelle 7.1 beschrieben.

Verzeichnis	Beschreibung des Verzeichnis
CVS	CVS Hauptverzeichnis
CVS/CVSROOT	Verzeichnis mit CVS Informationsdateien
CVS/emdebsys	Hier befindet sich das gesamte Embedded Debian System
CVS/lcd	einiger Sourcecode zum LC-Display
CVS/netsoft	Hauptverzeichnis der eigenentwickleten NetFire-Software
CVS/netsoft/netfired	NetFire-Daemon zur Ablaufsteuerung von NetFire
CVS/netsoft/proftpd	ProFTPd-Daemon für die FTP-Accounts auf NetFire
CVS/netsoft/jobtool	Einige Parser-Skripten
CVS/netsoft/webgui	Webseiten von NetFire
CVS/netsoft/lcdfired	Daemon zum Nachricht senden über das LCD
CVS/netsoft/skripts	Aufräumen von verwaisten Home-Verzeichnissen
CVS/netsoft/lcdsetup	Eingabe von IP usw. über LC-Display
CVS/netsoft/configurator	Skript zum setzen von Config-Dateien
CVS/netsoft/install	Skripten für die Install-CD
CVS/netsoft/liblcd	Libary des LC-Display Treibers
CVS/netsoft/cd_ok	Sourcecode zum Schublabenhandling des Brenners
CVS/eagle	Eagle-Schaltpläne der Display-Platine
CVS/zb	Hauptverzeichnis des Zwischenberichts
CVS/liblcd	Linux-Kernel-Modul fürs LC-Display
CVS/interfaces	Beschreibung der Interfaces der NetFire-Module
CVS/bugreport	Fehlerbericht von NetFirewährend der Entwicklung
CVS/endbericht	Hauptverzeichnis des Endberichts
CVS/artwork	NetFire-CD-Aufkleber, -CD-Hüllencover und -Logos
DOCS/hardware/lcd	Bilder vom LC-Display
DOCS/hardware/sv24	Bilder vom der Hardware
DOCS/SEMINARE	Seminararbeiten der PG-Teilnehmer
DOCS/software	Dokumentationen, CD-ROM Spezifikationen
DOCS/BESTELLUNGEN	Bestellungen der Hardware
PROTOKOLLE	Protokolle der PG-Sitzungen
SONSTIGE	Fotos der PG-Teilnehmer, Sequenzdiagramme, Testarchive

Tabelle 7.1: Übersicht einige Verzeichnisse auf der Archiv-CD

Kapitel 8

Installationsanleitung

Einleitung

In diesem Kapitel werden die Schritte beschrieben, wie man zu einem funktionierenden NetFire-System gelangt. Dabei gibt es drei Möglichkeiten, die jeweils einen Kompromiß zwischen der Einfachheit der Installation und der Flexibilität darstellen.

Installation von der fertigen CD

Bei dieser (einfachsten) Installationsvariante wird die Installations-CD in den Zielrechner eingelegt und dieser wird von ihr gebootet. Allerdings erfordert diese Installationsform, daß der Rechner einige Voraussetzungen erfüllt (im wesentlichen, daß er weitgehend wie der Entwicklungsrechner konfiguriert ist). Diese Voraussetzungen sind:

- Die Festplatte muß Master am primären IDE-Controller sein
- Das CD-ROM-Laufwerk muß Master am sekundären Controller sein

Erstellen der Installations-CD bei vorhandenem Image

Bei kleineren Änderungen (wie z.B. einem von dem Entwicklungsrechner unterschiedlichen CD-ROM-Pfad) genügt es, die Installations-CD neu zu generieren. Dazu müssen folgende Voraussetzungen geschaffen werden:

- Im Arbeitsverzeichnis müssen die Module `emdebsys` und `netsoft` sowie im Unterverzeichnis `linux` die Kernelquellen des Linux-Kernels 2.4.18 liegen.
- Ebenfalls im Arbeitsverzeichnis muß unter dem Namen `vmlinuz` eine Kernelversion liegen, die als Boot-Kernel für das installierte System verwendet werden soll
- Unter `/tmp/emdeb-files.tgz` muß ein mit `Emdebian` erzeugtes Root-Filesystem inkl. des Netfire-Systems liegen
- Es werden (zusätzlich zu den Voraussetzungen von Emdebian) die Tools `/sbin/losetup` und `/sbin/mke2fs` sowie das Paket `GRUB`, insbesondere die Dateien `/usr/sbin/grub` sowie `/boot/grub/stage1` und `/boot/grub/stage2`.

