J.E.

Industria y Comercio

SUPERINTENDENCIA

SOLICITUD
PATENTE DE INVENCIÓN

EXPEDIENTE N° 05-116723

TITULO: SISTEMA DE ARCHIVO EXTENSIBLE

CLASIFICACION INTERNACIONAL: G06F 17/60

SOLICITANTE: MICROSOFT CORPORATION

DOMICILIO: REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA

APODERADO: DARIO CARDENAS NAVAS

BOGOTÁ, D.C. _____

PETITORIO

Industria y Comercio
SUPERINTENDENCIA

SUPERINDUSTRIA Y COMERCIO Folios: 124
Radicacion : 05116723 00000000
Fecha (AMD): 2005-11-17 16:18:39
Tramite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia: 0020 DIVISION DE NUEVAS CREACIONES

FORMULARIO UN ...ITUD DE PATENTE
2000-3

1 SOLICITUD DE:

[X] Patente de Invención [ ] Patente de Modelo de Utilidad

			IDENTIFICACION
SOLICITANTE (71)	Nombre:	**MICROSOFT CORPORATION**	
	Dirección:	One Microsoft Way Redmond, Washington 98052, Estados Unidos de America	C.C. [ ] NIT [ ]
	Teléfono:	3123600 Fax: 3122410	C.E. [ ] Otro [X]
	E-Mail:	general@cardenasycardenas.com	
	Lugar de Constitución	Estado de Washington, Estados Unidos de América	Número
	Domicilio:	REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA.	

			IDENTIFICACION
REPRESENTANTE O APODERADO (74)	Nombre:	**DARIO CARDENAS NAVAS**	
	Dirección:	Carrera 7 No 71-52 Torre B Piso 9	C.C. [X] NIT [ ]
	Teléfono:	3123600 Fax: 3122410	C.E. [ ] Otro [ ]
	E-Mail:	general@cardenasycardenas.com	Número 17.066.629 BTA TP 1.250 C.S.J.

			IDENTIFICACION
4 INVENTOR(ES) (72)	Nombre:	1. Ravisankar V. Pudipeddi 2. Vishal V. Ghotge 3. Saroshi C. Havewala 4. Ravinder S. Thind 5. Mark J. Zbikowski	
	Dirección:	1. One Microsoft Way Redmond, Washington 98052, Estados Unidos de América 2. One Microsoft Way Redmond, Washington 98052, Estados Unidos de América 3. One Microsoft Way Redmond, Washington 98052, Estados Unidos de América 4. One Microsoft Way Redmond, Washington 98052, Estados Unidos de América 5. One Microsoft Way Redmond, Washington 98052, Estados Unidos de América	C.C. [ ] NIT [ ]
	Teléfono:	3123600 Fax: 3123600	C.E. [ ] Otro [X]
	E-Mail:	general@cardenasycardenas.com	
	Domicilio:	1. REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA. 2. REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA. 3. REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA. 4. REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA. 5. REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA.	

5 Título(54): G06F17/00

SISTEMA DE ARCHIVO EXTENSIBLE

6 Prioridad SI [X] NO [ ]

(33)País de Origen	(32)Fecha	(31)N° de Solicitud
Estados Unidos de América	Diciembre 17, 2004	60/637407
Estados Unidos de América	Septiembre 16, 2005	11/229485

Para publicar a partir de la fecha de la presente solicitud a los:
[ ] 6 meses [ ] 12 meses [ ] 18 meses [ ] Otro

Comprobante de Pago No.: 05- 87,441 / 05-87,445 / 05-87,444 **Fecha** Noviembre 15, 2005

AMR/JAG/amg

ANEXOS

- [X] Comprobante de pago de la tasa de presentación de la solicitud. (No. 05-87,441)
- [ ] Comprobante de pago de la tasa por concepto de excedente de palabras en la publicación.
- [X] Comprobante de pago de la tasa por concepto de reivindicación de prioridad. (No. 05-87,444/05-87,445)
- [X] Poderes, si fuere el caso. (Protocolo No 16.069)
- [X] Certificado de existencia y representación legal cuando el solicitante sea persona jurídica. Protocolo No 16.069)
- [ ] Copia certificada de la primera solicitud, si se reivindica prioridad.
- [X] Traducción simple de la primera solicitud, si se reivindica prioridad.
- [ ] Documento de cesión del inventor al solicitante o a su causante.
- [X] Descripción de la invención.
- [X] Una o más reivindicaciones.
- [X] Dibujos y/o planos necesarios.
- [ ] De ser el caso, copia del contrato de acceso.
- [ ] De ser el caso, documento que acredite la licencia o autorización de uso de conocimientos tradicionales de la comunidad de las comunidades indígenas.
- [ ] De ser el caso, certificado de depósito del material biológico.
- [X] Arte final 12x12 cm.
- [X] Resumen y extracto para publicación.

NOMBRE: DARIO CARDENAS NAVAS

FIRMA:

C.C. 17.086.629 BTA TP. 1.250 C.S.J.

3

SUPERINDUSTRIA Y COMERCIO
Radicacion : 05116723 00000000
Fecha (AMD): 2005-11-17 16:10:59 Folios: 124
Tramite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 05 - 87,441
FECHA : NOVIEMBRE 15 DE 2005

C O N S I G N A C I O N

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	050-00110-6	47049434	15/11/2005	7,735,000.00

C O N C E P T O

CANT.	RENTISTICO	CONCEPTO	TOTAL CONCEPTO
1	50005-01-01 SOLICITUDES	1 TRAMITES DE SOL. DE PATENTE DE IN	443.000.00
		TOTAL : $	443,000.00

SON: CUATROCIENTOS CUARENTA Y TRES MIL PESOS

CARDENAS & CARDENAS
ABOGADOS

RESPONSABLE : _____

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. SISTEMA DE ARCHIVO EXTENSIBLE

3

4

SUPERINDUSTRIA Y COMERCIO
Radicacion : 05116723 00000000 Folios: 124
Fecha (AMD): 2005-11-17 16:10:59
Tramite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 05 - 87,444
FECHA : NOVIEMBRE 15 DE 2005

***** C O N S I G N A C I O N *****

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	050-00110-6	47049434	15/11/2005	7,735,000.00

***** C O N C E P T O *****

CANT.	RENTISTICO	CONCEPTO		TOTAL CONCEPTO
1	50005-01-01 SOLICITUDES	82	INVOCACION DE UNA PRIORIDAD EN NU	124.000.00
			TOTAL : $	124,000.00

SON: CIENTO VEINTICUATRO MIL PESOS

CARDENAS & CARDENAS ABOGADOS

RESPONSABLE : _____

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. SISTEMA DE ARCHIVO EXTENSIBLE

4

5

SUPERINDUSTRIA Y COMERCIO
Radicacion : 05116723 00000000 Folios: 124
Fecha (AMD): 2005-11-17 16:10:59
Tramite : 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 05 - 87.445
FECHA : NOVIEMBRE 15 DE 2005

***** C O N S I G N A C I O N *****

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	050-00110-6	47049434	15/11/2005	7,735,000.00

***** C O N C E P T O *****

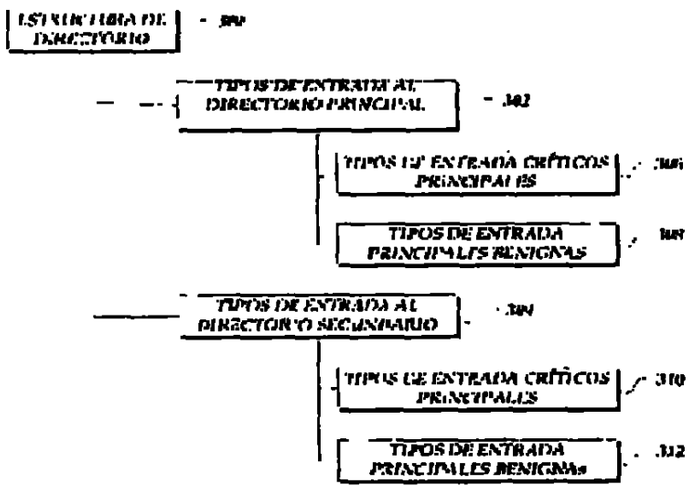
CANT.	RENTISTICO	CONCEPTO	TOTAL CONCEPTO
1	50005-01-01 SOLICITUDES	82 INVOCACION DE UNA PRIORIDAD EN NU	124,000.00
		TOTAL : $	124,000.00

SON: CIENTO VEINTICUATRO MIL PESOS

CARDENAS & CARDENAS ABOGADOS

RESPONSABLE : _____

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. SISTEMA DE ARCHIVO EXTENSIBLE.


ESTRUCTURA DE DIRECTORIO
TIPOS DE ENTRADA AL DIRECTORIO PRINCIPAL
TIPOS DE ENTRADA CRÍTICOS PRINCIPALES
TIPOS DE ENTRADA PRINCIPALES BENIGNAS
TIPOS DE ENTRADA AL DIRECTORIO SECUNDARIO
TIPOS DE ENTRADA CRÍTICOS PRINCIPALES
TIPOS DE ENTRADA PRINCIPALES BENIGNAs

MAU

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPL NO.	FILING OR 371 (c) DATE	ART UNIT	FIL FEE REC'D	ATTY.DOCKET NO	DRAWINGS	TOT CLMS	IND CLMS
11/229,485	09/16/2005	2161	0.00	MSFT124795	6	20	3

CONFIRMATION NO. 8916

38991
CHRISTENSEN, O'CONNOR, JOHNSON, KINDNESS, PLLC
1420 FIFTH AVENUE
SUITE 2800
SEATTLE, WA 98101-2347

FILING RECEIPT

OC000000017165309

DOCKETED

Date Mailed: 10/03/2005

Receipt is acknowledged of this regular Patent Application. It will be considered in its order and you will be notified as to the results of the examination. Be sure to provide the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. If an error is noted on this Filing Receipt, please mail to the Commissioner for Patents P.O. Box 1450 Alexandria Va 22313-1450. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections (if appropriate).

Applicant(s)

Ravisankar V. Pudipeddi, Bellevue, WA;
Vishal V. Ghotge, Seattle, WA;
Sarosh C. Havewala, Redmond, WA;
Ravinder S. Thind, Kirkland, WA;
Mark J. Zbikowski, Woodinville, WA;

Assignment For Published Patent Application

Microsoft Corporation, Redmond, WA

Power of Attorney: None

Domestic Priority data as claimed by applicant

This appln claims benefit of 60/637,407 12/17/2004

Foreign Applications

If Required, Foreign Filing License Granted: 10/03/2005

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is US11/229,485

Projected Publication Date: To Be Determined - pending completion of Missing Parts

RCD

PROCESSED DOCKETING
OCT 6 2005
CHRISTENSEN, O'CONNOR

MAS

8

8

Non-Publication Request: No

Early Publication Request: No

Title

Extensible file system

Preliminary Class

707

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process simplifies the filing of patent applications on the same invention in member countries, but does not result in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4158).

