(12) NACH DEM VERTRAG ÜBER DIE INTERNATIONALE ZUSAMMENARBEIT AUF DEM GEBIET DES PATENTWESENS (PCT) VERÖFFENTLICHTE INTERNATIONALE ANMELDUNG

(19) Weltorganisation für geistiges Eigentum
Internationales Büro

(43) Internationales Veröffentlichungsdatum
3. November 2005 (03.11.2005)

PCT

(10) Internationale Veröffentlichungsnummer
WO 2005/104018 A2

(51) Internationale Patentklassifikation[7]: G06K 19/00

(21) Internationales Aktenzeichen: PCT/EP2005/004182

(22) Internationales Anmeldedatum:
19. April 2005 (19.04.2005)

(25) Einreichungssprache: Deutsch

(26) Veröffentlichungssprache: Deutsch

(30) Angaben zur Priorität:
10 2004 019 683.4 22. April 2004 (22.04.2004) DE

(71) Anmelder *(für alle Bestimmungsstaaten mit Ausnahme von US)*: **GIESECKE & DEVRIENT GMBH** [DE/DE]; Prinzregentenstrasse 159, 81677 München (DE).

(72) Erfinder; und

(75) Erfinder/Anmelder *(nur für US)*: **HOCKAUF, Robert** [DE/DE]; Eggenfeldenerstrasse 127, 81929 München (DE). **ULBRICHT, Thorsten** [DE/DE]; Rothschwaigestrasse 39a, 80997 München (DE). **SCHUBERT, Rudolf** [DE/DE]; Gebr.-Batscheiderstrasse 13, 82041 Oberhaching (DE).

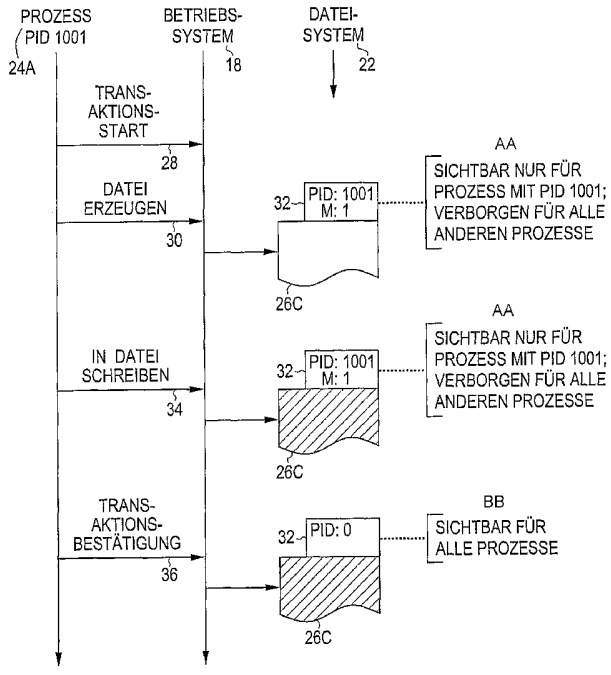
(74) Anwalt: **DENDORFER, Claus**; Wächtershäuser & Hartz, Weinstrasse 8, 80333 München (DE).

(81) Bestimmungsstaaten *(soweit nicht anders angegeben, für jede verfügbare nationale Schutzrechtsart)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA,

[Fortsetzung auf der nächsten Seite]

(54) **Title:** MANAGING A FILE SYSTEM IN A PORTABLE DATA CARRIER

(54) **Bezeichnung:** VERWALTEN EINES DATEISYSTEMS IN EINEM TRAGBAREN DATENTRÄGER



24A PID 1001 PROCESS
18 OPERATING SYSTEM
22 FILE SYSTEM
28 START TRANSACTION
30 CREATE FILE
34 WRITE IN FILE
36 CONFIRM TRANSACTION
AA VISIBLE ONLY FOR PROCESS WITH PID 1001; HIDDEN FOR ALL OTHER PROCESSES
BB VISIBLE FOR ALL PROCESSES

(57) Abstract: Disclosed is a method for managing a file system (22) in a portable data carrier. According to said method, process-specific visibility data (32) is managed for at least some structures (26x) in the file system (22) such that an event during which a process (24A) creates or deletes a structure (26x) in or from the file system (22) remains hidden for other concurrently executed processes at least until said event has been successfully completed. A portable data carrier and a computer program product are provided with corresponding characteristics. The invention allows at least some flawed sequences that occur when creating and/or deleting structures (26x) in/from the file system (22) to be prevented with little effort in terms of resources.

(57) Zusammenfassung: Bei einem Verfahren zum Verwalten eines Dateisystems (22) in einem tragbaren Datenträger werden prozeßspezifische Sichtbarkeitsinformationen (32) für zumindest manche Strukturen (26x) im Dateisystem (22) verwaltet, um zu bewirken, daß ein Vorgang, bei dem ein Prozeß (24A) eine Struktur (26x) im Dateisystem (22) erzeugt oder aus dem Dateisystem (22) löscht, für andere nebenläufig ausgeführte Prozesse zumindest bis zum erfolgreichen Abschluß dieses Vorgangs verborgen bleibt. Ein tragbarer Datenträger und ein Computerprogrammprodukt weisen entsprechende Merkmale auf. Durch die Erfindung lassen sich zumindest manche Fehlabläufe beim Erzeugen von Strukturen (26x) im Dateisystem (22) und/ oder beim Löschen von Strukturen (26x) aus dem Dateisystem (22) mit geringem Ressourcenauf wand vermeiden.

WO 2005/104018 A2

MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Bestimmungsstaaten *(soweit nicht anders angegeben, für jede verfügbare regionale Schutzrechtsart)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), eurasisches (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), europäisches (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Erklärung gemäß Regel 4.17:

— *hinsichtlich der Berechtigung des Anmelders, ein Patent zu beantragen und zu erhalten (Regel 4.17 Ziffer ii) für die folgenden Bestimmungsstaaten AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW, ARIPO Patent (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), eurasisches Patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), europäisches Patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI Patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG)*

Veröffentlicht:

— *ohne internationalen Recherchenbericht und erneut zu veröffentlichen nach Erhalt des Berichts*

Zur Erklärung der Zweibuchstaben-Codes und der anderen Abkürzungen wird auf die Erklärungen ("Guidance Notes on Codes and Abbreviations") am Anfang jeder regulären Ausgabe der PCT-Gazette verwiesen.