Die erforderlichen Anpassungen können durch die Variablen am Anfang des Skriptes
`netsoft/install/scripts/bin/partition/rc.S`
vorgenommen werden. Danach wechselt man in das Verzeichnis `netsoft/install` und ruft das Skript `./makeimage.sh` auf. Das Ergebnis findet sich unter `/tmp/netfire.iso` und kann mit entsprechenden Tools auf eine CD gebrannt werden.

Erstellen des Images

Sollten große Anpassungen notwendig sein (z.B.: SCSI statt IDE, oder ein anderes Display), muß das Emdebian-Image vorher erstellt werden. Dazu sind folgende Voraussetzungen zu erfüllen:

- Im Arbeitsverzeichnis müssen die Module `emdebsys`, `lcd` und `netsoft` sowie im Unterverzeichnis `emdebsys/snp/snp-cache/linux` die Kernelquellen des Linux-Kernels 2.4.18 liegen.

Nachdem die notwendigen Anpassungen vorgenommen wurden, wird ein Root-Filesystem erzeugt, indem man aus dem Arbeitsverzeichnis (im Folgenden $WORKDIR genannt) folgende Schritte ausführt [1]:

```
cd netsoft
su -c 'make clean'
./configure --prefix=/ --enable-full --with-webgui=/opt/var/www \
       --localstatedir=/opt/var
su -c 'make tgz'
cd ..

cd lcd
su -c 'make clean'
make INCLUDEDIR=$WORKDIR/emdebsys/snp/snp-cache/linux/include/
su -c 'make tgz PREFIX=/tmp/lcdtgz \
       INCLUDEDIR=$WORKDIR/emdebsys/snp/snp-cache/linux/include/'
cd ..

cp /tmp/netfiretgz/netfire.tgz $WORKDIR/emdebsys/addons
cd /tmp/netfiretgz
su -c 'tar czf netfireopttgz.tgz netfireopt.tgz'
cp /tmp/netfiretgz/netfireopt.tgz $WORKDIR/emdebsys/addons/
cp /tmp/netfiretgz/netfireopttgz.tgz $WORKDIR/emdebsys/addons/
cp /tmp/netfirelcd.tgz $WORKDIR/emdebsys/addons

cd $WORKDIR/emdebsys/snp
su -c './snp.py >snp.log'
cd $WORKDIR
```

Die dabei entstehende Datei `/tmp/emdeb-files.tgz` kann direkt für die Erzeugung der Installations-CD verwendet werden – den Kernel dazu findet man unter `$WORKDIR/emdebsys/snp/snp-cache/linux/arch/i386/boot/bzImage`. Eine detailliertere Beschreibung des Emdebian-Systems findet sich unter 4.2.2.

[1] Dadurch daß bei der Generierung des emdebsys-Systems Aktionen ausgeführt werden müssen, die Superuser-Rechte erfordern (Anlegen von Device-Files, Mounten von Dateisystemen usw.) sind die „su -c“- Einleitungen notwendig

Kapitel 9

Fazit

Jeder Teilnehmer dieser Projektgruppe wird wohl sein ganz persönliches eigenes Fazit aus der Arbeit der vergangenen 12 Monate ziehen. Besonders herauszustellen sind aber einige Punkte, welche wohl Allgemeingültigkeit für die Gruppe besitzen.

Während der Projektgruppe hat sich wohl für jeden gezeigt, wie wichtig eine längere Planung und ausführliche Recherche vor dem Bearbeiten eines größeren Problems sind. Ohne diese hätte unser Fokus bei der Entwicklung wohl auf anderen Aspekten gelegen und wir hätten nicht das System entwickelt welche nun vorliegt. Vielmehr war den Stimmen der einzelnen Mitglieder vor der Planungsphase zu entnehmen, daß sie sich auf größere Hardware-Entwicklungsarbeiten eingestellt haben. Einzig und allein der guten Recherche mit den daraus folgerichtig gezogenen Konsequenzen in der Seminar- und Planungsphase ist es wohl zu verdanken, daß wir uns am Ende weitestgehend auf Standard PC-Komponenten beschränkt haben und einen großen Teil unserer Ressourcen in die weitere Entwicklung von Software investieren konnten. Gerade in den letzten Monaten der Projektgruppe hat es sich gezeigt, wie aufwendig sich der von uns eingeschlagene Weg erwies. Zum jetzigen Zeitpunkt sei die Mutmaßung erlaubt, daß wir das Minimalziel der Projektgruppe in dieser Zeit wohl nicht hätten erreichen können, wenn wir wie anfänglich vermutet große Teile der Hardware selber entwickelt hätten.