LICENSE FOR FOREIGN FILING UNDER
Title 35, United States Code, Section 184
Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as

9

set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign Assets Control, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

10

10

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING OR 371 (c) DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET NUMBER
11/229,485	09/16/2005	Ravisankar V. Pudipeddi	MSFT124795

CONFIRMATION NO. 8916

38991
CHRISTENSEN, O'CONNOR, JOHNSON, KINDNESS, PLLC
1420 FIFTH AVENUE
SUITE 2800
SEATTLE, WA 98101-2347

FORMALITIES LETTER

Date Mailed: 10/03/2005

NOTICE TO FILE MISSING PARTS OF NONPROVISIONAL APPLICATION

FILED UNDER 37 CFR 1.53(b)

Filing Date Granted

Items Required To Avoid Abandonment:

An application number and filing date have been accorded to this application. The item(s) indicated below, however, are missing. Applicant is given **TWO MONTHS** from the date of this Notice within which to file all required items and pay any fees required below to avoid abandonment. Extensions of time may be obtained by filing a petition accompanied by the extension fee under the provisions of 37 CFR 1.136(a).

- The statutory basic filing fee is missing.
  Applicant must submit $ 300 to complete the basic filing fee for a non-small entity. If appropriate, applicant may make a written assertion of entitlement to small entity status and pay the small entity filing fee (37 CFR 1.27).
- The oath or declaration is missing. *A properly signed oath or declaration in compliance with 37 CFR 1.63, identifying the application by the above Application Number and Filing Date, is required.*
  Note: If a petition under 37 CFR 1.47 is being filed, an oath or declaration in compliance with 37 CFR 1.63 signed by all available joint inventors, or if no inventor is available by a party with sufficient proprietary interest, is required.

The applicant needs to satisfy supplemental fees problems indicated below.

The required item(s) identified below must be timely submitted to avoid abandonment:

- To avoid abandonment, a surcharge (for late submission of filing fee, search fee, examination fee or oath or declaration) as set forth in 37 CFR 1.16(f) of $130 for a non-small entity, must be submitted with the missing items identified in this letter.

SUMMARY OF FEES DUE:

Total additional fee(s) required for this application is $1130 for a Large Entity

- **$300** Statutory basic filing fee.

RCD

11

11

- $130 Surcharge.

- The application search fee has not been paid. Applicant must submit $500 to complete the search fee.
- The application examination fee has not been paid. Applicant must submit $200 to complete the examination fee for a large entity

Replies should be mailed to: Mail Stop Missing Parts
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

A copy of this notice MUST be returned with the reply.

Office of Initial Patent Examination (571) 272-4000, or 1-800-PTO-9199, or 1-800-972-6382
PART 1 - ATTORNEY/APPLICANT COPY

17

EXTENSIBLE FILE SYSTEM

CROSS-REFERENCE TO RELATED APPLICATION

This application claims the benefit of U.S. Provisional Application No. 60/637,407, entitled FILE SYSTEM FORMAT FOR PORTABLE MEDIA, and filed on December 17, 2004. U.S. Provisional Application No. 60/637,407 is incorporated by reference herein.

BACKGROUND

Generally described, there are a number of portable computing devices, such as digital still cameras, digital video cameras, media players, mobile phones, mobile computing devices, personal digital assistants, and the like that maintain data on a storage media, such as a portable storage media. The continued development of more complex portable computing devices and larger storage capacity portable storage media places a greater demand for flexibility on the file system format used on the storage media. Current file system format approaches can become deficient in that they may provide adequate flexibility for increasing storage size capacities and/or storage media applications.

SUMMARY

An extensible file system format for portable storage media is provided. The extensible file system format includes the specification of primary and secondary directory entry types that may be custom defined. The primary and secondary directory entry types can be further classified as critical and benign directory entries.

In accordance with an aspect of the present invention, a computer-readable medium having computer-executable components for storing data is provided. The computer-readable components can include a boot parameters component for specifying boot parameters for a file system. The computer-readable components also include a file

allocation table component for defining a file allocation table associated with the file system. Additionally, the computer-readable components include a primary directory entry component for specifying data in a root directory of the file system. Still further, the computer-readable components include at least one secondary entry component corresponding to the primary directory entry component. The secondary entry component defines defining meta data associated with the primary directory entry component. The primary and secondary directory entry components can be further classified as critical or benign.

In accordance with another aspect of the present invention, a computer-readable medium having computer-executable components for storing data is provided. The computer-readable components include a boot parameters component for specifying boot parameters for a file system. The computer-readable components also include a file allocation table component for defining a file allocation table associated with the file system. Still further, the computer-readable components include a root directory component for specifying data in a root directory of the file system. Additionally, the computer-readable components include at least extensible one meta data component corresponding to the root directory entry component. The meta data component defines meta data associated with the root directory component.

In an illustrative embodiment, a file system will not mount a volume for a critical primary or root directory entry that is not recognized. The file system can ignore benign primary directory entries, critical secondary directory entries and benign secondary directory entries that are not recognized.

This summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This summary is not intended to identify key features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter.

DESCRIPTION OF THE DRAWINGS

The foregoing aspects and many of the attendant advantages of this invention will become more readily appreciated as the same become better understood by reference to the following detailed description, when taken in conjunction with the accompanying drawings, wherein:

FIGURES 1A-1C are block diagrams illustrative of an illustrative environment including a portable computing device and a storage device implementing the extensible file system format in accordance with an aspect of the present invention;

FIGURE 2 is a block diagram illustrative of various volume layout components corresponding to an extensible file system format in accordance with an aspect of the present invention;

FIGURE 3 is a block diagram illustrative of an extensible file system directory structures including primary and secondary directory entry structures in accordance with an aspect of the present invention;

FIGURE 4 is a block diagram illustrative of data components for implementing a boot process block in an extensible file system format in accordance with an aspect of the present invention;

FIGURE 5 is a block diagram illustrative of data components for implementing directory entries in an extensible file system format in accordance with an aspect of the present invention

FIGURE 6 is a block diagram illustrative of data components for implementing a file name and extensions in an extensible file system format in accordance with an aspect of the present invention;

FIGURE 7 is a block diagram illustrative of data components for implementing a volume identifier in an extensible file system format in accordance with an aspect of the present invention;

FIGURE 8 is a block diagram illustrative of data components for implementing an extensible directory entry in an extensible file system format in accordance with an aspect of the present invention;

FIGURE 9 is a block diagram illustrative of data components for implementing an extensible directory entry in an extensible file system format in accordance with an aspect of the present invention;

FIGURE 10 is a block diagram illustrative of data components for implementing an access control list in an extensible file system format in accordance with an aspect of the present invention; and

FIGURE 11 is a flow diagram illustrative of a file name creation routine for an extensible file system format in accordance with an aspect of the present invention.

DETAILED DESCRIPTION

Generally described, the present invention relates to an extensible file system format and various processes associated with the extensible file system format. In an illustrative embodiment, the extensible file system format corresponds to an extensible file system format for portable storage media and various processes associated with the extensible file system format on the portable storage media. Although the present invention will be described with regard to a portable storage media file system format, one skilled in the relevant art will appreciate that the disclosed embodiments are illustrative in nature and should not be construed as limiting. Additionally, one skilled in the relevant art will appreciate that the data structures and data layouts used in the illustrative examples may require additional information related to performance, security, and the like.

FIGURES 1A-1C are block diagrams illustrative of various operating environments 100 for the extensible file system format of the present invention. With reference to FIGURE 1A, in an illustrative embodiment, the extensible file system format is utilized to store data from a computing device, such as a mobile computing device 102, and a storage media, such as a portable storage media 104. In an illustrative embodiment, the mobile computing device 102 can correspond to any one of a variety of computing devices, including but not limited to, portable computing devices, mobile telephones, personal digital assistants, music players, media players. The portable storage media can also include, but is not limited to, hard drives, flash media, micro-drives and other storage media. In an illustrative embodiment, the extensible file system on the portable storage media 104 does not have to include any type of executable or readable software components, such as an operating environment, utilized by the mobile computing device 102. Alternatively, the extensible file system on the portable storage media 104 may include executable or readable software components used by the mobile device 102.

In an illustrative embodiment, the mobile computing device 102 may be in communication with other computing devices for collecting/exchanging data to be stored on the portable storage media 104. With reference to FIGURE 1B, the mobile computing device 102 may be in direct communication with another computing device 106 and storage media 108. In an illustrative embodiment, the direct communication can correspond to various wired and wireless communication methods. In an illustrative

embodiment, the other storage media 108 is not required to be formatted in accordance with the extensible file system format of the present invention. With reference to FIGURE 1C, in a similar manner, the mobile computing device 102 may also be in communication with another computing device 110 and storage media 112, via a network connection. In an illustrative embodiment, the network connection can correspond to local area network (LAN) and wide are network (WAN) connections.

With reference now to FIGURE 2, an illustrative embodiment volume layout 200 for an extensible file system format will be described. The volume layout 200 includes a boot parameters component 202 that include various information related to a description of the file system parameters of the partition. In an illustrative embodiment, the boot parameters component 202 can include code for bootstrapping from a defined partition, fundamental file system parameters for the defined partition, and various error checking information. A data structure for defining at least a portion of the boot parameters will be described below with regard to FIGURE 4.

The volume layout 200 also includes an extensible parameters component, designated as OEM parameters 204, that define various additional data structures used in conjunction with the file system. In an illustrative embodiment, an original equipment manufacture (OEM) may specify various extensible data structures, such as performance parameters for a storage medium, that can be defined at time of manufacture. The volume layout 200 can further include a file allocation table component 206 that defines file and directory allocations. In an illustrative embodiment, each entry in the file allocation table component 206 corresponds to a 32-bit entry that represents an allocated cluster, an unallocated cluster or an unusable cluster. The volume layout 200 can still further include series of file data components 208A-208X that correspond to the data stored according to the file system format. Various data structures for defining a portion of the file data components 208A-208X will be defined with regard to FIGURES 3-10.

Turning now to FIGURE 3, in one aspect, the file data components 208 may include one or more directory entries according to a directory structure 300. In an illustrative embodiment, directory structure 300 is organized into primary directory entries 302 and secondary directory entries 304. Each directory entry in the primary and secondary entries is typed. For example, in an illustrative embodiment, type values for the primary and secondary directory entries can correspond to a range of 1-255. Primary

directory entries 302 correspond to the entries in the root directory of the file system. Secondary directory entries 304 follow a primary directory entry and are associated with the primary directory entry. Secondary directory entries extend the metadata associated with the correlated primary directory entry.

With continued reference to FIGURE 3, in an illustrative embodiment, the primary directory entries 302 can be further classified as critical primary directory entries 306 and benign primary directory entries 308. Critical primary directory entries 306 define potentially different formats for each directory entry. In an illustrative embodiment, an operating environment will not mount a volume corresponding to the extensible file system format with an unknown critical primary directory entry, as will be described below. Examples of known primary directory entries 306 can include allocation bitmaps, up-case tables, volume labels, encryption keys, and normal directory entries. Benign primary directory entries 308 also define potential different formats for each directory entry, but can be ignored by the file system if a particular benign primary directory entry is not understood. Benign primary directory entries 308 can be associated with another cluster chain the volume. Additionally, benign primary directory entries 308 can also be associated a number of secondary directory entries 304.

In a manner similar to primary directory entries 302, secondary directory entries 304 may also be further classified as critical secondary directory entries 310 and benign secondary directory entries 312. As described above, the critical secondary directory entries 310 and benign secondary directory entries 312 are associated with a benign primary directory entry and extend the metadata associated with the primary directory entry. Both the critical secondary directory entries 310 and the benign secondary directory entries 312 can be associated with another cluster chain the volume.