Verwalten eines Dateisystems in einem tragbaren Datenträger

Die Erfindung betrifft allgemein das technische Gebiet des Verwaltens eines Dateisystems in einem tragbaren Datenträger, wobei der Datenträger ein Betriebssystem aufweist, das nebenläufige - also parallele oder quasi-parallele - Prozesse unterstützt. Ein tragbarer Datenträger im Sinne des vorliegenden Dokuments kann insbesondere eine Chipkarte (*smart card*) in diversen Bauformen oder ein Chipmodul oder ein sonstiges ressourcenbeschränktes System sein.

Tragbare Datenträger werden mit immer mehr Speicherplatz und immer größerer Rechenleistung hergestellt. In einem internen Forschungsprojekt der Giesecke & Devrient GmbH wird gegenwärtig untersucht, inwieweit ein multitaskingfähiges Betriebssystem in einem modernen tragbaren Datenträger implementiert werden kann. In diesem Zusammenhang ist insbesondere an die Implementierung eines UNIX®-artigen Betriebssystems, z.B. des unter der Marke Linux® bekannten Betriebssystems, gedacht. Es ergeben sich jedoch erhebliche Probleme aus der Tatsache, daß der Ressourcenbedarf einer typischen Linux-Implementierung die Speicher- und Rechenkapazität heutiger tragbarer Datenträger voll ausschöpft oder sogar übersteigt. Aus diesem Grund ist der Einsatz ressourcenschonender Verfahren von entscheidender Wichtigkeit.

Ferner werden bei tragbaren Datenträgern in der Regel hohe Anforderungen an die Sicherheit und Zuverlässigkeit der Hardware, des Betriebssystems und der ausgeführten Anwendungsprogramme gestellt. So müssen z.B. Zugriffskonflikte und Inkonsistenzen, die auftreten können, wenn mehrere Prozesse einer multitaskingfähigen Chipkarte unabhängig voneinander auf ein gemeinsam genutztes Dateisystem zugreifen, sicher vermieden werden. Es muß dafür gesorgt werden, daß Änderungen im Dateisystem immer

atomar, d.h. entweder ganz oder gar nicht, erfolgen. Für Schreibzugriffe auf eine vorhandene Struktur des Dateisystems ist es in diesem Zusammenhang bekannt, die Struktur - eine Datei oder ein Verzeichnis - für die Dauer des Schreibzugriffs gegen Zugriffe der übrigen Prozesse zu sperren.

Eine besondere Situation ergibt sich jedoch, wenn eine Struktur im Dateisystem, auf die andere Prozesse möglicherweise zugreifen, neu angelegt oder gelöscht wird. So könnte z.B. bei einer GSM-Karte von einem Prozeß ein Telefonbuch angelegt werden und von einem zweiten Prozeß überprüft werden, ob ein Telefonbuch vorhanden ist. Wenn das Anlegen des Telefonbuchs fehlschlägt - z.B. wegen einer plötzlichen Unterbrechung der Energieversorgung des Datenträgers -, dann muß wegen der Forderung nach einer atomaren Ausführung des Vorgangs die angelegte Datei rückstandsfrei entfernt werden. Wenn nun der zweite Prozeß zufällig während der kurzzeitigen Existenz der Datei eine Dateiabfrage durchgeführt hat, könnte er zu dem fehlerhaften Ergebnis kommen, daß ein Telefonbuch vorliegt.

Es ergibt sich damit das Problem, Fehlabläufe der gerade beschriebenen Art bei der Ausführung nebenläufiger Prozesse in einem tragbaren Datenträger zuverlässig zu vermeiden. Eine entsprechende Problematik stellt sich beim Löschen einer Struktur im Dateisystem.

Das US-Patent 6,220,510 offenbart eine Chipkarte, die mehrere Anwendungsprogramme auszuführen vermag. Jedem Anwendungsprogramm sind je ein statischer und ein dynamischer Speicherbereich zugeordnet, die gegenüber den anderen Anwendungsprogrammen abgeschirmt sind. Aufgaben können über einen Befehls/Antwort-Mechanismus von einem ersten Anwendungsprogramm an ein zweites Anwendungsprogramm delegiert werden; hierbei

werden die Befehle und Antworten in einen öffentlichen Datenspeicherbereich geschrieben.

Die Erfindung hat die Aufgabe, einen Mechanismus zum Verwalten eines Dateisystems in einem tragbaren Datenträger bereitzustellen, durch den sich zumindest manche Fehlabläufe beim Erzeugen von Strukturen im Dateisystem und/oder Löschen von Strukturen aus dem Dateisystem mit geringem Ressourcenaufwand vermeiden lassen. Insbesondere sollen Inkonsistenzen der oben genannten Art, die durch Zugriffe nebenläufiger Prozesse auf das Dateisystem auftreten können, verhindert werden.

Erfindungsgemäß wird diese Aufgabe ganz oder zum Teil gelöst durch ein Verfahren gemäß Anspruch 1, einen tragbaren Datenträger gemäß Anspruch 14 und ein Computerprogrammprodukt gemäß Anspruch 15. Die abhängigen Ansprüche betreffen bevorzugte Ausgestaltungen der Erfindung.