Auch zu erwähnen ist, die Erkenntnis, daß wohl jeder Teilnehmer der Projektgruppe ganz neue Erfahrungen in Bezug auf Teamarbeit gesammelt hat. So hat es während dieser Projektgruppe Höhen und Tiefen im Klima gegeben, welche sich jedoch letztendlich äußerst fruchtbar in vorliegendem Ergebnis niedergeschlagen haben. Dabei fiel auf, daß gerade in schwierigen Zeiten der direkte Kommunikationsbedarf erhöht war, der sich nur schwer bis gar nicht durch Email Verkehr befriedigen ließ. Gerade bei der Integrationsarbeit machte sich die Erkentnis breit, daß es Phasen gibt, in denen ein echtes Treffen aller Beteiligten wenn nicht sogar zwingend nötig ist, so zumindest einiges sehr viel einfacher macht und den Zeitaufwand um ein Vielfaches reduziert. Des weiteren werden Mißverständnisse, welche auch trotz guter Absprachen sowie Schnittellen-, und Modulspezifikationen immer wieder auftauchen, schnell beseitigt oder sogar ganz vermieden.

Nun, am Ende der Projektgruppe, bleibt jedem Teilnehmer das Gefühl, während zwei Semestern etwas von seinem zuvor im Studium erworbenen Wissen zur praktischen Anwendung gebracht zu haben und dabei etwas geplant, entworfen und letztendlich auch gebaut zu haben, was einen echten praktischen Nutzen hat. Das Gerät kommt am LS12 zum Einsatz und kann von Jedermann nachgebaut werden.

Wenn man ein Resumee über die gesamte Projektarbeit in einem Satz formulieren wollte, so käme man sicherlich nur zu dem Schluß, daß jeder Teilnehmer innerhalb der letzten zwei Semester ein großes Maß an praktischer Erfahrung gesammelt hat und gelernt hat, ein größeres Problem durch Zerlegung in Teilprobleme und koordinierte Bearbeitung der Teilprobleme in einem strukturierten Team zu bewältigen. Dieses Kurzresumee entspricht ziemlich genau dem Lernziel einer Projektgruppe laut Projektgruppenordnung [23] was zeigt, daß jedes Projektgruppenmitglied etwas Sinnvolles gelernt, und sich für jeden persönlich das vordefinierte Lernziel als Fazit ergeben hat.

Anhang A

GPL

GNU General Public License
Deutsche Übersetzung der Version 2, Juni 1991 [19]

Diese Übersetzung ist kein rechtskräftiger Ersatz für die englischsprachige Originalversion!

A.1 Vorwort

Die meisten Softwarelizenzen sind daraufhin entworfen worden, Ihnen die Freiheit zu nehmen, die Software weiterzugeben und zu verändern. Im Gegensatz dazu soll Ihnen die GNU General Public License , die Allgemeine Öffentliche GNU-Lizenz, ebendiese Freiheit garantieren. Sie soll sicherstellen, daß die Software für alle Benutzer frei ist. Diese Lizenz gilt für den Großteil der von der Free Software Foundation herausgegebenen Software und für alle anderen Programme, deren Autoren ihr Datenwerk dieser Lizenz unterstellt haben. Auch Sie können diese Möglichkeit der Lizenzierung für Ihre Programme anwenden. (Ein anderer Teil der Software der Free Software Foundation unterliegt stattdessen der GNU Library General Public License , der Allgemeinen Öffentlichen GNU-Lizenz für Bibliotheken.) [Mittlerweile wurde die GNU Library Public License von der GNU Lesser Public License abgelöst - Anmerkung des Übersetzers.]

Die Bezeichnung „freie"Software bezieht sich auf Freiheit, nicht auf den Preis. Unsere Lizenzen sollen Ihnen die Freiheit garantieren, Kopien freier Software zu verbreiten (und etwas für diesen Service zu berechnen, wenn Sie möchten), die Möglichkeit, die Software im Quelltext zu erhalten oder den Quelltext auf Wunsch zu bekommen. Die Lizenzen sollen garantieren, daß Sie die Software ändern oder Teile davon in neuen freien Programmen verwenden dürfen - und daß Sie wissen, daß Sie dies alles tun dürfen.

Um Ihre Rechte zu schützen, müssen wir Einschränkungen machen, die es jedem verbieten, Ihnen diese Rechte zu verweigern oder Sie aufzufordern, auf diese Rechte zu verzichten. Aus diesen Einschränkungen folgen bestimmte Verantwortlichkeiten für Sie, wenn Sie Kopien der Software verbreiten oder sie verändern.

Beispielsweise müssen Sie den Empfängern alle Rechte gewähren, die Sie selbst haben, wenn Sie - kostenlos oder gegen Bezahlung - Kopien eines solchen Programms verbreiten. Sie müssen sicherstellen, daß auch die Empfänger den Quelltext erhalten bzw. erhalten können. Und Sie müssen ihnen diese Bedingungen zeigen, damit sie ihre Rechte kennen.