To mount a corresponding to the extensible file system format, the file system implements a mount volume procedure. In an illustrative embodiment, the mount volume procedure attempts to a look at a version number for the volume. If the version number is not understood (e.g., the version number is higher), the volume will not be mounted. During a normal directory enumeration, any critical primary directory entries not known by the file system will prevent the volume from being mounted. Thereafter, various user-initiated processes, such as a file open, will cause the file system to enumerate the secondary directory entries. If the critical secondary directory entries 310 are not known

by a file system, the entire directory entry will be skipped. Additionally, if benign secondary directory entries 312 are not known by the file system, the particular unknown benign secondary directory entry will be ignored.

With reference now to FIGURE 4, a block diagram illustrative of data components 400 for implementing a boot process block in the boot parameters component 202 (FIGURE 2) will be described. The data components 400 include an OEM name component 402 for specifying a name for the file system format of the storage media. The data components 400 also include a data size descriptor component 404 for specifying various characteristics of the data stored in the file system. For example, the data size descriptor component 404 can specify a count of bytes per sector, a number of sectors per allocation unit, a FAT table offset, and a count of sectors for all data structures. The data components include an active FAT flags component 406 for specifying a number of active FATs on the file system. In an illustrative embodiment, a file system may support multiple FATs for utilization with some operating system environments. The data components 400 can further include a volume identification component 408 for identifying a volume serial number and/or version number. Still further, the data components 400 can include a file system type for specifying the file system format for the file system. One skilled in the relevant art will appreciate that the data components 400 can include a number of additional/alternative rows for implementing the above-identified components 402-410 and additional components.

Turning now to FIGURE 5, a block diagram illustrative of data components 500 for implementing directory entries in an extensible file system format will be described. Turning now to FIGURE 6, a block diagram data components 500 for implementing a file name and extensions will be described. The data components 500 include an in use component 502 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 500 further include a type designation component 504 for specifying that the directory entry is associated with a normal directory entry. The data components 500 further include a secondary directory entries component 504 for specifying a number of secondary entries associated with the normal directory entry. The data components 500 also include a file attributes component 508 for specifying various file system attributes for the directory entry. Still further, the data

components 500 include a time component 510 for specifying various time information such as a creation timestamp, modification time stamp and other time information. Additionally, the data components 500 further include a time zone component 512 for specifying a time zone for the last created time stamp. One skilled in the relevant art will appreciate that the data components 500 can include a number of additional/alternative rows for implementing the above-identified components 502-512 and additional components.

Turning now to FIGURE 6, a block diagram data components 600 for implementing a file name and extensions will be described. The data components 600 include an in use component 602 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 600 further include a type designation component 604 for specifying that the directory entry is associated with a file system name. The data components further include a file name length component 606 and a file name has component 608. The utilization of the file name hash component 608 will be described below. The data components 600 also include a file name component 610 for specifying the file name. One skilled in the relevant art will appreciate that the data components 600 can include a number of additional/alternative rows for implementing the above-identified components 602-610 and additional components. Additionally, file name directory entries may be extended by secondary directory entries.

Turning now to FIGURE 7, a block diagram illustrative of data components 700 for implementing a volume identifier in an extensible file system format is provided. The data components 700 include an in use component 702 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 700 further include a type designation component 704 for specifying that the directory entry is associated with a volume identifier. The data components 700 further include a secondary directory entries component 706 for specifying a number of secondary entries associated with the volume identifier. The data components 700 also include a volume identifier 708, such as a global unique identifier. One skilled in the relevant art will appreciate that the data components 700 can include a number of additional/alternative

rows for implementing the above-identified components 702-708 and additional components. Additionally, in an illustrative embodiment, the data components 700 correspond to a benign directory entry that can be ignored by a file system that does not support volume identifiers.

With reference now to FIGURES 8 and 9, in an illustrative embodiment, parties, such as an OEM, may be able to define specific benign primary directory entry types 308 and benign secondary directory entry types 312. As discussed above, in the event the file system would not recognize or understand either the specific benign primary directory entry types 308 or benign secondary directory entry types 312, the file system could ignore the defined directory entry types.

With reference to FIGURE 8, a block diagram illustrative of data components 800 for implementing an extensible benign primary directory entry 308 in an extensible file system format will be described. The data components 800 include an in use component 802 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 800 further include a type designation component 804 for specifying that the directory entry is a benign primary directory entry. The data components 800 further include a secondary directory entries component 806 for specifying a number of secondary entries associated with the volume identifier. The data components 800 also include a volume identifier 808, such as a global unique identifier. The data components 800 can further include additional information 810, such as verification information and a starting cluster. One skilled in the relevant art will appreciate that the data components 800 can include a number of additional/alternative rows for implementing the above-identified components 802-506 and additional components.

With reference to FIGURE 9, a block diagram illustrative of data components 900 for implementing a benign secondary directory entry in an extensible file system format will be described. The data components 900 include an in use component 902 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 900 further include a type designation component 904 for specifying that the directory entry is a benign primary directory entry. The data components 900

further include a secondary directory entries component 906 for specifying a number of secondary entries associated with the volume identifier. The data components 900 also include a volume identifier 908, such as a global unique identifier. The data components 900 can further include additional information 910, such as verification information and a starting cluster. One skilled in the relevant art will appreciate that the data components 900 can include a number of additional/alternative rows for implementing the above-identified components 902-906 and additional components.

In an illustrative embodiment, a benign primary directory entry and/or secondary directory entries may be associated with access control list (ACL) information. FIGURE 10 is a block diagram illustrative of data components 1000 for implementing an access control list in an extensible file system format. The data components 1000 include an in use component 1002 for specifying whether the particular directory entry is in use. In an illustrative embodiment, the high bit of the data components will be set to "1" if the directory entry is in use. The data components 1000 further include a type designation component 1004 for specifying that the directory entry is an ACL directory entry. The data components 1000 further include a number of ACL fields 1006, such as ACL flags, pointers to ACL databases, and the like. One skilled in the relevant art will appreciate that the data components 1000 can include a number of additional/alternative rows for implementing the above-identified components 1002-1006 and additional components.

With reference now to FIGURE 11, a file name creation routine 1100 for an extensible file system format will be described. At block 1102, a file system obtains a request to create a directory entry with a specific file name. In an illustrative embodiment, the specific file name can correspond to a naming convention, such as a digital camera picture naming convention. At block 1104, the file system generates a target name hash. At block 1106, an iterative loop is begun by examining the next directory entry hash value. An illustrative directory entry type for storing directory entry hash values is described above with regard to data components 600 (FIGURE 6).

At decision block 1108, a test is conducted to determine whether the target hash value matches the current directory entry hash value. If they do not match, the routine 1100 returns to block 1106 (until all the directory entries have been examined. If the hash values match at decision block 1108, at block 1110, the file system obtains the full file name for the potentially matching directory entry. An illustrative directory entry

type for storing directory entry full file names is described above with regard to data components 600 (FIGURE 6). At decision block 1112, a test is conducted to determine whether the target file name matches the full file name of the potentially matching directory entry. If so, the routine 1100 terminates by reporting a conflict and the file system will be required to select a new file name. If the full file does not match, the routine 1100 will return to block 1106 to continue checking hash values for all the directory entries in the file system.

In accordance with an aspect of the present invention, various additional functionality may be added through the specification of specific directory types. For example, name streams may be supported by specifying a name stream directory entry. Additionally, on-disk encryption may also be supported through the utilization of specific encryption algorithms and key exchanges. Still further, time zone conversions may be associated with directory entries to automatically convert a current time zone with a time zone with the directory entry was made.

While illustrative embodiments have been illustrated and described, it will be appreciated that various changes can be made therein without departing from the spirit and scope of the invention.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A computer-readable medium having computer-executable components for storing data, the computer-readable components comprising:

a boot parameters component for specifying boot parameters for a file system;

a file allocation table component for defining a file allocation table associated with the file system;

a primary directory entry component for specifying data in a root directory of the file system; and

at least one secondary entry component corresponding to the primary directory entry component and defining meta data associated with the primary directory entry component.

2. The computer-readable components recited in Claim 1, wherein the primary directory entry component is a critical primary directory entry component that must be understood by the file system.

3. The computer-readable components recited in Claim 1, wherein the primary directory entry component is a benign primary directory entry component that can be ignored if unknown to the file system.

4. The computer-readable components recited in Claim 1, wherein the secondary directory entry component is a critical secondary directory entry that will cause the primary directory component and the secondary directory entry component to be ignored if unknown to the file system.

5. The computer-readable components recited in Claim 1, wherein the secondary directory entry component is a benign secondary directory entry that can be ignored if unknown to the file system.

6. The computer-readable components recited in Claim 1 further comprising two secondary entry components corresponding to the primary directory entry component and defining meta data associated with the primary directory entry

7. The computer-readable components recited in Claim 1, wherein the primary directory entry component corresponds to an allocation bitmap defining cluster availability within a storage media.

8. The computer-readable components recited in Claim 1, wherein the primary directory entry component corresponds to a volume identifier.

9. The computer-readable components recited in Claim 1, wherein the primary directory entry component corresponds to a file name identifier.

10. The computer-readable components as recited in Claim 9, wherein the file name identifier includes a full file name and a file name hash.

11. The computer-readable components recited in Claim 1, wherein the at least one secondary directory entry corresponds to an extensible secondary directory entry.

12. The computer-readable components recited in Claim 1 further comprising a manufacturer data component for specifying manufacturer data structures.

13. A computer-readable medium having computer-executable components for storing data, the computer-readable components comprising:

a boot parameters component for specifying boot parameters for a file system;

a file allocation table component for defining a file allocation table associated with the file system;

means for specifying data in a root directory of the file system and meta data associated with data in the root directory.

14. The computer-readable components recited in Claim 13, wherein the means for specifying data in a root directory of the file system and meta data associated with data in the root directory corresponds to a primary directory entry component for specifying data in a root directory of the file system.

15. The computer-readable components recited in Claim 14, wherein the primary directory component can correspond to a critical primary directory entry.

16. The computer-readable components recited in Claim 14, wherein the primary directory component can correspond to a benign primary directory entry.

17. The computer-readable components recited in Claim 13, wherein the means for specifying data in a root directory of the file system and meta data associated with data in the root directory includes at least one secondary entry component corresponding to the data in the root directory and defining meta data associated with the primary directory entry.

18. A computer-readable medium having computer-executable components for storing data, the computer-readable components comprising:

a boot parameters component for specifying boot parameters for a file system;

a file allocation table component for defining a file allocation table associated with the file system;

a root directory component for specifying data in a root directory of the file system; and

at least extensible one meta data component corresponding to the root directory entry component and defining meta data associated with the root directory component.

19. The computer-readable components recited in Claim 18, wherein the root directory component is a critical root directory component that must be understood by the file system.

20. The computer-readable components recited in Claim 18, wherein the root directory entry component is a benign root directory entry component that can be ignored if unknown to the file system.

ABSTRACT

An extensible file system format for portable storage media is provided. The extensible file system format includes the specification of primary and secondary directory entry types that may be custom defined. The primary and secondary directory entry types can be further classified as critical and benign directory entries.

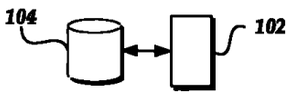


Fig.1A.

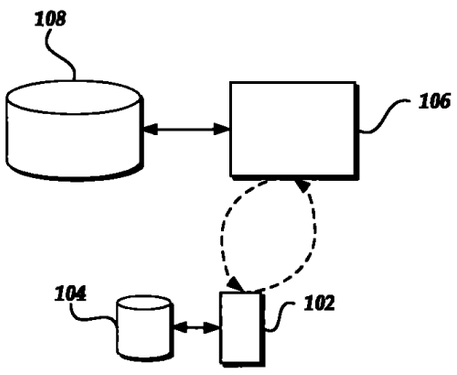


Fig.1B.