Die Erfindung geht von der Grundidee aus, für zumindest manche Strukturen im Dateisystem prozeßspezifische Sichtbarkeitsinformationen zu verwalten, um zu bewirken, daß ein Vorgang, bei dem ein Prozeß eine Struktur im Dateisystem erzeugt oder aus dem Dateisystem löscht, für die anderen Prozesse zumindest bis zum erfolgreichen Abschluß dieses Vorgangs verborgen bleibt. Für die anderen Prozesse wird daher z.B. eine neu angelegte Datei erst dann sichtbar, wenn der vollständige Vorgang des Anlegens der Datei - gegebenenfalls einschließlich des Speicherns von Daten in die Datei - erfolgreich abgeschlossen worden ist, also wenn keine Möglichkeit eines Abbruchs oder Fehlschlags dieses Vorgangs mehr besteht. Fehler durch Zugriffskonflikte können somit zuverlässig vermieden werden.

Ein besonderer Vorteil der Erfindung besteht darin, daß – bei geeigneter Implementierung – für die Sichtbarkeitsinformationen nur wenig Speicher benötigt wird. Ferner sind keine aufwendigen Zusatzoperationen – z.B. das Anlegen umfangreicher Sicherungskopien in einem Schattenspeicher oder Rückführpuffer (*rollback buffer*) – erforderlich.

In bevorzugten Ausgestaltungen der Erfindung enthalten die Sichtbarkeitsinformationen zumindest für diejenigen Strukturen des Dateisystems, die gerade angelegt werden oder bei denen der Löschvorgang noch nicht abgeschlossen ist, einen Bezeichner des für den Erzeugungs- oder Löschvorgang zuständigen Prozesses und eine Marke (*flag*), die angibt, ob es sich um einen Erzeugungs- oder um einen Löschvorgang handelt. In anderen Ausführungsformen können die Sichtbarkeitsinformationen aus einer Marke bestehen, die anzeigt, ob bei einem Zugriffsversuch auf die Struktur, der die Sichtbarkeitsinformationen zugeordnet sind, zunächst eine Sichtbarkeitsüberprüfung erfolgen soll oder nicht. Wenn eine Sichtbarkeitsüberprüfung durchzuführen ist, kann in diesen Ausgestaltungen ein Anhang vorgesehen sein, der angibt, für welchen Prozeß die jeweilige Struktur sichtbar bzw. verborgen ist.

In manchen Ausgestaltungen ist vorgesehen, daß Strukturen, denen keine oder keine gültigen Sichtbarkeitsinformationen zugeordnet sind, für alle Prozesse sichtbar sein sollen.

In bevorzugten Ausführungsformen der Erfindung werden beim Erzeugen einer neuen Struktur im Dateisystem die Sichtbarkeitsinformationen für diese Struktur so eingestellt, daß die Struktur nur für den erzeugenden Prozeß sichtbar wird. Erst nach einem erfolgreichen Abschluß des Erzeugungsvorgangs – z.B. nach Bestätigung durch einen *Commit*-Befehl – werden die

Sichtbarkeitsinformationen so eingestellt, daß die Struktur allgemein sichtbar wird. Je nach der Art und Bedeutung der Sichtbarkeitsinformationen kann diese Einstellung z.B. dadurch erfolgen, daß die Sichtbarkeitsinformationen gelöscht werden oder ein in ihnen enthaltener Prozeßbezeichner auf einen ungültigen Wert gesetzt wird. Bei einem Abbruch des Vorgangs wird die erzeugte Struktur, die für die anderen Prozesse nie sichtbar war, aus dem Dateisystem gelöscht.

Zum Löschen einer Struktur aus dem Dateisystem werden in bevorzugten Ausgestaltungen zunächst nur die Sichtbarkeitsinformationen so eingestellt, daß die Struktur für den die Löschung anfordernden Prozeß verborgen wird, jedoch für alle anderen Prozesse sichtbar bleibt. Erst wenn feststeht, daß die Löschung unwiderruflich durchgeführt werden soll, wird die Struktur tatsächlich aus dem Dateisystem gelöscht. Bei einem Abbruch des Löschvorgangs - wenn also die Struktur auch für den die Löschung anfordernden Prozeß wieder sichtbar werden soll - werden die Sichtbarkeitsinformationen entsprechend eingestellt. Dies kann in manchen Ausgestaltungen dadurch erfolgen, daß die Sichtbarkeitsinformationen gelöscht oder auf einen ungültigen Wert gesetzt werden.

Der das Erzeugen oder Löschen der Struktur beinhaltende Vorgang ist in bevorzugten Ausgestaltungen ein atomarer Vorgang, der entweder vollständig ausgeführt oder rückstandsfrei abgebrochen wird. Der Abbruch kann z.B. durch einen auftretenden Fehler oder Spannungsausfall oder durch einen *Abort*-Befehl ausgelöst werden. Der erfolgreiche Abschluß erfordert in manchen Ausführungsformen eine Bestätigung durch einen *Commit*-Befehl, während in anderen Ausführungsformen der Vorgang immer dann erfolgreich abgeschlossen wird, wenn kein Fehler auftritt. Der Vorgang kann z.B. eine atomare Transaktion oder ein atomarer Vorgang sein, bei dem zunächst

eine Datei neu angelegt wird und dann Daten in dieser Datei gespeichert werden.

Allgemein ist die Erfindung im Zusammenhang mit allen Strukturen im Dateisystem einsetzbar, die von nebenläufigen Prozessen - dies können Prozesse des Betriebssystems und/oder eines Anwendungsprogramms sein - angelegt und/oder gelöscht werden können. Solche Strukturen können z.B. Dateien oder Verzeichnisse sein. Es sind auch Ausgestaltungen vorgesehen, in denen nur Dateien als Strukturen im Sinne der Erfindung angesehen werden. Dies schließt Ausgestaltungen ein, bei denen Verzeichnisse im Dateisystem als besondere Dateiart ausgebildet sind.

Das erfindungsgemäße Computerprogrammprodukt kann ein körperliches Medium mit gespeicherten Programmbefehlen sein, beispielsweise ein Halbleiterspeicher oder eine Diskette oder eine CD-ROM. Das Computerprogrammprodukt kann jedoch auch ein nicht-körperliches Medium sein, beispielsweise ein über ein Computernetzwerk übermitteltes Signal. In bevorzugten Ausgestaltungen weisen der Datenträger und/oder das Computerprogrammprodukt Merkmale auf, die den oben beschriebenen und/oder den in den abhängigen Verfahrensansprüchen genannten Merkmalen entsprechen.