Wir schützen Ihre Rechte in zwei Schritten: (1) Wir stellen die Software unter ein Urheberrecht (Copyright), und (2) wir bieten Ihnen diese Lizenz an, die Ihnen das Recht gibt, die

Software zu vervielfältigen, zu verbreiten und/oder zu verändern.

Um die Autoren und uns zu schützen, wollen wir darüberhinaus sicherstellen, daß jeder erfährt, daß für diese freie Software keinerlei Garantie besteht. Wenn die Software von jemand anderem modifiziert und weitergegeben wird, möchten wir, daß die Empfänger wissen, daß sie nicht das Original erhalten haben, damit irgendwelche von anderen verursachte Probleme nicht den Ruf des ursprünglichen Autors schädigen.

Schließlich und endlich ist jedes freie Programm permanent durch Software-Patente bedroht. Wir möchten die Gefahr ausschließen, daß Distributoren eines freien Programms individuell Patente lizensieren - mit dem Ergebnis, daß das Programm proprietär würde. Um dies zu verhindern, haben wir klargestellt, daß jedes Patent entweder für freie Benutzung durch jedermann lizenziert werden muß oder überhaupt nicht lizenziert werden darf.

Es folgen die genauen Bedingungen für die Vervielfältigung, Verbreitung und Bearbeitung:

A.2 Allgemeine Öffentliche GNU-Lizenz Bedingungen für die Vervielfältigung, Verbreitung und Bearbeitung

§ 0.

Diese Lizenz gilt für jedes Programm und jedes andere Datenwerk, in dem ein entsprechender Vermerk des Copyright-Inhabers darauf hinweist, daß das Datenwerk unter den Bestimmungen dieser General Public License verbreitet werden darf. Im folgenden wird jedes derartige Programm oder Datenwerk als ,,das Programm" bezeichnet; die Formulierung ,,auf dem Programm basierendes Datenwerk" bezeichnet das Programm sowie jegliche Bearbeitung des Programms im urheberrechtlichen Sinne, also ein Datenwerk, welches das Programm, auch auszugsweise, sei es unverändert oder verändert und/oder in eine andere Sprache übersetzt, enthält. (Im folgenden wird die Übersetzung ohne Einschränkung als ,,Bearbeitung" eingestuft.) Jeder Lizenznehmer wird im folgenden als ,,Sie" angesprochen.

Andere Handlungen als Vervielfältigung, Verbreitung und Bearbeitung werden von dieser Lizenz nicht berührt; sie fallen nicht in ihren Anwendungsbereich. Der Vorgang der Ausführung des Programms wird nicht eingeschränkt, und die Ausgaben des Programms unterliegen dieser Lizenz nur, wenn der Inhalt ein auf dem Programm basierendes Datenwerk darstellt (unabhängig davon, daß die Ausgabe durch die Ausführung des Programmes erfolgte). Ob dies zutrifft, hängt von den Funktionen des Programms ab.

§ 1.

Sie dürfen auf beliebigen Medien unveränderte Kopien des Quelltextes des Programms, wie sie ihn erhalten haben, anfertigen und verbreiten. Voraussetzung hierfür ist, daß Sie mit jeder Kopie einen entsprechenden Copyright-Vermerk sowie einen Haftungsausschluß veröffentlichen, alle Vermerke, die sich auf diese Lizenz und das Fehlen einer Garantie beziehen, unverändert lassen und desweiteren allen anderen Empfängern des Programms zusammen mit dem Programm eine Kopie dieser Lizenz zukommen lassen.

Sie dürfen für den eigentlichen Kopiervorgang eine Gebühr verlangen. Wenn Sie es wünschen, dürfen Sie auch gegen Entgeld eine Garantie für das Programm anbieten.

§ 2.

Sie dürfen Ihre Kopie(n) des Programms oder eines Teils davon verändern, wodurch ein auf dem Programm basierendes Datenwerk entsteht; Sie dürfen derartige Bearbeitungen unter den Bestimmungen von Paragraph 2 vervielfältigen und verbreiten, vorausgesetzt, daß zusätzlich alle im folgenden genannten Bedingungen erfüllt werden:

1. Sie müssen die veränderten Dateien mit einem auffälligen Vermerk versehen, der auf die von Ihnen vorgenommene Modifizierung und das Datum jeder Änderung hinweist.

2. Sie müssen dafür sorgen, daß jede von Ihnen verbreitete oder veröffentlichte Arbeit, die ganz oder teilweise von dem Programm oder Teilen davon abgeleitet ist, Dritten gegenüber als Ganzes unter den Bedingungen dieser Lizenz ohne Lizenzgebühren zur Verfügung gestellt wird.