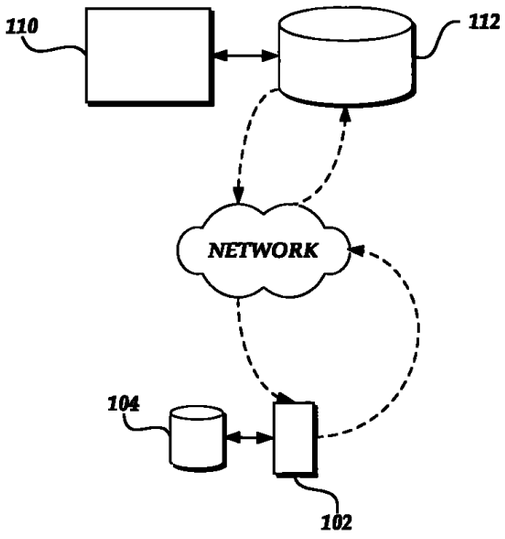


Fig.1C

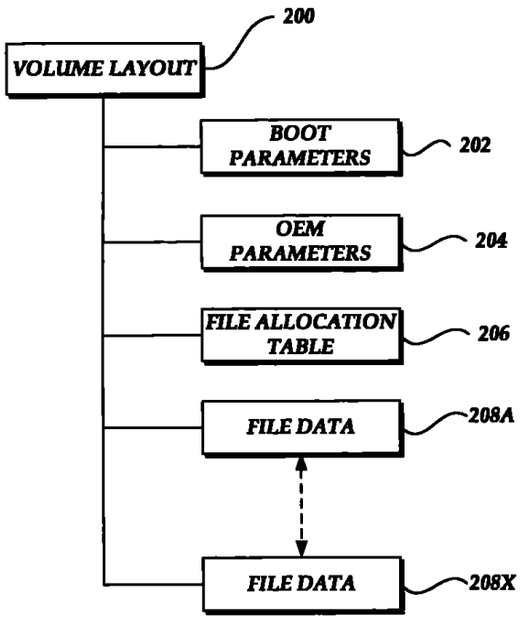

200
VOLUME LAYOUT
BOOT PARAMETERS
202
OEM PARAMETERS
204
FILE ALLOCATION TABLE
206
FILE DATA
208A
FILE DATA
208X

Fig.2.

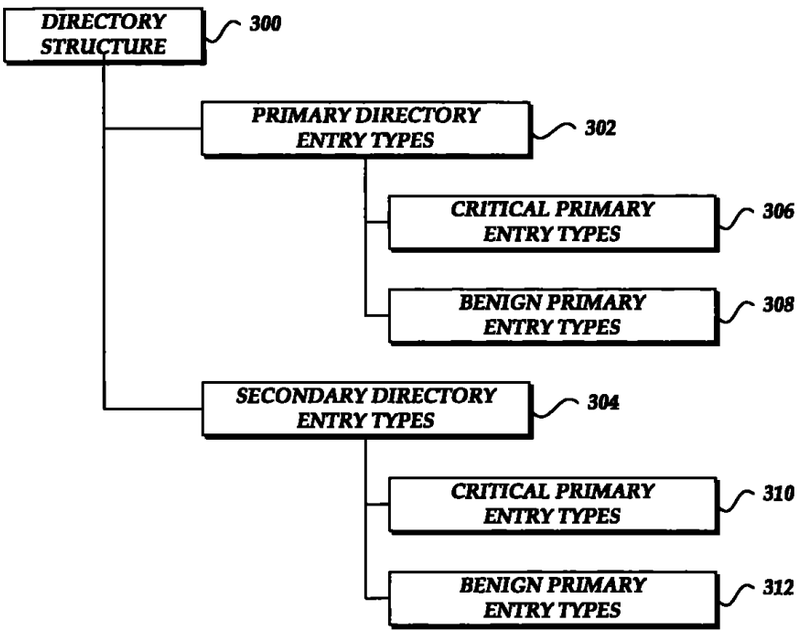

DIRECTORY STRUCTURE
300
PRIMARY DIRECTORY ENTRY TYPES
302
CRITICAL PRIMARY ENTRY TYPES
306
BENIGN PRIMARY ENTRY TYPES
308
SECONDARY DIRECTORY ENTRY TYPES
304
CRITICAL PRIMARY ENTRY TYPES
310
BENIGN PRIMARY ENTRY TYPES
312

Fig.3.

NAME	SIZE	
OEM NAME	3	402
DATA SIZE DESCRIPTORS	X	404
ACTIVE FAT	2	406
VOLUME SERIAL NUMBER	4	408
FILE SYSTEM TYPE	X	410

Fig.4.

NAME	SIZE	
IN USE	11	502
TYPE	17	504
SECONDARY ENTRIES	1	506
ATTRIBUTES	2	508
TIME	X	510
TIME ZONE	1	512

Fig.5.

NAME	SIZE	
IN USE	11	602
TYPE	17	604
CHARACTERS	1	606
NAME HASH	2	608
FILE NAME	28	610

Fig.6.

NAME	SIZE	
IN USE	11	702
TYPE	17	704
SECONDARY ENTRIES	1	706
GUID	16	708

Fig.7.

NAME	SIZE	
IN USE	11	802
TYPE	17	804
SECONDARY ENTRIES	1	806
GUID	16	808
OTHER	X	810

Fig.8.

NAME	SIZE	
IN USE	11	902
TYPE	17	904
GUID	16	906
OTHER	X	910

Fig.9.

NAME	SIZE	
IN USE	11	1002
TYPE	17	1004
ACL INFORMATION	X	1006

Fig.10.

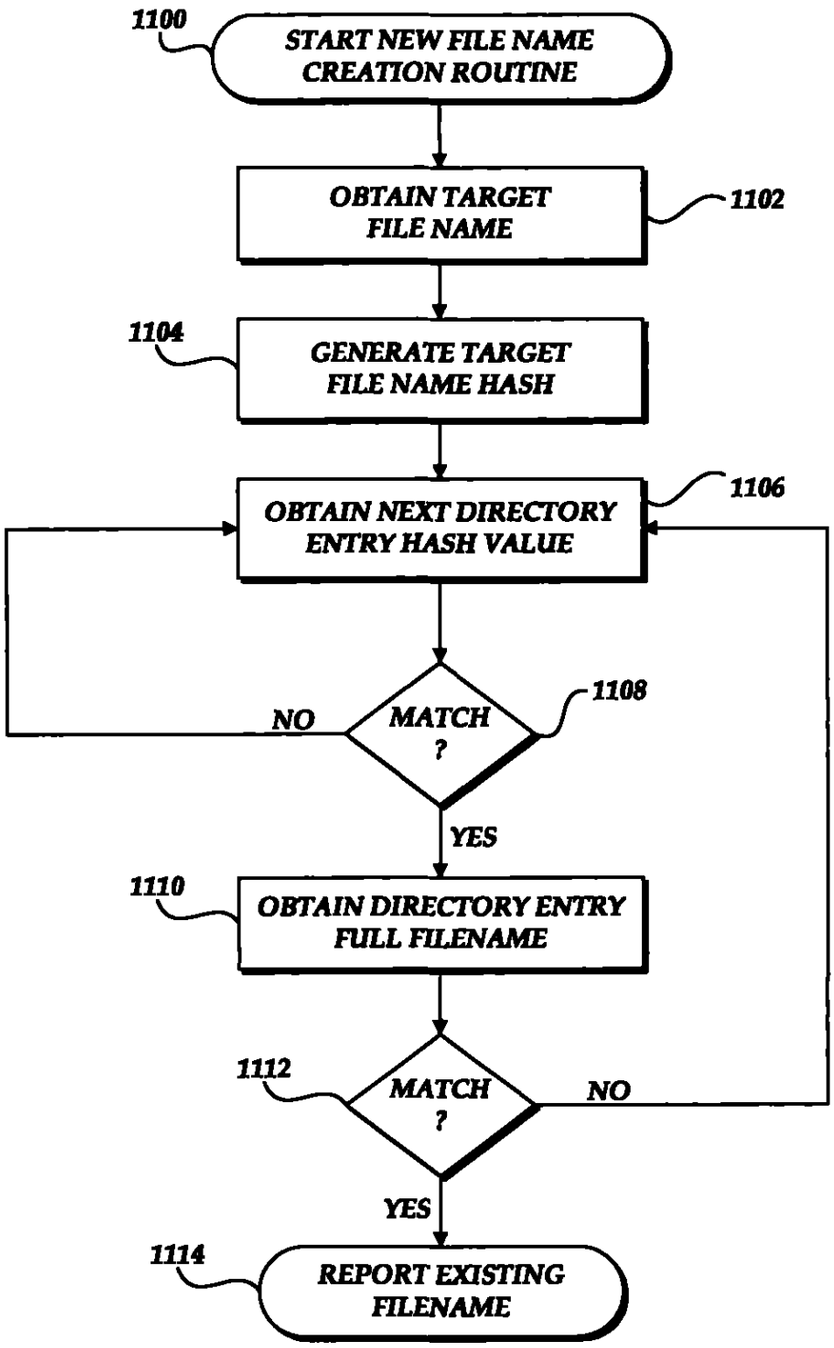

1100
START NEW FILE NAME CREATION ROUTINE
OBTAIN TARGET FILE NAME
1102
1104
GENERATE TARGET FILE NAME HASH
OBTAIN NEXT DIRECTORY ENTRY HASH VALUE
1106
NO
MATCH ?
1108
YES
1110
OBTAIN DIRECTORY ENTRY FULL FILENAME
1112
MATCH ?
NO
YES
1114
REPORT EXISTING FILENAME

Fig.11.

FILE SYSTEM FORMAT FOR PORTABLE STORAGE MEDIA

FIELD OF THE INVENTION

The present invention relates to software and an in particular, to a file system protocol.

BACKGROUND OF THE INVENTION

1. FAT32 is the accepted Filesystem of choice for portable media (such as NAND flash based devices that are commonly used in cameras/cell phones etc.). The capacities of these devices are expected to go up to 32GB+ in the next few years.

2. Allocation speedups are needed – FAT32 has a linear lookup table

3. File size is limited to 4GB in FAT32. Since files such as movies are typically stored on these media, we need to support larger file sizes.

4. Extensibility of FAT32: FAT32 cannot be extended easily now since all the bits in the directory entry are used up. The Long file name format uses a bit mask that precludes any further extension based on bit masks in the directory entries.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

FAT++ LAYOUT

A FAT++ volume has the following layout.

Name	Offset (sector)	Size (sectors)	Description
Boot Sector and BIOS Parameter Block (BPB)	0	1	This sector contains code for boot-strapping from this partition, and fundamental file system parameters describing the type and size of the file system in this partition. The code for this sector is loaded and executed by code in the Master Boot Record. Last 4 bytes should be 0xAA550000.