Weitere Merkmale, Vorteile und Aufgaben der Erfindung gehen aus der folgenden genauen Beschreibung eines Ausführungsbeispiels und mehrerer Ausführungsalternativen hervor. Es wird auf die schematischen Zeichnungen verwiesen, in denen zeigen:

Fig. 1 ein Blockdiagramm mit Funktionseinheiten eines Datenträgers nach einem Ausführungsbeispiel der Erfindung,

Fig. 2 eine Darstellung des Zugriffs von Prozessen auf ein Dateisystem beim Betrieb des Datenträgers von Fig. 1,

Fig. 3 eine Ablaufdarstellung des Erzeugens einer Datei in dem Datenträger von Fig. 1, und

Fig. 4 eine Ablaufdarstellung des Löschens einer Datei in dem Datenträger von Fig. 1.

Der in Fig. 1 dargestellte Datenträger 10 weist auf einem einzigen Halbleiterchip einen Prozessor 12, einen Speicher 14 und eine Schnittstellenschaltung 16 zur kontaktlosen oder kontaktgebundenen Kommunikation mit einem externen Terminal (in Fig. 1 nicht gezeigt) auf. Der Speicher 14 ist in an sich bekannter Weise in mehrere in unterschiedlichen Technologien ausgestaltete Speicherfelder – im vorliegenden Ausführungsbeispiel RAM, ROM und EEPROM – unterteilt.

Im Speicher 14 befindet sich Programmcode, der ein Betriebssystem 18 implementiert. Das Betriebssystem 18 ist im vorliegenden Ausführungsbeispiel eine auf den Einsatz im Datenträger 10 zugeschnittene Variante des unter der Marke Linux bekannten Betriebssystems. Weiter enthält der Speicher 14 mindestens ein Anwendungsprogramm 20 sowie ein Dateisystem 22, das Datei- und Verzeichnisstrukturen in einer baumartigen Anordnung aufweist.

Wie in Fig. 2 gezeigt, laufen beim Betrieb des Datenträgers 10 mehrere Prozesse 24A, 24B, 24C, ... – im folgenden zusammenfassend mit 24x bezeichnet – quasi-parallel ab. Die Prozesse 24x können Systemprozesse des Betriebssystems 18 und/oder Benutzerprozesse des Anwendungsprogramms 20

sein. Jeder der Prozesse 24x weist einen eindeutigen Prozeßbezeichner PID (*process identifier*) auf; in Fig. 2 sind beispielhaft die Prozeßbezeichner "1001", "1002" und "1003" gezeigt. Das Betriebssystem 18 steuert und koordiniert den nebenläufigen Ablauf der Prozesse 24x. Ferner vermögen die Prozesse 24x über das Betriebssystem 18 auf das Dateisystem 22 zuzugreifen. In Fig. 2 sind als Strukturen des Dateisystems 22 beispielhaft zwei Verzeichnisse 26A, 26B und zwei Dateien 26C, 26D gezeigt; diese und weitere im Dateisystem 22 enthaltene Strukturen werden im folgenden zusammenfassend mit 26x bezeichnet.

Es besteht nun allgemein das Problem, daß bei gleichzeitigen oder fast gleichzeitigen Operationen, die die Prozesse 24x in dem gemeinsam genutzten Dateisystem 22 ausführen, temporäre Inkonsistenzen auftreten können. Dies kann insbesondere dann der Fall sein, wenn ein Prozeß 24x mehrere solche Operationen in einer atomaren Transaktion ausführt. Durch die im folgenden beschriebenen Verfahren wird dieses Problem bei den Operationen des Erzeugens und Löschens von Strukturen 26x im Dateisystem 22 vermieden.

Fig. 3 stellt einen beispielhaften Ablauf dar, bei dem der Prozeß 24A - Prozeßbezeichner "1001" - in einer erfolgreichen Transaktion eine neue Datei - hier beispielhaft die Datei 26C - anlegt und Daten in diese Datei schreibt. Die senkrechten Pfeile in Fig. 3 geben den zeitlichen Ablauf des Vorgangs an; in der mit "Dateisystem" überschriebenen Spalte ist die Abfolge unterschiedlicher Zustände des Dateisystems 22 während des Vorgangs veranschaulicht.

Der Ablauf gemäß Fig. 3 beginnt mit dem Starten einer neuen Transaktion durch einen Befehl 28, den der Prozeß 24A dem Betriebssystem 18 übermittelt. Weiter gibt der Prozeß 24A einen Befehl 30 zum Erzeugen einer neuen

Datei an das Betriebssystem 18 aus. Das Betriebssystem 18 legt daraufhin die neue Datei 26C im Dateisystem 22 an. Hierbei werden der Datei 26C Sichtbarkeitsinformationen 32 zugeordnet, die den Bezeichner PID des befehlsgebenden Prozesses - im vorliegenden Fall den Bezeichner "1001" des Prozesses 24A - und eine Marke M (*flag*) enthalten. Die Marke M gibt einen Sichtbarkeitsmodus für die Datei 26C an.

Der hier beim Anlegen einer Datei verwendete Sichtbarkeitsmodus "1" besagt allgemein, daß der in den Sichtbarkeitsinformationen 32 enthaltene Prozeßbezeichner PID denjenigen Prozeß angibt, für den die Datei sichtbar sein soll. Für alle anderen Prozesse soll die Datei verborgen sein. Ein Sichtbarkeitsmodus "0" würde dagegen aussagen, daß der in den Sichtbarkeitsinformationen 32 enthaltene Prozeßbezeichner PID denjenigen - einzigen - Prozeß angibt, für den die Datei verborgen sein soll. Dieser Sichtbarkeitsmodus "0" wird im Zusammenhang mit dem noch zu beschreibenden Löschen einer Datei verwendet.