3. Wenn das veränderte Programm normalerweise bei der Ausführung interaktiv Kommandos einliest, müssen Sie dafür sorgen, daß es, wenn es auf dem üblichsten Wege für solche interaktive Nutzung gestartet wird, eine Meldung ausgibt oder ausdruckt, die einen geeigneten Copyright-Vermerk enthält sowie einen Hinweis, daß es keine Gewährleistung gibt (oder anderenfalls, daß Sie Garantie leisten), und daß die Benutzer das Programm unter diesen Bedingungen weiter verbreiten dürfen. Auch muß der Benutzer darauf hingewiesen werden, wie er eine Kopie dieser Lizenz ansehen kann. (Ausnahme: Wenn das Programm selbst interaktiv arbeitet, aber normalerweise keine derartige Meldung ausgibt, muß Ihr auf dem Programm basierendes Datenwerk auch keine solche Meldung ausgeben).

Diese Anforderungen gelten für das bearbeitete Datenwerk als Ganzes. Wenn identifizierbare Teile des Datenwerkes nicht von dem Programm abgeleitet sind und vernünftigerweise als unabhängige und eigenständige Datenwerke für sich selbst zu betrachten sind, dann gelten diese Lizenz und ihre Bedingungen nicht für die betroffenen Teile, wenn Sie diese als eigenständige Datenwerke weitergeben. Wenn Sie jedoch dieselben Abschnitte als Teil eines Ganzen weitergeben, das ein auf dem Programm basierendes Datenwerk darstellt, dann muß die Weitergabe des Ganzen nach den Bedingungen dieser Lizenz erfolgen, deren Bedingungen für weitere Lizenznehmer somit auf das gesamte Ganze ausgedehnt werden - und somit auf jeden einzelnen Teil, unabhängig vom jeweiligen Autor.

Somit ist es nicht die Absicht dieses Abschnittes, Rechte für Datenwerke in Anspruch zu nehmen oder Ihnen die Rechte für Datenwerke streitig zu machen, die komplett von Ihnen geschrieben wurden; vielmehr ist es die Absicht, die Rechte zur Kontrolle der Verbreitung von Datenwerken, die auf dem Programm basieren oder unter seiner auszugsweisen Verwendung zusammengestellt worden sind, auszuüben.

Ferner bringt auch das einfache Zusammenlegen eines anderen Datenwerkes, das nicht auf dem Programm basiert, mit dem Programm oder einem auf dem Programm basierenden Datenwerk auf ein- und demselben Speicher- oder Vertriebsmedium dieses andere Datenwerk nicht in den Anwendungsbereich dieser Lizenz.

§ 3.

Sie dürfen das Programm (oder ein darauf basierendes Datenwerk gemäß Paragraph 3) als Objectcode oder in ausführbarer Form unter den Bedingungen der Paragraphen 2 und 3 kopieren und weitergeben - vorausgesetzt, daß Sie außerdem eine der folgenden Leistungen erbringen:

1. Liefern Sie das Programm zusammen mit dem vollständigen zugehörigen maschinenlesbaren Quelltext auf einem für den Datenaustausch üblichen Medium aus, wobei die Verteilung unter den Bedingungen der Paragraphen 2 und 3 erfolgen muß. Oder:

2. Liefern Sie das Programm zusammen mit einem mindestens drei Jahre lang gültigen schriftlichen Angebot aus, jedem Dritten eine vollständige maschinenlesbare Kopie des Quelltextes zur Verfügung zu stellen - zu nicht höheren Kosten als denen, die durch den physikalischen Kopiervorgang anfallen -, wobei der Quelltext unter den Bedingungen der Paragraphen 2 und 3 auf einem für den Datenaustausch üblichen Medium weitergegeben wird. Oder:

3. Liefern Sie das Programm zusammen mit dem schriftlichen Angebot der Zurverfügungstellung des Quelltextes aus, das Sie selbst erhalten haben. (Diese Alternative ist nur für nicht-kommerzielle Verbreitung zulässig und nur, wenn Sie das Programm als Objectcode oder in ausführbarer Form mit einem entsprechenden Angebot gemäß Absatz b erhalten haben.)

Unter dem Quelltext eines Datenwerkes wird diejenige Form des Datenwerkes verstanden, die für Bearbeitungen vorzugsweise verwendet wird. Für ein ausführbares Programm bedeutet „der komplette Quelltext“: Der Quelltext aller im Programm enthaltenen Module einschließlich aller zugehörigen Modulschnittstellen-Definitionsdateien sowie der zur Compilation und Installation verwendeten Skripte. Als besondere Ausnahme jedoch braucht der verteilte Quelltext nichts von dem zu enthalten, was üblicherweise (entweder als Quelltext oder in binärer Form) zusammen mit den Hauptkomponenten des Betriebssystems (Kernel, Compiler usw.) geliefert wird, unter dem das Programm läuft - es sei denn, diese Komponente selbst gehört zum ausführbaren Programm.