Name	Offset (sector)	Size (sectors)	Description

Extended Boot Sector 1	1	2	These sectors hold additional code for boot-strapping from the partition, and an FSINFO structure at offset 0x1E4:
			DWORD 0x37373937 -- signature of FSinfo (this differs from FAT32)
			DWORD EraseBlockSize -- Size of erase block for flash devices expressed in number of sectors. For hard disks this should be 1.
			// A total of 10 different OEM specific parameter structures can be supported.
			struct {
			// MS assigned oem types
			DWORD OemParameterType;
			// OEM specific param structure
			UCHAR OemParameters[44]
			} OemParameterArea[10];
			
			OEMs can for instance add performance parameters here for flash devices. These can be added by OEM formatting utility/specific code in OEM implementations of the FAT spec. These will be ignored by Windows implementations.
			UCHAR Reserved[20];
			
			Last 4 bytes should be 0xAA550000 for BOTH sectors.
			Second sector's first 4 bytes should be 0x73739373

Name	Offset (sector)	Size (sectors)	Description
Reserved1	3	3	Zeros
BackupBoot	6	3	Exact copy of sectors 0,1,2.
Reserved2	9	NumReserved-9	Reserved sectors. The number of reserved sectors is defined in the BPB. This area is reserved for future operating system use, and should not be used or modified. NumReserved is *usually* 32 sectors, but it will NOT always be 32.
File Allocation Table (FAT)	NumReserved	varies	A table of 32-bit entries which define file and directory allocations. Each entry either represents an allocated cluster, an unallocated cluster, or an unusable cluster. When defining an allocated cluster, the value for this entry indicates the FAT entry for the next cluster in the file directory, or the end of the allocation chain.
Cluster Heap	...	varies	The rest of the partition is file data, divided into cluster size allocations as defined in the BPB.

FILE SYSTEM REGION NOTES

The following sections provide notes on the regions defined by FAT++.

IMPORTANT: All "reserved" fields below are currently defined to be zero. However, these fields should not be assumed to contain this or any other value. Disk utilities and other programs should not look at nor modify reserved fields. It is the responsibility of the utility to *preserve* whatever value is contained in these fields.

3C

THE BOOT SECTOR AND BPB

Name	Offset (byte)	Size (bytes)	Description
jmpBoot	0	3	jmp instruction to "boot" code
OEMName	3	8	"MSWIN5.0"
bytes per sector	11	2	Count of bytes per sector.
sectors per cluster	13	1	Number of sectors per allocation unit.
reserved sectors	14	2	Number of reserved sectors after the extended boot record plus the size of the boot record and extended boot record. These should be padded as appropriate so that the starting cluster is on the desired boundary – for flash devices, that would be to round it up to an erase block boundary. This enables the data area of the Filesystem which is written to frequently – to start on a properly aligned erase block boundary
Number of FATs	16	1	2
Reserved Entries	17	2	Reserved
Zero	19	2	Should be zero (this is the old bsSectors field).
media descriptor	21	1	OEM defined
Reserved	22	2	Should be zero
sectors per Track	24	2	Sectors per track for interrupt 13h

Name	Offset (byte)	Size (bytes)	Description
count of heads	26	2	Number of heads for interrupt 13h
count of hidden sectors	28	4	Hidden sectors in partition so boot loader can do int 13h.
count of sectors	32	4	Total number of sectors for all data structures and data for this drive.
Big sectors per FAT	36	4	DWORD count of sectors in a single file allocation table (FAT).
Extended Flags	40	2	Bits 0-3 : number of Active FAT. This definitively indicates which FAT is active. 1.) If 0, the primary FAT is used. 2.) If 1, the secondary FAT will be considered the master. This can be used by TFAT++ implementations to indicate the consistent FAT. 3.) If it is 2, then the volume is in a 'dirty' state – i.e. the primary FAT is the right one, but a Filesystem consistency checker has to run to make sure that the FATs/bitmap/directory structure is consistent. Bits 4-15: reserved, should be zero
FS version	42	2	High byte is major revision number. Low byte is minor revision number. This document defines the version to 1:0. **Note: Disk Utilities should respect this field and not operate on volumes with a higher major or minor version number than that for which they were designed.**

Name	Offset (byte)	Size (bytes)	Description
First Cluster of Root	44	4	Cluster number of the first cluster of the root directory. Could be 2, but not necessary. **Note: Disk Utilities that change the location of the root directory should make every effort to place the first cluster of the root directory in the first non-bad cluster on the drive (i.e. in cluster 2, unless it's marked bad).**
FS info sector number	48	2	Sector number of FS info structure. Usually 1. Note that there will be a copy of the FSINFO structure in BackupBoot, but **only the copy pointed to by this field will be kept up to date - i.e. both the primary and backup boot record will point to the same FSINFO sector.**
Backup Boot Record sector number.	50	2	If non zero, indicates the sector number of a copy of the boot record. Usually 6.
Reserved	52	12	Reserved for future expansion.
Drive Number	64	1	Int 13h drive number (e.g. 80h).
Reserved1	65	1	Reserved (used by Windows NT).
Boot signature	66	1	Extended boot signature (29h).
Volume ID	67	4	Volume serial number
Volume label	71	11	Volume label

Name	Offset (byte)	Size (bytes)	Description
File System type	82	8	"FAT++"

FAT++ will strictly enforce the required FS version number, file system type and partition type to be identical to the description above. Any non-conformity will result in the volume being not-recognized – or worse recognized as FAT32 potentially.

FILE ALLOCATION TABLE (FAT)

The File Allocation Table is a packed list of 32-bit entries which have a one-for-one mapping with the data clusters.

Each entry is defined as :

Bit Offset	Size (bits)	Name	Description
0	28	cluster value	0x00000000 unallocated cluster 0x00000001 invalid value 0x00000002 - *0x0FFFFFF7* allocated cluster in chain 0x0FFFFFF7 bad cluster mark *(but only if not part of a cluster chain or not a legal cluster number. NOTE: this means this can be a valid cluster number in a cluster chain!)* 0x0FFFFFF8 - 0x0FFFFFFF end of File
29	4	Reserved	Reserved for future use.

There is no change in the FAT structure from FAT32.

ALLOCATION BITMAP

In addition to the FAT, the format also allows for a bitmap of available/used cluster ranges to be stored on the disk. Each bit in the bitmap indicates if any cluster in the corresponding cluster range is free or all are used. 1 indicates all the clusters are used, and 0 indicates at least one cluster is fri. The bitmap will have as many bits as there are *cluster*

ranges on the volume. A cluster range is a region of contiguous clusters, and is defined in terms of numbers of clusters. Cluster ranges can only be multiples of clusters in powers of 2 – i.e. it could be 1, 2, 4, 8, 16 etc. It is expressed in the entry as a logarithm to the base 2 of the number of clusters that comprise a cluster range. The size of the cluster range is fixed at format time for the entire volume. It cannot be changed until the volume is re-formatted.

For instance, for a 1 GB volume, with a cluster size of 16K, and size of cluster range = 1, the number of cluster ranges will be 64K. So the size of the bitmap is 64K/8 = 8 Kilobytes.

If the size of the cluster range = 2, the number of cluster ranges will be 32k.

The bitmap exists only as a root directory entry of Type 0x1 (refer to next section for a description of the type field in a directory entry). The format of the bitmap root directory entry is:

Name	Offset (byte)	Size (bytes)	Description
Type	0	1	Value is 1
Size of cluster range	1	1	Bitmap granularity. This is expressed as logarithm of number of clusters. The default is 0 – indicating a cluster range size of 2^0, i.e. 1 cluster in which case there is 1 bit per cluster in the bitmap. If the value is 1 for instance, there is 1 bit per 2 contiguous clusters.
Reserved	2	20	Reserved: should be zero
File size	22	6	Size of bit map: expressed in little endian manner.
Starting Cluster	28	4	Starting cluster for the bitmap: expressed in little endian manner

The FAT will contain cluster chains that link the allocations for the clusters that contain the bitmap. However we do recommend that the bitmap be allocated contiguously when the Filesystem is formatted. The bitmap will contain 1's for the clusters that are assigned to the bitmap itself.

DIRECTORY STRUCTURE

The directory structure for FAT++ has been changed as below. It is incompatible with FAT32 – however directory entries are still 32-bytes each in size. Short file names have been eliminated (but it is easy to add them back as we'll explain below if necessary). Primarily directory entries are *typed*. There are 2 categories of types:

1. **Primary types.** These are directory entries which have the type values 0-127. Primary type 0 refers to a normal, user visible file/directory entry. These will be exposed to the user directly. All other primary types (i.e. types 1 thru 127) – can only exist as entries in the root directory in FAT++. Each primary type entry defines a potentially different format for the directory entry. **FAT++ implementations should not assume the format for any primary type they do not understand.** Bytes 22-27 of a primary type entry however will always indicate the size the entry describes and bytes 28-32 of the primary type entry will always point to the starting cluster of the chain of clusters this entry describes. If the size is 0, the starting cluster should be 0 as well.

Secondary types: one or more directory entries may follow a primary type directory entry and are associated with the primary type directory entry. The secondary types extend the metadata contained in the primary directory entry. They should follow the primary type entry. Each secondary type has a different format for the entry. **FAT++ implementations should not assume the format for any secondary type they do not understand.**

Normal directory entry

Name	Offset (byte)	Size (bytes)	Description
Type	0	1	Type of the directory entry. This format describes type 0 entry. 0-127 - are primary types. 0 - is a normal directory entry, that will show up enumerated 1 - indicates this is a bitmap entry and is described above 2 - this is reserved for use as a password encrypted encryption master key entry for the volume The rest are currently reserved. 128-255 are secondary types. These extend the current directory entry. Every primary type directory entry may be followed by zero or more secondary type directory entries. Currently we define the following secondary type: 129 is a file name entry. This will always follow a Type 0 directory entry. There may be more than 1 such entry if the file name does not fit in one. Directory enumeration will enumerate type 0 entries, retrieving file names from the secondary type entries of type 129. When a primary type entry is encountered again, it will be the start of a new directory entry. The rest are currently reserved.

Name	Offset (byte)	Size (bytes)	Description
Number of secondary entries that follow	1	1	This indicates the number of extended directory entries (i.e. of type 128-255) that follow this entry. This **should** match the number of extended directory entries that follow this entry. If they don't, corruption of the FS will be flagged by Windows and a chkdsk will be initiated to fix the problem (which may require deleting the directory entry completely).
Checksum	2	2	2-byte checksum for this directory entry
Attributes	4	2	File attributes: ATTR_READ_ONLY 0x0001 ATTR_HIDDEN 0x0002 ATTR_SYSTEM 0x0004 ATTR_VOLUME_ID 0x0008 ATTR_DIRECTORY 0x0010 ATTR_ARCHIVE 0x0020 The upper two bits of the low attribute byte are reserved. The low attribute byte is the at offset 4 the high one is at offset 5. The high attribute byte represent extended attributes for FAT+ that are not yet defined We also propose support for ATTR_ENCRYPTED to files/directories. Encryption will be discussed below in a separate section.
Create time	6	8	Create timestamp expressed in milliseconds after January 1, 1970 00:00:00 GMT. The time stamp is expressed in a little endian

Name	Offset (byte)	Size (bytes)	Description
Last modified time	14	8	Last modified timestamp expressed in milliseconds after January 1, 1970 00:00:00 GMT
File size	22	6	Size of file (48-bit size). This allows for a maximum size of 256 terabytes for a file, that is bigger than the volume sizes that are supported for FAT+. The 6 bytes are in little endian order.
Starting cluster	28	4	Starting cluster number of this file This is expressed in the little endian manner: i.e. LSB is at offset 12 and MSB is at offset 15.