Im Beispiel von Fig. 3 ist also die nach der Ausführung des Befehls 30 erzeugte Datei 26C nur für den Prozeß 24A mit dem Prozeßbezeichner "1001" sichtbar und für alle anderen Prozesse 24B, 24C, ... verborgen. Wenn ein solcher anderer Prozeß 24B, 24C, ... zum jetzigen Zeitpunkt eine Aufstellung der im Dateisystem 22 enthaltenen Strukturen 26x anfordern würde, dann würde die Datei 26C nicht darin enthalten sein. Ebenso würde der andere Prozeß 24B, 24C, ... bei einem Versuch, auf die Datei 26C zuzugreifen, eine Fehlermeldung wegen einer nicht existierenden Datei erhalten.

In einem folgenden Befehl 34 schreibt der Prozeß 24A Daten in die neu angelegte Datei 26C. Die Sichtbarkeitsinformationen 32 bleiben dabei unverän-

dert. Die Datei 26C ist daher nach wie vor nur für den Prozeß 24A sichtbar und für alle anderen Prozesse 24B, 24C, ... verborgen.

Wenn der Prozeß 24A die Transaktion mit einem *Commit*-Befehl 36 bestätigt, wird die erfolgreiche Beendigung der Transaktion dadurch angezeigt, daß der in den Sichtbarkeitsinformationen 32 enthaltene Prozeßbezeichner PID auf einen ungültigen Wert - z.B. den Wert "0" - gesetzt wird. Die Sichtbarkeitsinformationen 32 sind somit insgesamt ungültig, was besagt, daß keine Einschränkungen hinsichtlich der Sichtbarkeit der Datei 26C mehr bestehen. Alle Prozesse 24x können nun die Datei 26C sehen und uneingeschränkt auf sie zugreifen.

Wird die Transaktion nicht bestätigt, sondern - z.B. in Reaktion auf einen *Abort*-Befehl oder aufgrund eines Fehlers - abgebrochen, so wird die Datei 26C aus dem Dateisystem 22 gelöscht. Dies kann unmittelbar in Reaktion auf den *Abort*-Befehl oder beim nächsten Hochfahren des Datenträgers 10 oder zu einem anderen geeigneten Zeitpunkt erfolgen. Bis auf den die Transaktion ausführenden Prozeß 24A hat in diesem Fall kein anderer Prozeß 24B, 24C, ... zu irgendeinem Zeitpunkt Kenntnis von der temporär angelegten Datei 26C erlangen können.

In dem beispielhaften Ablauf von Fig. 3 sind vier voneinander getrennte Befehle 28, 30, 34, 36 gezeigt. Es versteht sich, daß diese Befehle in Ausführungsalternativen ganz oder teilweise miteinander kombiniert werden können. So können beispielsweise die Befehle 30 und 34 zu einem einzigen Befehl zusammengefaßt werden, der Daten in eine neu anzulegende Datei schreibt. In weiteren Ausführungsalternativen kann dieser Befehl - oder der in Fig. 3 gezeigte Befehl 30 - implizit den Beginn einer neuen Transaktion anzeigen, so daß der Befehl 28 entfallen kann. Ferner kann in manchen

Ausgestaltungen auf eine explizite Transaktionsbestätigung durch einen *Commit*-Befehl verzichtet werden.

Fig. 4 veranschaulicht den Ablauf eines Vorgangs, bei dem eine existierende Datei im Dateisystem 22 - hier beispielhaft die Datei 26D - gelöscht wird. Der Ausgangszustand für diesen Ablauf ist, daß die Datei 26D im Dateisystem 22 enthalten und für alle Prozesse 24x sichtbar ist. Dies kann z.B. durch fehlende oder ungültige Sichtbarkeitsinformationen 32 - im vorliegenden Beispiel durch einen ungültigen Prozeßbezeichner PID mit dem Wert "0" - angezeigt werden.

Mit den Befehlen 38 und 40 fordert der Prozeß 24A vom Betriebssystem 18 den Start einer neuen Transaktion und das Löschen der Datei 26D an. Das Betriebssystem 18 löscht die Datei 26D zu diesem Zeitpunkt jedoch noch nicht aus dem Dateisystem 22, sondern verbirgt sie nur vor dem die Löschung anfordernden Prozeß 24A. Hierzu werden in die Sichtbarkeitsinformationen 32 der Wert "1001" als Prozeßbezeichner PID des die Löschung anfordernden Prozesses 24A und der Wert "0" als Sichtbarkeitsmodus M eingetragen; die Bedeutung dieses Werts für den Sichtbarkeitsmodus M wurde oben bereits erläutert. Die Datei 26D ist damit für alle anderen Prozesse 24B, 24C, ... nach wie vor sichtbar.

Das Betriebssystem 18 veranlaßt erst in Reaktion auf den Erhalt eines *Commit*-Befehls 42 zur Transaktionsbestätigung die tatsächliche Löschung der Datei 26D aus dem Dateisystem 22. Erst zu diesem Zeitpunkt können die anderen Prozesse 24B, 24C, ... Kenntnis von der - nun erfolgreich abgeschlossenen - Transaktion erlangen.

Wird die Löschtransaktion nicht erfolgreich abgeschlossen, sondern abgebrochen, so verbleibt die Datei 26D im Dateisystem 22. Die Sichtbarkeitsinformationen 32 werden dann wieder so eingestellt - z.B. durch Setzen des Prozeßbezeichners PID auf den ungültigen Wert "0" - daß die Datei 26D für alle Prozesse 24x sichtbar ist. Bei einem Transaktionsabbruch durch einen *Abort*-Befehl erfolgt dies unmittelbar in Reaktion auf diesen Befehl; bei einem Transaktionsabbruch durch einen Spannungsausfall werden die Sichtbarkeitsinformationen 32 beim nächsten Neustart des Datenträgers 10 entsprechend zurückgesetzt. Insgesamt war in diesem Fall die Datei 26D während des gesamten Vorgangs für alle anderen Prozesse 24B, 24C, ... - also alle Prozesse außer dem den Löschbefehl 40 abgebenden Prozeß 24A - ununterbrochen sichtbar.