Wenn die Verbreitung eines ausführbaren Programms oder von Objectcode dadurch erfolgt, daß der Kopierzugriff auf eine dafür vorgesehene Stelle gewährt wird, so gilt die Gewährung eines gleichwertigen Zugriffs auf den Quelltext als Verbreitung des Quelltextes, auch wenn Dritte nicht dazu gezwungen sind, den Quelltext zusammen mit dem Objectcode zu kopieren.

§ 4.

Sie dürfen das Programm nicht vervielfältigen, verändern, weiter lizenzieren oder verbreiten, sofern es nicht durch diese Lizenz ausdrücklich gestattet ist. Jeder anderweitige Versuch der Vervielfältigung, Modifizierung, Weiterlizenzierung und Verbreitung ist nichtig und beendet automatisch Ihre Rechte unter dieser Lizenz. Jedoch werden die Lizenzen Dritter, die von Ihnen Kopien oder Rechte unter dieser Lizenz erhalten haben, nicht beendet, solange diese die Lizenz voll anerkennen und befolgen.

§ 5.

Sie sind nicht verpflichtet, diese Lizenz anzunehmen, da Sie sie nicht unterzeichnet haben. Jedoch gibt Ihnen nichts anderes die Erlaubnis, das Programm oder von ihm abgeleitete Datenwerke zu verändern oder zu verbreiten. Diese Handlungen sind gesetzlich verboten, wenn Sie diese Lizenz nicht anerkennen. Indem Sie das Programm (oder ein darauf basierendes Datenwerk) verändern oder verbreiten, erklären Sie Ihr Einverständnis mit dieser Lizenz und mit allen ihren Bedingungen bezüglich der Vervielfältigung, Verbreitung und Veränderung des Programms oder eines darauf basierenden Datenwerks.

§ 6.

Jedesmal, wenn Sie das Programm (oder ein auf dem Programm basierendes Datenwerk) weitergeben, erhält der Empfänger automatisch vom ursprünglichen Lizenzgeber die Lizenz, das Programm entsprechend den hier festgelegten Bestimmungen zu vervielfältigen, zu verbreiten und zu verändern. Sie dürfen keine weiteren Einschränkungen der Durchsetzung der hierin zugestandenen Rechte des Empfängers vornehmen. Sie sind nicht dafür verantwortlich, die Einhaltung dieser Lizenz durch Dritte durchzusetzen.

§ 7.

Sollten Ihnen infolge eines Gerichtsurteils, des Vorwurfs einer Patentverletzung oder aus einem anderen Grunde (nicht auf Patentfragen begrenzt) Bedingungen (durch Gerichtsbeschluß, Vergleich oder anderweitig) auferlegt werden, die den Bedingungen dieser Lizenz widersprechen, so befreien Sie diese Umstände nicht von den Bestimmungen dieser Lizenz. Wenn es Ihnen nicht möglich ist, das Programm unter gleichzeitiger Beachtung der Bedingungen in dieser Lizenz und Ihrer anderweitigen Verpflichtungen zu verbreiten, dann dürfen Sie als Folge das Programm überhaupt nicht verbreiten. Wenn zum Beispiel ein Patent nicht die gebührenfreie Weiterverbreitung des Programms durch diejenigen erlaubt, die das Programm direkt oder indirekt von Ihnen erhalten haben, dann besteht der einzige Weg, sowohl das Patentrecht als auch diese Lizenz zu befolgen, darin, ganz auf die Verbreitung des Programms zu verzichten.

Sollte sich ein Teil dieses Paragraphen als ungültig oder unter bestimmten Umständen nicht durchsetzbar erweisen, so soll dieser Paragraph seinem Sinne nach angewandt werden; im übrigen soll dieser Paragraph als Ganzes gelten.

Zweck dieses Paragraphen ist nicht, Sie dazu zu bringen, irgendwelche Patente oder andere Eigentumsansprüche zu verletzen oder die Gültigkeit solcher Ansprüche zu bestreiten; dieser Paragraph hat einzig den Zweck, die Integrität des Verbreitungssystems der freien Software zu schützen, das durch die Praxis öffentlicher Lizenzen verwirklicht wird. Viele Leute haben großzügige Beiträge zu dem großen Angebot der mit diesem System verbreiteten Software im Vertrauen auf die konsistente Anwendung dieses Systems geleistet; es liegt am Autor/Geber, zu entscheiden, ob er die Software mittels irgendeines anderen Systems verbreiten will; ein Lizenznehmer hat auf diese Entscheidung keinen Einfluß.