FILENAME DIRECTORY ENTRY

Following each directory entry of type 0, *one* or more type 128 directory entries should follow immediately. Note that this is different from FAT32 where LFN entries precede the short file name entry.. These represent the file name for the entry. Names are restricted to 255 characters – like FAT32 – not including a trailing NULL. Characters are 16-bit UNICODE.

Name	Offset (byte)	Size (bytes)	Description
Type	0	1	128
Reserved	1	1	Reserved : should be zero.
Checksum	2	2	2-byte checksum for this entry
File-name	4	28	UNICODE file name characters. This allows for 14 UNICODE characters.

FAT++ will no longer support the '.' and '..' entries in a directory. The Filesystem is expected to materialize these as needed in usage. This simplifies directory rename logic.

Note also that short file names have been removed. The file has only one name.

Directory entries may be extended in future via types 129-255: however these should again be collocated with the primary type directory entry. For instance named streams/extended attributes for files can be implemented by extended entries if necessary.

IMPLEMENTATION REQUIREMENTS

DEALING WITH UNKNOWN TYPES

All FAT++ implementations should deal with the described directory type entries in this document. However it is possible that in future we may add more primary as well as extended types (i.e. both in the 0-127 range as well as in the 128-255 range). Here are the requirements for down-level FAT++ implementations to deal with the extensions gracefully:

1.) Any unknown primary type directory entry should not be visible to users. Secondly these entries should not be touched, nor the clusters that they point to ever garbage collected. Bytes 22-27 describe the size of the region the entry describes, and bytes 28-32 describe the starting cluster number (if any). Unknown primary type entries should not be deleted or updated in any manner by implementations.

2.) Any unknown secondary type directory entry should be simply ignored by the implementation and skipped over. When a file is deleted however – *all* secondary type entries should be deleted as well.

FILESYSTEM RECOGNITION

FAT++ implementations will strictly enforce the requirements as opposed to FAT32 which had a lot of variants. The only permissible extensions are via new primary/secondary types and these should always be reviewed. This allows all implementations to work well with each other.

ENCRYPTION SUPPORT

FAT++ would support encryption in the on-disk format. However it is not required by all file system implementations to add encryption support. Encryption on portable media that can move between Windows & devices, for instance, will have to support a symmetric

key produced via AES, using a user password salt that can be stored on the media. We propose the following implementation for encryption support:

1. User's password is a secret, that will be stored encrypted on the volume. This is encrypted using the symmetric master key.
2. The symmetric master key will be used to encrypt all files/data. Note this is not a per user-key, but a global key for that volume. This is encrypted using the user password.

Mutual encryption will ensure that if the media is lost, the key is safe, and so the data cannot be decrypted. Now consider the scenario:

1. User turns encryption on for the CF on his device. The device uses the device's unlock password to generate the symmetric key. If the device doesn't support unlock PINs, it can prompt the user for a media-specific password. Now the symmetric key & password are mutually encrypted and stored on the media in a Type 2 file.
2. User plugs the CF card into a Windows machine.
3. Filesystem recognizes the volume/directory/file (as the case maybe) is encrypted, and prompts the user for a password (this can be indirect-ed via Shell, which prompts the user and supplies the password to the file system).
4. Windows decrypts the symmetric key on the volume using the user supplied pin/password.
5. Windows encrypts the clear text password in memory and compares with the encrypted password stored on the CF card.
6. If not identical, the user gets ERROR_ACCESS_DENIED.
7. Otherwise the in-memory only key is now used to decrypt/encrypt the contents of the volume/file.

   Now the user plugs back the CF card into the device.
8. The device will compare the encrypted form of the in-memory copy of the device's user password – when the device was unlocked – with the on-disk password. If they are identical, it does not have to re-prompt the user for the password. If not, it prompts the user for a different password to be used for that volume.
9. Device decrypts the master key using the in-memory clear-text password.
10. The in-memory only key now is used to decrypt/encrypt contents.

57

Note that we have not yet proposed any actual encryption algorithms/ key sizes. However we'll iron them out as part of the Type 2 dir entry description which will outline the key & password sizes, and the on disk structure lay out, as well as potentially a version & description identifier for the mode of the encryption that is used.

An alternative to storing the encrypted password on the disk is to use a unique predetermined value (i.e. a cookie) or a one-way hash of the password (for better security) that is encrypted with the symmetric key and stored on the disk. Windows would now decrypt the key using the user password, and once it obtains the key, it uses the key to do a one-way hash of the clear text user password in memory. If it matches with the on-disk value, the key can be used now to decrypt/encrypt contents. Otherwise Windows returns STATUS_ACCESS_DENIED.

Metadata (i.e. the FAT, bitmap, boot sector and the directory entries) should not be encrypted for FAT++ as the media will not be recognized. Only the contents should be encrypted on a per volume or a per-file basis using the master symmetric key.

TFAT++ SUPPORT

WinCE implements a version of FAT called TFAT, that ensures that the FAT is not corrupted. The NumberOfFats field is used to achieve the atomicity in updating FAT. Essentially the NumberOfFats field is flipped to 0 after updating the secondary FAT, and flipped back to 2 after updating the primary FAT. So if there's a crash, and NumberOfFats is 2, the primary FAT is used, and if it is 0, the secondary FAT is copied to the primary FAT as part of recovery, before flipping it back to 2.

Support for TFAT++ in Windows would not use the NumberOfFats field. It will use the Active FAT bit in the boot sector. Refer to the Extended Flags field in the boot sector above for the description.

Deeper support would include using the same algorithm for updating the FAT as TFAT. Note that on Windows, we write through all data, so in essence this will protect us only when someone attempts to hot unplug a piece of media when there are open handles. This is an interesting scenario – the Windows support will ensure that WinCE /devices can continue to implement TFAT++ for the devices that it makes sense for (cell phones for instance), and we support recovery in Windows.

NAMED STREAMS

Named streams can be easily supported in FAT+ via a new type code. Type code 129 will be reserved for this purpose. Named streams would be useful in the sense that the metadata that is usually stored by cameras for instance in separate files, thumbnails stored by Explorer, can be collocated with the same file, and copied from Windows NTFS based stores and the devices without loss of fidelity.

ACCESS CONTROL

Access to files can be restricted by using Access control lists (ACLs). FAT++ allows files to have access control lists via a new secondary type. Type code 130 will be reserved for ACLs for a file. In addition, a primary type code – 3, will be reserved to serve as a central ACL store.

While illustrative embodiments of the invention has been illustrated and described, it will be appreciated that various changes can be made therein without departing from the spirit and scope of the invention.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. All embodiments described and disclosed above.

MS Docket No. 312054.02
(OC Docket No. MSFT124795)

PATENT ASSIGNMENT

WE, Ravisankar V. Pudipeddi, Vishal V Ghotge, and Mark J. Zbikowski ("ASSIGNORS 1"), have invented subject matter ("INVENTION 1") disclosed and/or claimed in a patent application entitled "FILE SYSTEM FORMAT FOR PORTABLE STORAGE MEDIA" (APPLICATION 1"), which was filed on December 17, 2004, and was given U.S. Provisional Application No. 60/637,407, and Ravisankar V. Pudipeddi, Vishal V. Ghotge, Sarosh C. Havewala, Ravinder S. Thind, and Mark J. Zbikowski ("ASSIGNORS 2"), have invented subject matter ("INVENTION 2") disclosed and/or claimed in a patent application entitled "EXTENSIBLE FILE SYSTEM" ("APPLICATION 2"), which was filed on September 16, 2005 and was given U.S. Application No. 11/229,485;

Microsoft Corporation, a Washington corporation, on behalf of itself and its successors and assigns ("ASSIGNEE"), is entitled to, and is desirous of acquiring, the entire and exclusive rights, title, and interest in INVENTION 1 and INVENTION 2 and APPLICATION 1 and APPLICATION 2 (and all other applications and patents derived therefrom, such as continuing applications, in and for the United States, its territories, and all foreign countries ("APPLICATION DERIVATIVES"));

For good and valuable consideration, the receipt of which is hereby acknowledged by ASSIGNORS 1 and ASSIGNORS 2, ASSIGNORS 1 and ASSIGNORS 2 hereby sell, assign, and transfer to the ASSIGNEE the entire and exclusive rights, title, and interest in INVENTION 1 and INVENTION 2 and APPLICATION 1 and APPLICATION 2 (and APPLICATION DERIVATIVES);

ASSIGNORS 1 and ASSIGNORS 2 agree to execute all instruments and documents required for the making and prosecution of APPLICATION 1 and APPLICATION 2 (and APPLICATION DERIVATIVES), for litigation regarding letters patent derived therefrom, and for the purpose of protecting and perfecting title to APPLICATION 1 and APPLICATION 2 (and APPLICATION DERIVATIVES).

10/24/05	PVRavisankar
Date	Ravisankar V. Pudipeddi
10/24/05	
Date	Vishal V Ghotge
10/24/05	
Date	Sarosh C. Havewala
10/24/2005	
Date	Ravinder S. Thind
10/24/2005	Mark J. Zbikowski
Date	Mark J. Zbikowski

MAU:md

51

UNITED STATES PATENT AND TRADEMARK OFFICE

UNDER SECRETARY OF COMMERCE FOR INTELLECTUAL PROPERTY AND
DIRECTOR OF THE UNITED STATES PATENT AND TRADEMARK OFFICE

OCTOBER 24, 2005

PTAS *500056849A*

MAURICIO A. URIBE, ESQ.
1420 FIFTH AVENUE, SUITE 2800
SEATTLE, WA 98101

UNITED STATES PATENT AND TRADEMARK OFFICE
NOTICE OF RECORDATION OF ASSIGNMENT DOCUMENT

THE ENCLOSED DOCUMENT HAS BEEN RECORDED BY THE ASSIGNMENT DIVISION OF THE U.S. PATENT AND TRADEMARK OFFICE. A COMPLETE MICROFILM COPY IS AVAILABLE AT THE ASSIGNMENT SEARCH ROOM ON THE REEL AND FRAME NUMBER REFERENCED BELOW.

PLEASE REVIEW ALL INFORMATION CONTAINED ON THIS NOTICE. THE INFORMATION CONTAINED ON THIS RECORDATION NOTICE REFLECTS THE DATA PRESENT IN THE PATENT AND TRADEMARK ASSIGNMENT SYSTEM. IF YOU SHOULD FIND ANY ERRORS OR HAVE QUESTIONS CONCERNING THIS NOTICE, YOU MAY CONTACT THE EMPLOYEE WHOSE NAME APPEARS ON THIS NOTICE AT 571-272-3350. PLEASE SEND REQUEST FOR CORRECTION TO: U.S. PATENT AND TRADEMARK OFFICE, MAIL STOP: ASSIGNMENT SERVICES DIVISION, P.O. BOX 1450, ALEXANDRIA, VA 22313.