Patentansprüche

1. Verfahren zum Verwalten eines Dateisystems (22) in einem tragbaren Datenträger (10), wobei der tragbare Datenträger (10) ein Betriebssystem (18) aufweist, welches nebenläufige Prozesse (24x) unterstützt, die auf das Dateisystem (22) zuzugreifen vermögen, und wobei prozeßspezifische Sichtbarkeitsinformationen (32) für zumindest manche Strukturen (26x) im Dateisystem (22) verwaltet werden, um zu bewirken, daß ein Vorgang, bei dem ein Prozeß (24A) eine Struktur (26x) im Dateisystem (22) erzeugt oder aus dem Dateisystem (22) löscht, für die anderen Prozesse (24B, 24C) bis zum erfolgreichen Abschluß dieses Vorgangs verborgen bleibt.

2. Verfahren nach Anspruch 1, **dadurch gekennzeichnet, daß** der Prozeß (24A) bei dem Vorgang die Erzeugung einer neuen Struktur (26x) im Dateisystem (22) anfordert, und daß daraufhin die Struktur (26x) im Dateisystem (22) erzeugt wird, aber die Sichtbarkeitsinformationen (32) für diese Struktur (26x) so eingestellt werden, daß die Struktur (26x) nur für den erzeugenden Prozeß (24A) sichtbar wird.

3. Verfahren nach Anspruch 2, **dadurch gekennzeichnet, daß** nach dem erfolgreichen Abschluß des Vorgangs die Sichtbarkeitsinformationen (32) für die erzeugte Struktur (26x) so eingestellt werden, daß die Struktur (26x) für alle Prozesse (24x) sichtbar wird.

4. Verfahren nach Anspruch 2 oder Anspruch 3, **dadurch gekennzeichnet, daß** bei einem Abbruch des Vorgangs die Struktur (26x) aus dem Dateisystem (22) gelöscht wird.

5. Verfahren nach einem der Ansprüche 1 bis 4, **dadurch gekennzeichnet, daß** der Prozeß (24A) bei dem Vorgang das Löschen einer bestehenden Struktur (26x) aus dem Dateisystem (22) anfordert, und daß daraufhin die Sichtbarkeitsinformationen (32) für diese Struktur (26x) so eingestellt werden, daß die Struktur (26x) nur für den die Löschung anfordernden Prozeß (24A) verborgen wird, während die Struktur (26x) im Dateisystem (22) verbleibt.

6. Verfahren nach Anspruch 5, **dadurch gekennzeichnet, daß** nach dem erfolgreichen Abschluß des Vorgangs die Struktur (26x) aus dem Dateisystem (22) gelöscht wird.

7. Verfahren nach Anspruch 5 oder Anspruch 6, **dadurch gekennzeichnet, daß** bei einem Abbruch des Vorgangs die Sichtbarkeitsinformationen (32) für die Struktur (26x) so eingestellt werden, daß die Struktur (26x) wieder für alle Prozesse (24x) sichtbar wird.

8. Verfahren nach einem der Ansprüche 1 bis 7, **dadurch gekennzeichnet, daß** die Sichtbarkeitsinformationen (32) für zumindest manche Strukturen (26x) im Dateisystem (22) einen Bezeichner (PID) eines Prozesses (24A) und eine Information (M) darüber enthalten, ob die Struktur (26x) für diesen Prozeß (24A) sichtbar und für alle anderen Prozesse (24B, 24C) verborgen oder für diesen Prozeß (24A) verborgen und für alle anderen Prozesse (24B, 24C) sichtbar sein soll.

9. Verfahren nach einem der Ansprüche 1 bis 8, **dadurch gekennzeichnet, daß** Strukturen (26x) des Dateisystems (22), denen keine oder keine gültigen Sichtbarkeitsinformationen (32) zugeordnet sind, für alle Prozesse (24x) sichtbar sind.

10. Verfahren nach einem der Ansprüche 1 bis 9, **dadurch gekennzeichnet, daß** der Vorgang eine Transaktion ist, die entweder abgebrochen oder durch eine Bestätigung erfolgreich abgeschlossen werden kann.

11. Verfahren nach einem der Ansprüche 1 bis 10, **dadurch gekennzeichnet, daß** der Vorgang das Speichern von Daten in einer neu anzulegenden Datei (26C, 26D) ist.

12. Verfahren nach einem der Ansprüche 1 bis 11, **dadurch gekennzeichnet, daß** die Struktur eine Datei (26C, 26D) oder ein Verzeichnis (26A, 26B) im Dateisystem (22) ist.

13. Verfahren nach einem der Ansprüche 1 bis 12, **dadurch gekennzeichnet, daß** der Datenträger (10) ein UNIX-artiges Betriebssystem (22), insbesondere ein Linux-Betriebssystem aufweist.

14. Tragbarer Datenträger (10), insbesondere Chipkarte oder Chipmodul, mit einem Prozessor (12) und mindestens einem Speicher (14), wobei der Speicher (14) Programmbefehle enthält, die dazu eingerichtet sind, den Prozessor (12) zur Ausführung eines Verfahrens nach einem der Ansprüche 1 bis 13 zu veranlassen.

15. Computerprogrammprodukt, das maschinenlesbare Programmbefehle für einen Prozessor (12) eines tragbaren Datenträgers (10) aufweist, die dazu eingerichtet sind, den Prozessor (12) zur Ausführung eines Verfahrens nach einem der Ansprüche 1 bis 13 zu veranlassen.