Dieser Paragraph ist dazu gedacht, deutlich klarzustellen, was als Konsequenz aus dem Rest dieser Lizenz betrachtet wird.

§ 8.

Wenn die Verbreitung und/oder die Benutzung des Programms in bestimmten Staaten entweder durch Patente oder durch urheberrechtlich geschützte Schnittstellen eingeschränkt ist, kann der Urheberrechtsinhaber, der das Programm unter diese Lizenz gestellt hat, eine explizite geographische Begrenzung der Verbreitung angeben, in der diese Staaten ausgeschlossen werden, so daß die Verbreitung nur innerhalb und zwischen den Staaten erlaubt ist, die nicht ausgeschlossen sind. In einem solchen Fall beinhaltet diese Lizenz die Beschränkung, als wäre sie in diesem Text niedergeschrieben.

§ 9.

Die Free Software Foundation kann von Zeit zu Zeit überarbeitete und/oder neue Versionen der General Public License veröffentlichen. Solche neuen Versionen werden vom Grundprinzip her der gegenwärtigen entsprechen, können aber im Detail abweichen, um neuen Problemen und Anforderungen gerecht zu werden.

Jede Version dieser Lizenz hat eine eindeutige Versionsnummer. Wenn in einem Programm angegeben wird, daß es dieser Lizenz in einer bestimmten Versionsnummer oder „jeder späteren Version“ (“any later version”) unterliegt, so haben Sie die Wahl, entweder den Bestimmungen der genannten Version zu folgen oder denen jeder beliebigen späteren Version, die von der Free Software Foundation veröffentlicht wurde. Wenn das Programm keine Versionsnummer angibt, können Sie eine beliebige Version wählen, die je von der Free Software Foundation veröffentlicht wurde.

§ 10.

Wenn Sie den Wunsch haben, Teile des Programms in anderen freien Programmen zu verwenden, deren Bedingungen für die Verbreitung anders sind, schreiben Sie an den Autor, um ihn um die Erlaubnis zu bitten. Für Software, die unter dem Copyright der Free Software Foundation steht, schreiben Sie an die Free Software Foundation ; wir machen zu diesem Zweck gelegentlich Ausnahmen. Unsere Entscheidung wird von den beiden Zielen geleitet werden, zum einen den freien Status aller von unserer freien Software abgeleiteten Datenwerke zu erhalten und zum anderen das gemeinschaftliche Nutzen und Wiederverwenden von Software im allgemeinen zu fördern.

A.3 Keine Gewährleistung

§ 11.

Da das Programm ohne jegliche Kosten lizenziert wird, besteht keinerlei Gewährleistung für das Programm, soweit dies gesetzlich zulässig ist. Sofern nicht anderweitig schriftlich bestätigt, stellen die Copyright-Inhaber und/oder Dritte das Programm so zur Verfügung, ,,wie es ist", ohne irgendeine Gewährleistung, weder ausdrücklich noch implizit, einschließlich - aber nicht begrenzt auf - Marktreife oder Verwendbarkeit für einen bestimmten Zweck. Das volle Risiko bezüglich Qualität und Leistungsfähigkeit des Programms liegt bei Ihnen. Sollte sich das Programm als fehlerhaft herausstellen, liegen die Kosten für notwendigen Service, Reparatur oder Korrektur bei Ihnen.

§ 12.

In keinem Fall, außer wenn durch geltendes Recht gefordert oder schriftlich zugesichert, ist irgendein Copyright-Inhaber oder irgendein Dritter, der das Programm wie oben erlaubt modifiziert oder verbreitet hat, Ihnen gegenüber für irgendwelche Schäden haftbar, einschließlich jeglicher allgemeiner oder spezieller Schäden, Schäden durch Seiteneffekte (Nebenwirkungen) oder Folgeschäden, die aus der Benutzung des Programms oder der Unbenutzbarkeit des Programms folgen (einschließlich - aber nicht beschränkt auf - Datenverluste, fehlerhafte Verarbeitung von Daten, Verluste, die von Ihnen oder anderen getragen werden müssen, oder dem Unvermögen des Programms, mit irgendeinem anderen Programm zusammenzuarbeiten), selbst wenn ein Copyright-Inhaber oder Dritter über die Möglichkeit solcher Schäden unterrichtet worden war.

Anhang B

Dateiformate

B.1 INI-Style Konfigurationsdateien

Konfigurationsdateien im INI-Style sind normale Textdateien, die aus Zeilen der der Form

`Schlüssel = Wert`

bestehen.