RECORDATION DATE: 10/24/2005 REEL/FRAME: 016677/0359
NUMBER OF PAGES: 3

BRIEF: ASSIGNMENT OF ASSIGNOR'S INTEREST (SEE DOCUMENT FOR DETAILS).
DOCKET NUMBER: MSFT-1-24795

ASSIGNOR:
PUDIPEDDI, RAVISANKAR V. DOC DATE: 10/24/2005

ASSIGNOR:
GHOTGE, VISHAL V. DOC DATE: 10/24/2005

ASSIGNOR:
HAVEWALA, SAROSH C. DOC DATE: 10/24/2005

ASSIGNOR:
THIND, RAVINDER S. DOC DATE: 10/24/2005

ASSIGNOR:
ZBIKOWSKI, MARK J. DOC DATE: 10/24/2005

52

016677/0359 PAGE 2

ASSIGNEE:
MICROSOFT CORPORATION
ONE MICROSOFT WAY
REDMOND, WASHINGTON 98052

SERIAL NUMBER: 11229485 FILING DATE: 09/16/2005
PATENT NUMBER: ISSUE DATE:
TITLE: EXTENSIBLE FILE SYSTEM

KIMBERLY WHITE, EXAMINER
ASSIGNMENT DIVISION
OFFICE OF PUBLIC RECORDS

53

PATENT ASSIGNMENT

Electronic Version v1.1
Stylesheet Version v1.1

10/24/2005
500056849

SUBMISSION TYPE:	NEW ASSIGNMENT
NATURE OF CONVEYANCE:	ASSIGNMENT

CONVEYING PARTY DATA

Name	Execution Date
Ravisankar V. Pudipeddi	10/24/2005
Vishal V. Ghotge	10/24/2005
Sarosh C. Havewala	10/24/2005
Ravinder S. Thind	10/24/2005
Mark J. Zbikowski	10/24/2005

RECEIVING PARTY DATA

Name:	Microsoft Corporation
Street Address:	One Microsoft Way
City:	Redmond
State/Country:	WASHINGTON
Postal Code:	98052

PROPERTY NUMBERS Total: 1

Property Type	Number
Application Number:	11229485

CH $40.00 11229485

CORRESPONDENCE DATA

Fax Number: (206)224-0779
Correspondence will be sent via US Mail when the fax attempt is unsuccessful.
Phone: 206.695.1653
Email: efiling@oojk.com
Correspondent Name: Mauricio A. Uribe, Esq.
Address Line 1: 1420 Fifth Avenue, Suite 2800
Address Line 4: Seattle, WASHINGTON 98101

ATTORNEY DOCKET NUMBER:	MSFT-1-24795
NAME OF SUBMITTER:	Mauricio A. Uribe, Esq.

84

54

Total Attachments: 1
source=24795 Assignment#page1.tif

55

OFICINA DE PATENTES Y MARCAS REGISTRADAS DE E.U.

DEPARTMENTO DE COMERCIO DE EU
Oficina De Patentes Y Marcas Registradas de E.U.
Dirección: COMISIONADO PARA PATENTES
P.O. Box 1450
Alexandria, Virginia 21313-1450

Solicitud No	Fecha(c) de presentación o 371	UNIDAD ART	TASA DE ARCHIVADO	ATTY DOCKET NO.	DIBUJOS	TOT CLMS	IND CLMS
11/229.485	09/16/2005	2161	0.00	MSFT124795	6	20	3

38991
CHRISTENSEN, O'CONNOR, JOHNSON, K1NDNESS, PLLC
1420 FIFTH AVENUE
SUITE 2800
SEATTLE, WA 98101-2347

CONFIRMACIÓN NO. 8916
RECIBO DE ENTREGA
OCCO0000017165309

Fecha de envio: 10/03/2005

Se acusa recibo de esta Solicitud de Patente corriente. Se le considerará en su orden y se le notificará sobre los resultados del examen. Asegúrase de suministrar el número de solicitud de E.U., fecha de entrega, nombre del solicitante, y título del invento cuando consulte acerca de esta solicitud. Las tasas que se transmiten con cheque o giro estarán sujetas a recolección. Por favor verifique que los datos presentados en este recibo estén correctos. Si encuentra un error en este recibo de entrega, por favor escriba Al Comisionado para Patentes P.O. Box Alexandria Va 22313-1450.. Por favor suministre una copia de este Recibo de Entrega con los cambios que desea. Si usted recibe una "Notificación de Partes Faltantes al Archivo" para esta solicitud, por favor envíe las correcciones a este recibo de entrega con su respuesta a la notificación. Cuando el USPTO procesa la respuesta a la notificación, la uspto generara otro recibo de entrega incorporando las correcciones solicitadas (si es apropiado).

Solicitante(s)
Ravisankar V. Pudipeddi, Bellevue, WA
Vishal V. Ghotge, Seattle, WA
Sarosh C. Havewala, Redmont, WA
Ravinder S. Thind, Kirkland, WA
Mark J. Zbikowski, Woodinville, WA

Asignación para la Solicitud de Patente Publicada
Microsoft Corporation, Redmond, WA;

Poder del Abogado: Ninguno

Datos de Prioridad Domésticos como los Reivindica el Solicitante
Esta solicitud reclama los beneficios de 60/637.407 12/17/2004
Solicitudes Extranjeras:

Si se requiere, Licencia de Entrega Extranjera Concedida: 10/03/2005
El código del país y el numero de su solicitud de prioridad que se debe usar para entrega extranjera bajo la convención de Paris, es US11/229,485

Fecha proyectada de publicación del: Para determinarse- en espera de documentos faltantes

Solicitud de no-publicación: No

Solicitud de publicación temprana: No
Titulo: Sistema de Archivos Extensible

Clase Preliminar
707

PROTEGIENDO SU INVENTO FUERA DE LOS ESTADOS UNIDOS

Puesto que los derechos otorgados por la patente de E.U. se extiende únicamente a través de los territorios de los Estados Unidos y no tienen efecto en países extranjeros, un inventor que desee proteger la patente en otro país debe solicitar la patente en el país específico o en las oficinas regionales. El solicitante puede considerar llenar una Solicitud Internacional bajo el Tratado de Cooperación de Patentes (PCT). Una solicitud internacional (PCT) generalmente tiene el mismo efecto que una solicitud de patente nacional en cada uno de los países miembros del PCT. El proceso PCT simplifica la solicitud de patentes sobre el mínimo invento en los países que son miembros, pero no resulta en el otorgamiento de una "patente internacional" y no elimina la necesidad de llenar documentos adicionales y tasas en países donde se desee la protección de la patente.

Cada país tiene sus propias leyes de patente, y una persona que desee una patente en un país particular debe hacer la solicitud de patente en ese país de acuerdo a sus leyes particulares. Puesto que las leyes de muchos países difieren en varios respectos de las leyes de patente en los Estados Unidos, se le advierte a los solicitantes buscar una guía en el país extranjero específico para asegurarse que los derechos de patente no se pierdan en forma prematura.

También se le aconseja a los solicitantes en caso de inventos hechos dentro de los Estados Unidos, que el Director del USPTO debe emitir una licencia antes de que los solicitantes puedan aplicar para una patente en un país extranjero. La entrega de una solicitud de patente de E.U. sirve como petición para la solicitud de una licencia extranjera. El recibo de petición de la solicitud contiene información adicional en cuanto al status de las licencia para presentación en el extranjero.

Los solicitantes que desee consultar el manual USPTO "Información general concerniente a las patentes" (específicamente la sección llamada "tratados y patentes extranjeras") para información adicional sobre el tiempo y las fechas límites para entregar La guía se consigue contactando al USPTO Contact Center at 800-786-9199, o se puede ver en la página web de USPTO en http://www.uspto.gov/web/offices/pac/doc/general/index.html.

Para información sobre la prevención de robo de la propiedad intelectual (patentes, marcas registradas y derechos de autor), usted puede consultar la página web del gobierno de los EU en, http://www.stopfakes.gov. parte de la iniciativa del Departamento de Comercio, esta página web incluye kits de herramientas de auto ayuda se le da una vía en cómo proteger la propiedad intelectual en países específicos tales como China, Corea y México. para preguntas concernientes a la ejecución el solicitante puede llamar a la línea directa del Gobierno de los EU al 1-866-999-HALT (1-866-999-4158).

LICENCIA PARA ENTREGA EXTRANJERA BAJO

Titulo 35, Código de los Estados Unidos, Sección 184
Titulo 37, Código de Reglamento Federal, 5.11 & 5.15

CONCEDIDA

Al solicitante se le ha concedido la licencia bajo 35 U.S.C. 184, si la frase "SI SE REQUIERE, LA LICENCIA DE ENTREGA EXTRANJERA SE CONCEDE" seguida de una fecha que aparece en este formato. Estas licencias se conceden para todas las solicitudes donde las condiciones que se requieren para una licencia

57

se han cumplido, sin importar si la licencia se requiere o no tal como se expone en 37 CFR 5.15. El alcance y las limitaciones de estas licencias se exponen en 37 CFR 5.15(a) al menos de que una licencia anterior se haya remitido bajo 37 CFR 5.15(b). La licencia está sujeta a revocación con una notificación por escrito. La fecha que se indica en la fecha efectiva de la licencia, al menos de que una licencia anterior de alcance similar se haya concedido bajo 37 CFR 5.13 o 5.14.

El licenciado debe guardar esta licencia y la puede utilizar en cualquier momento en la fecha que se hace efectiva o luego al menos de que se revoque. Esta licencia se transfiere automáticamente a cualquier aplicación relacionada que se haya entregado bajo 37 CFR 1.53(d). Esta licencia no es retroactiva.

La concesión de una licencia no aminora la responsabilidad del licenciado para la seguridad del tema a tratar como lo impone cualquier contrato con el Gobierno o las prescripciones de leyes existentes relacionadas con el espionaje y la seguridad nacional o la exportación de datos técnicos. Los licenciados se deben informar sobre el reglamento vigente especialmente respecto a ciertos países, otras agencias, y particularmente la Oficina de Defensa de Controles de Comercio, departamento de estado (con respecto a Armas, Municiones e Implementos de Guerra (22 cfr 121-128)); la Oficina de Administración de Exportaciones, Departamento de Comercio (15 cfr 370.10(J)); la Oficina de Control de Activos, Departamento del Tesoro (31 CFR partes 500+) y el Departamento de Energía.

NO CONCEDIDA

Por el momento no se ha concedido la licencia bajo 35 U.S.C. 184, si la frase "si se requiere, la licencia de entrega extranjera se concede" no aparece en este formato. El solicitante puede todavía hacer una petición para una licencia bajo 37 CFR 5.12 si desea una licencia antes del vencimiento de seis meses a partir de de la fecha entrega de los solicitud. Si han pasado seis meses desde la fecha de entrega de esta solicitud y la licencia no no ha recibido una indicación de una orden de secreto bajo 35 U.S.C. 181, el licenciado puede entregar la solicitud extranjera de acuerdo a 37 CFR 5.

OFICINA DE PATENTES Y MARCAS REGISTRADAS DE E.U.

DEPARTMENTO DE COMERCIO DE EU
Oficina De Patentes Y Marcas Registradas de E.U.
Dirección: COMISIONADO PARA PATENTES
P.O. Box 1450
Alexandria, Virginia 21313-1450

SOLICITUD No.	FECHA DE ENTREGA O 371	PRIMER SOLICITANTE	NO. ROTULO DEL ABOGADO
11/229.485	09/16/2005	Ravisankar V.Pudippedi	MSFT124785

CONFIRMACIÓN NO. 8916

38991
CHRISTENSEN, O'CONNOR, JOHNSON, K1NDNESS, PLLC
1420 FIFTH AVENUE
SUITE 2800
SEATTLE, WA 98101-2347

Fecha de entrega 10/03/2005

NOTIFICACION DE PARTES FALTANTES A LA SOLICITUD NO PROVISIONAL
ENTREGADO BAJO 37 CFR 1.53(b)
Fecha de entrega otorgada

Items requeridos para evitar el abandono:

En un número de solicitud y fecha de entrega se le ha otorgado a esta solicitud. Los items indicados más abajo, sin embargo, hacen falta. Al solicitante se le dan dos meses a partir de la fecha de esta Notificación dentro de los cuales debe entregar todos los items requeridos y pagar las tasas requeridas más abajo para evitar el abandono. La extensión de tiempo se puede obtener solicitando una petición acompañada del pago de la extensión bajo la provisión 37CFR 1.136 (a).