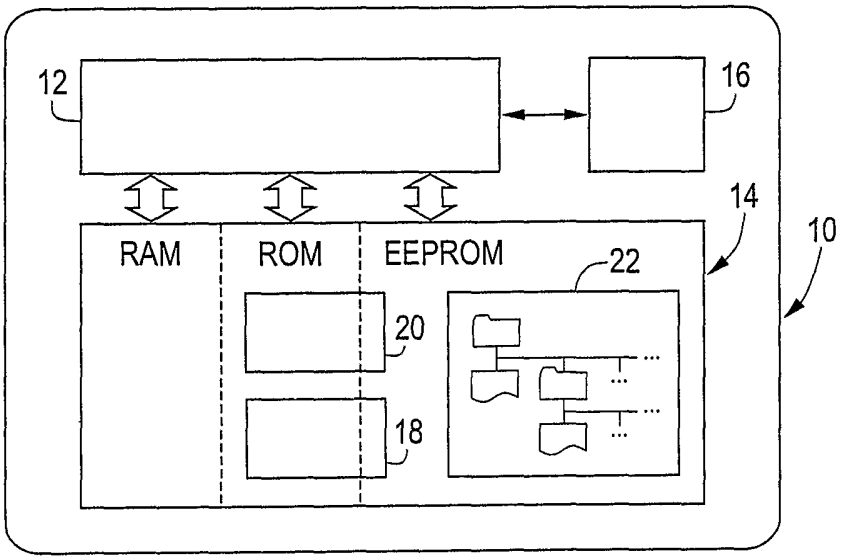

12
16
14
10
RAM
ROM
EEPROM
22
20
18

Fig. 1

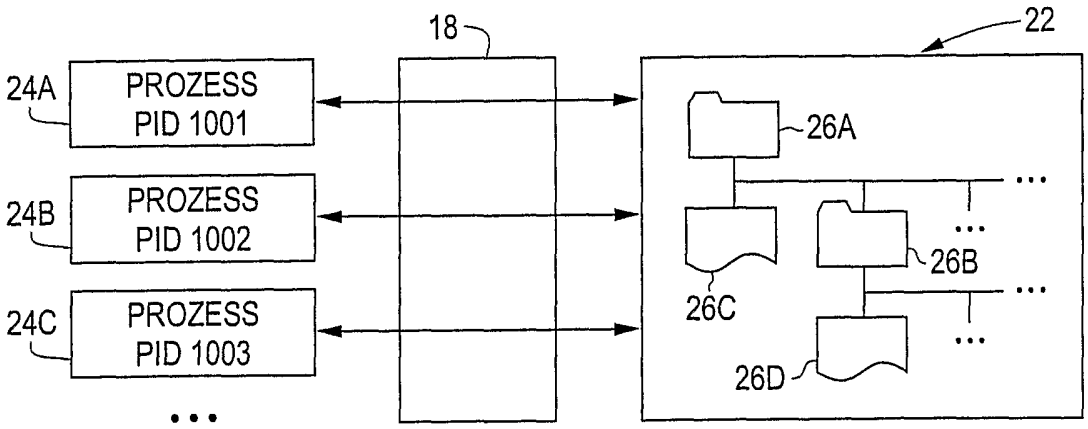

18
22
24A
PROZESS
PID 1001
24B
PROZESS
PID 1002
24C
PROZESS
PID 1003
26A
26B
26C
26D

Fig. 2

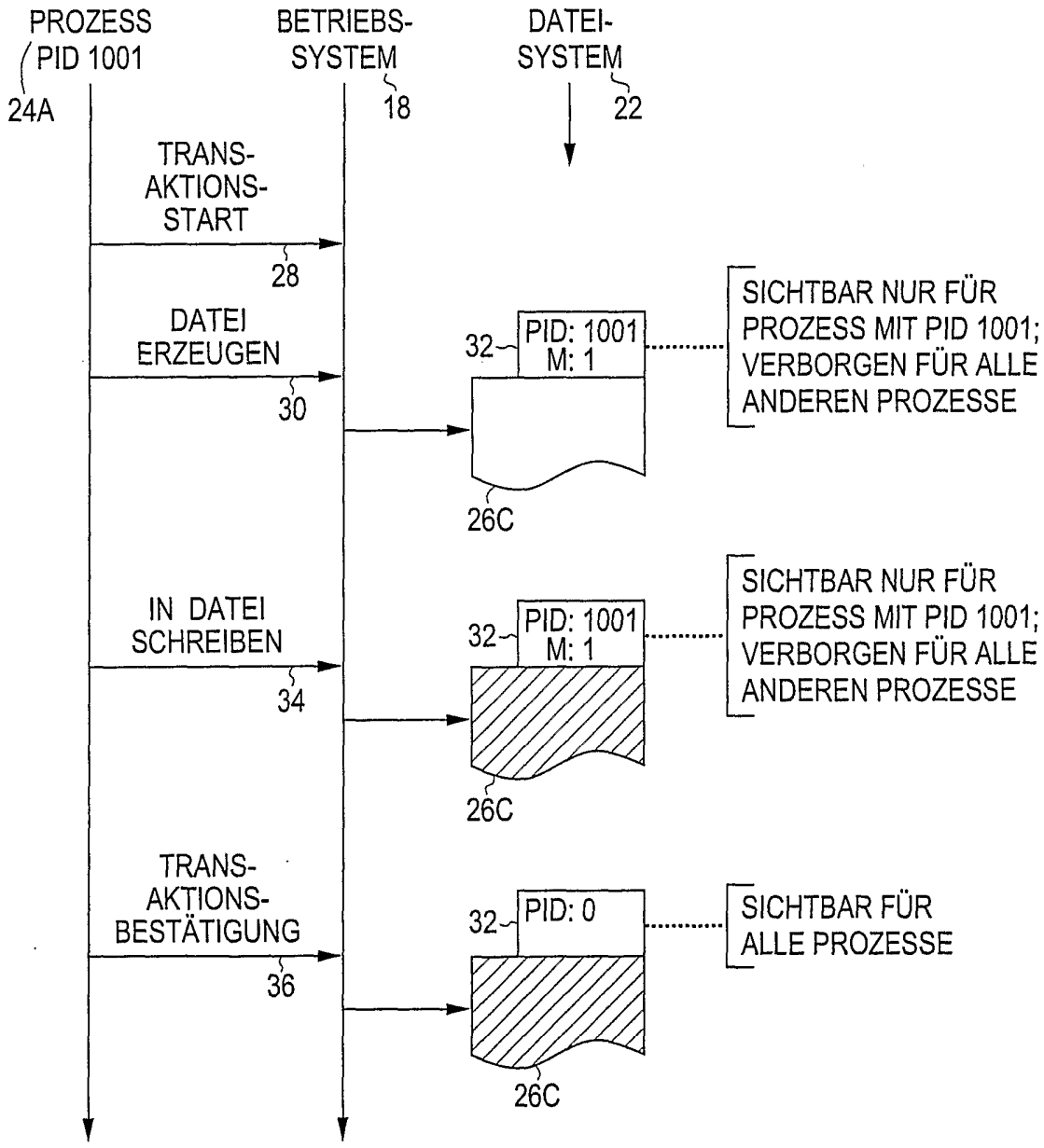

PROZESS
PID 1001
24A
BETRIEBS-
SYSTEM
18
DATEI-
SYSTEM
22
TRANS-
AKTIONS-
START
28
DATEI
ERZEUGEN
30
32
PID: 1001
M: 1
SICHTBAR NUR FÜR
PROZESS MIT PID 1001;
VERBORGEN FÜR ALLE
ANDEREN PROZESSE
26C
IN DATEI
SCHREIBEN
34
32