- Zwischen `Schlüssel`, dem Gleichzeichen und `Wert` dürfen Leerzeichen vorkommen.
- `Wert` kann eine Zahl oder eine Zeichenkette sein.
- Zeichenketten, die Leerzeichen enthalten, müssen durch Doublequotes eingeschlossen werden (also: `"We rt"`).
- Zeilen, die mit `#` beginnen, sind Kommentare.

B.2 Shellkompatible Konfigurationsdateien

Shellkompatible Konfigurationsdateien müssen von der `bash` ausgewertet werden können. Sie sind ebenfalls normale Textdateien, die aus Zeilen der Form

`Schlüssel=Wert`

bestehen.

- Zwischen `Schlüssel`, dem Gleichzeichen und `Wert` dürfen keine Leerzeichen vorkommen.
- `Wert` kann eine Zahl oder eine Zeichenkette sein.
- Zeichenketten, die Leerzeichen enthalten, müssen durch Doublequotes eingeschlossen werden (also: `"We rt"`).
- Zeilen, die mit `#` beginnen, sind Kommentare.

Literaturverzeichnis

[1] American National Standards Institute: *Information Technology - Serial Storage Achritecture - SCSI-2-Protokoll*, 1996. ANSI X3.294-1996.

[2] Personal Computer Memory Card International Association: *Detailed Overview of the PC Card Standard.* http://www.pc-card.com/pccardstandard.htm, 1998.

[3] Intel Corp., Microsoft Corp. und NEC Corp.: *Universial serial Bus Specification 1.1.* http://www.usb.org, 1998.

[4] Institute of Electrical and Electronics Engineers: *Standrad for a High Performance Serial Bus*, 1995. IEEE Std 1394-1995.

[5] Institute of Electrical and Electronics Engineers: *Carrier Sence Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifikations*, 1985. IEEE Std 802.3-1985.

[6] Carasso, C., J. Peek und J. Sinjou: *The Compact Disc digital Audio System.* Philips Tech. Rev., 1982. Vol.40, No.6, Nov. 1982, pp. 151.

[7] Fujitani, Larry: *Laser Optical Disk: The Coming Revolution in On-Line Storage.* Communications of the ACM, 1984. Vol.27, No.6, pp.546-554.

[8] c't, Magazin für Computertechnik: *200000 Seiten auf einer Scheibe, CD-ROMs auf dem Vormarsch.* c't 3/87, S.24, 1987. Messebericht von der CeBIT über verfügbare CD-ROM-Anwendungen und CD-ROM-Laufwerke.

[9] Gravesteijn, D.: *Materials developments for write.once and erasable phase-change optical recording.* Appl. Opt. 27, pp. 736-738, 1988.

[10] The Orange Forum Office. http://www.orangeforum.or.jp/e/index.htm, 2001.

[11] Optical Storage Technology Association. http://www.osta.org.

[12] Kozlovsky, W.: *Optical Recording in the Blue Using a Frequency Doubled Diode Laser.* SPIE Proceedings, 1992.

[13] Kurt, S.: *The Physics of Optical Recording.* Springer-Verlag Berlin, 1993.

[14] Otha, T., K. Yosihioka, H. Isomura und T. Akiyama: *High sensitivity overwritable phase-change optical disk for PC systems.* in Optical Data Storage '95, 1995. R. Knight, H. Ooki and S.-T. Tyan, eds., Proc. SPIE 2514, pp. 302-311.

[15] Yokota, C., T. Sasakawa und H. Hyakutake: *Phthalocyanine CDR for high Speed Recording.* Proc. SPIE-Int. Soc. Opt. Eng. (USA), Vol.2514, pp. 249-257, 1995.

[16] Kweon, Seok-Kyu: *Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing.* IEEE Real-Time Technology and Applications Symposium, 2000.

[17] PLEXTOR INC. http://www.plextor.com/english/news/press/pr06192000.html, 2001.

[18] POSTEL, J. und J. REYNOLDS: *File Transfer Protocol (FTP)*. RFC959, 1985.

[19] *GNU Library General Public License*. http://www.gnu.org/copyleft/gpl.html, http://www.gnu.de/gpl-ger.html, 2000.

[20] POSTEL, J. und J. REYNOLDS: *FILE TRANSFER PROTOCOL (FTP)*. http://www.ietf.org/rfc/rfc959.txt, 1985.

[21] NCSA: *The Common Gateway Interface - NCSA-Spezifikation und Dokumentation*. http://hoohoo.ncsa.uiuc.edu/cgi/overview.html.

[22] THE PHP GROUP: *PHP*. http://www.php.net.

[23] FACHBEREICH INFORMATIK, UNIVERSITÄT DORTMUND: *PG-Ordnung*. http://ls4-www.cs.uni-dortmund.de/PGB/alles/node4.html, 2002.