PID: 1001
M: 1
SICHTBAR NUR FÜR
PROZESS MIT PID 1001;
VERBORGEN FÜR ALLE
ANDEREN PROZESSE
26C
TRANS-
AKTIONS-
BESTÄTIGUNG
36
32
PID: 0
SICHTBAR FÜR
ALLE PROZESSE
26C

Fig. 3

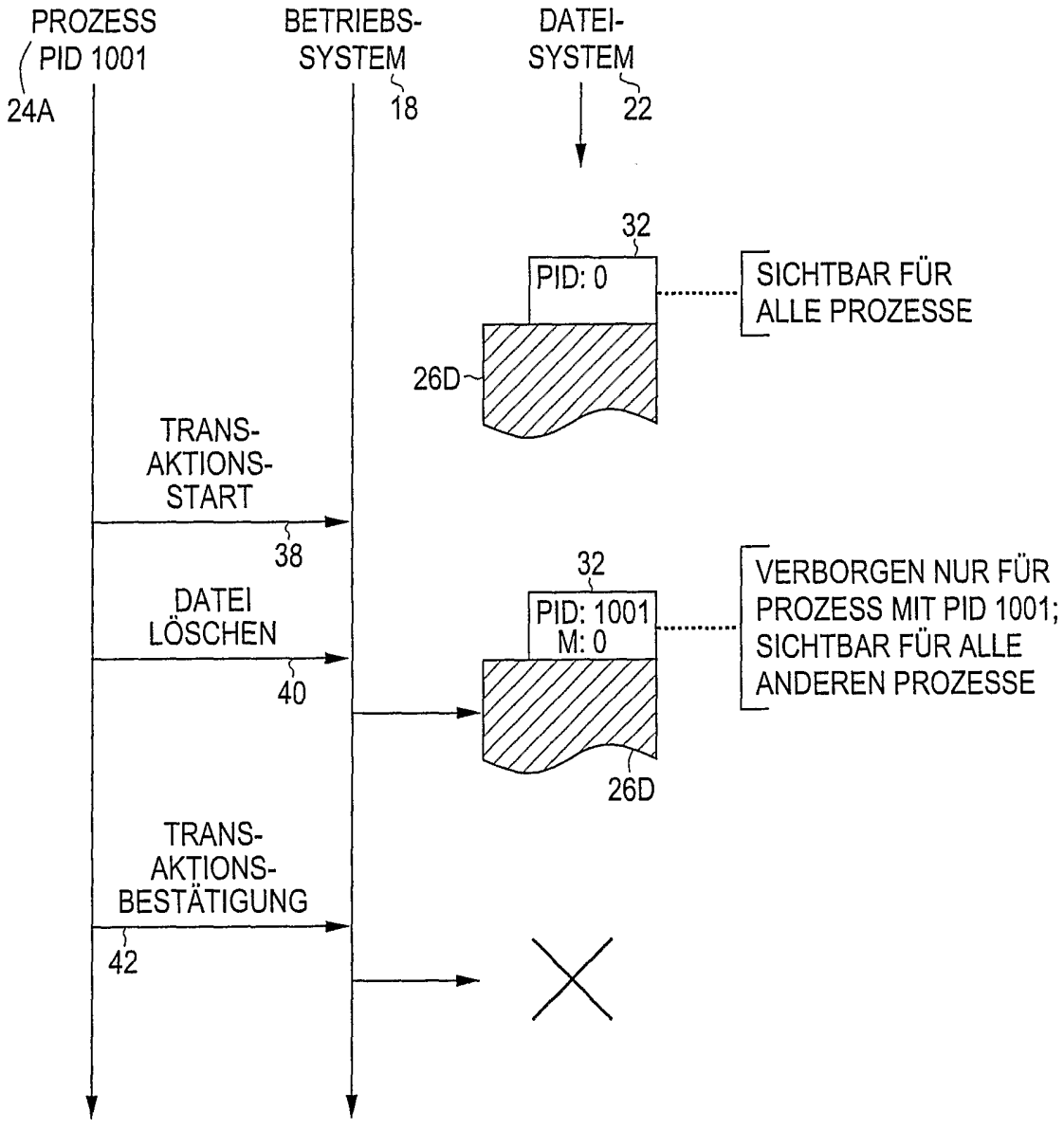

PROZESS
PID 1001
24A
BETRIEBS-
SYSTEM
18
DATEI-
SYSTEM
22
32
PID: 0
SICHTBAR FÜR
ALLE PROZESSE
26D
TRANS-
AKTIONS-
START
38
32
DATEI
LÖSCHEN
PID: 1001
M: 0
VERBORGEN NUR FÜR
PROZESS MIT PID 1001;
SICHTBAR FÜR ALLE
ANDEREN PROZESSE
40
26D
TRANS-
AKTIONS-
BESTÄTIGUNG
42

Fig. 4