- La tasa básica de entrega establecida por la ley hace falta.
  El solicitante debe enviar$300 para completar la tasa básica para entidades no pequeñas. Si es apropiado, el solicitante puede enviar una solicitud escrita que lo acredite con el status de pequeña entidad y pagar la tasa correspondiente a pequeña entidad (37CFR 1. 27).
- La declaración juramentada hace falta. Una declaración juramentada y firmada en forma apropiada de acuerdo con 37CFR 1. 63, identificando la solicitud con el Número de Solicitud y Fecha de Entrega se requiere.
  Nota: si la solicitud bajo 37CFR 1. 47 se entrega, una declaración juramentada de acuerdo con 37CFR 1. 63 firmada por todos los inventores juntos, o si no hay un inventor disponible por una parte con suficiente interes en la propiedad, se requiere.

El solicitante necesita llenar a los requisitos adicionales de tasas indicados más abajo.

Los items identificados más abajo se deben entregar a tiempo para evitar el abandono.

Para evitar el abandono, al cargo adicional (para entregas tardías una tasa de entrega, una tasa de búsqueda, una tasa de examen o declaración juramentada) como se fija en 37CFR 1. 16 (f) de $130 para entidades no pequeñas, se debe remitir con los items faltantes identificados en esta carta.

RESUMEN DE LAS TASAS A PAGAR
Total de tasas adicionales requeridas para esta solicitud es de $1130 para grandes entidades.

- $300 tasa básica establecida por la ley

- $130 cargo adicional

- La tasa de búsqueda no sea apagado. El solicitante debe remitir $500 para completar la tasa de búsqueda.
- La tasa de examen no sea apagado. El solicitante debe remitir $200 para completar la tasa de examen para una entidad grande.

La respuesta se debe enviar a: Mail Stop Missing Parts
Commissioner for Patents
PO Box 1450
Alexandria,VA 22313-1450

Una copia de esta notificación se debe devolver con la respuesta.

Firma

Oficina de examen inicial de patentes (571) 272-4000, o uno-800-PTO-9199, o 1-800-972-6382
PARTE 1-COPIA ABOGADO/SOLICITANTe

60

SISTEMA DE ARCHIVO EXTENSIBLE

REFERENCIA CRUZADA CON OTRAS SOLICITUDES RELACIONADAS

Esta solicitud reivindica los beneficios de la Solicitud Provisional de los EU No. 60/637,407, llamada, FORMATO PARA SISTEMA DE ARCHIVOS PARA MEDIOS PORTÁTILES, presentada en diciembre 17, 2004. La solicitud provisional No. 60/637,407 se incorpora en el presente documento por referencia.

ANTECEDENTES

Hay una serie de dispositivos portátiles de computo generalmente descritos, , tales como cámaras digitales, cámaras de vídeo, programas de reproducción, teléfonos móviles, dispositivos de computo móviles, asistentes digitales personales, y cosas por el estilo que mantienen datos en un medio de almacenaje, tales como medios portátiles de almacenaje. El desarrollo continúo de dispositivos de cómputo portátiles más complejos y medios de almacenaje portátiles con mayor capacidad requieren de formatos de sistemas de archivo más flexibles para uso en los medios de almacenaje. Los formatos actuales de sistemas de archivos pueden ser deficientes en la flexibilidad que suministran para los medios de almacenaje de mayor capacidad y/o medios de almacenaje de aplicaciones.

RESUMEN

Un formato para un sistema de archivos extensible para medios de almacenaje portátiles. El formato para un sistema de archivos extensible incluye la especificación de tipos de entradas a un directorio principal y a uno secundario que pueden estar definidos a gusto. Los tipos de entrada al directorio principal y secundario se pueden clasificar adicionalmente como entradas críticas y benignas al directorio.

De acuerdo con un aspecto del presente invento, se provee un medio legible por computador que tiene componentes ejecutables por computador para almacenar datos. El componente legible por computador puede incluir un componente de parámetros de inicialización para especificar los parámetros de inicialización para un sistema de archivos. El componente legible por computador también puede incluir un componente de tabla de asignaciones para definir una tabla de asignaciones asociada con el sistema de archivos. Adicionalmente, el componente legible por computador incluye un componente de entrada al directorio principal para especificar datos en el directorio raíz del sistema de archivos. Aún más, el componente legible por computador incluye por lo menos un componente de entrada secundario que corresponde al componente de entrada al directorio principal. El componente secundario de entrada define los metadatos asociados con el componente de entrada al directorio principal. El componente de entrada al directorio principal y secundario se puede clasificar adicionalmente como crítica o benigna.

De acuerdo con otro aspecto del presente invento, se provee un medio legible por computador que tiene un componente ejecutables por computador para almacenar datos. El componente legible por computador incluye un componente de parámetros de inicialización para especificar los parámetros de inicialización para un sistema de archivos. El componente del medio legible por computador también incluye una tabla de asignación de archivos para definir una tabla de asignación de archivos asociadas con el sistema de archivos. Aún más, el componente del medio legible por computador incluye un componente de directorio raíz para especificar datos en el directorio raíz del sistema de archivos. Adicionalmente, el componente legible por computador incluye por lo menos un componente de metadatos extensible correspondiente a las entradas del directorio principal. El componente de meta datos define los meta datos asociados con el componente del directorio principal.

En una implementación ilustrativa, un sistema de archivos no monta un volumen para una entrada crítica o entrada del directorio principal que no se

reconoce. El sistema de archivos puede ignorar entradas al directorio principal benignas, entradas al directorio secundario críticas y entradas al directorio secundario benignas que no se reconocen.

Esté resumen se suministra para introducir una selección de componentes en forma simplificada que se describen más abajo en la Descripción Detallada. Este resumen no tiene la intención de identificar rasgos primordiales del tópico que se reivindica, ni tampoco tiene la intención de ser una ayuda en determinar el alcance del tema reivindicado.

DESCRIPCIÓN DE LOS DIBUJOS

Los aspectos anteriores y muchas de las ventajas de este invento se podrán apreciar mejor en la medida que se comprende mejor a través de las referencias a la siguiente descripción detallada y cuando se toman en conjunto con los dibujos que la acompañe, donde:

La FIGURA 1A-1C son diagramas de bloque que muestran un ambiente ejemplar incluyendo un dispositivo de cómputo portátil y un dispositivo de almacenaje que implementa el formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento;

La FIGURA 2 es un diagrama de bloque que muestran varios componentes de diseño de volumen correspondiente al formato del sistema de archivos extensible de acuerdo con un aspecto del presente invento;

LA FIGURA 3 es un diagrama de bloque ilustrativo de una estructura del directorio en un sistema de archivos extensible que incluye estructuras de entrada para un directorio principal y un directorio secundario de acuerdo con un aspecto del presente invento;

LA FIGURA 4 es un diagrama de bloque ilustrativo de un componente de datos para incrementar un proceso de bloque de inicialización en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento;

LA FIGURA 5 es un diagrama de bloque ilustrativo de un componente de datos para implementar entradas en un directorio en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento

LA FIGURA 6 es un diagrama de bloque ilustrativo de un componente de datos para implementar un nombre de archivo y extensión en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento;

LA FIGURA 7 es un diagrama de bloque ilustrativo de un componente de datos para implementar identificadores de volumen en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento;

LA FIGURA 8 es un diagrama de bloque ilustrativo de un componente de datos para implementar una entrada extensible de directorio en un formato de sistema de archivos extensibles de acuerdo con un aspecto del presente invento;

LA FIGURA 9 es un diagrama de bloque ilustrativo de un componente de datos para implementar una entrada en un directorio extensible en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento;

LA FIGURA 10 es un diagrama de bloque ilustrativo de un componente de datos para implementar una lista de control de acceso en un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento; y

LA FIGURA 11 es un diagrama de flujo ilustrativo de una rutina de creación de nombre de archivo para un formato de sistema de archivos extensible de acuerdo con un aspecto del presente invento.

DESCRIPCIÓN DETALLADA

Descrito en forma general, el presente invento se relaciona con un formato para sistemas de archivos extensibles y varios procesos asociados con el formato de sistema de archivos extensibles. En una implementación ilustrativa, el formato de sistema de archivos extensibles corresponde a un

formato de sistema de archivos extensibles para medios de almacenaje portátiles y los varios procesos asociados con el formato de sistema de archivos extensibles en los medios de almacenaje portátiles. Aunque el presente invento se describirá en relación a un formato de sistema de archivos para medios de almacenaje portátiles, uno diestro en el arte relacionado podrá apreciar que las implementaciones divulgadas son de naturaleza ilustrativa y no limitante. Adicionalmente, uno diestro en el arte relacionado podrá apreciar que las estructuras de datos y disposición de los datos utilizados en el ejemplo ilustrativo pueden requerir información adicional relacionada a la ejecución, seguridad, y cosas parecidas.

LA FIGURA 1A-1C son diagramas de bloque ilustrativos de varios ambientes operativos 100 para el formato de sistema de archivos extensibles del presente invento. Con referencia a la figura 1A, en una implementación ilustrativa, el formato de sistema de archivos extensibles se utiliza para almacenar datos de un dispositivo de cómputo, tal como un dispositivo de cómputo móvil 102, o un medio de almacenaje tal como en el medio de almacenaje portátil 104. En una implementación ilustrativa, el dispositivo de cómputo móvil 102 puede corresponder a cualquier variedad de dispositivos de cómputo, incluyendo pero no limitado a, dispositivos de cómputo portátiles, teléfonos móviles, asistentes digitales personales, reproductores de música, reproductores de medios. Los medios de almacenaje portátiles también puede incluir, pero no está limitado a, discos duros, medios flash, micro-drives y otros medios de almacenaje. En una implementación ilustrativa, el sistema de archivos extensibles en el medio de almacenaje portátiles 104 no tiene que incluir ningún tipo de componente de software legible o exportable, tal como un ambiente operativo utilizado por el dispositivo de cómputo móvil 102. En forma alterna, el sistema de archivos extensible en el medio de almacenaje portátiles 104 puede incluir componentes de software legibles o ejecutables utilizados por el dispositivo móvil 102.

En una implementación ilustrativa, el dispositivo de cómputo móvil 102 puede estar comunicado con otro dispositivo de cómputo para recolectar/intercambiar datos para almacenar en el medio de almacenaje portátil 104. En referencia a la figura 1B, el dispositivo de cómputo móvil 102 puede estar en comunicación directa con otro dispositivo de cómputo 106 y medios de almacenaje 108. En una implementación ilustrativa, la comunicación directa puede corresponder a varios métodos de comunicación alambrada e inalámbrica. En una implementación ilustrativa, los otros medios de almacenaje 108 no se tienen que formatear de acuerdo con el formato de sistema de archivos extensibles del presente invento. En referencia a la figura 1C, de forma similar, el dispositivo de cómputo móvil 102 puede estar comunicado con otro dispositivo de cómputo 110 y medios de almacenaje 112 a través de una conexión de red. En una implementación ilustrativa, la conexión de red puede corresponder a una red de área local (LAN) o a una red de área amplia (WAN).

En referencia ahora a la figura 2, se describe una implementación ilustrativa del diseño de volumen 200 para un formato de sistema de archivos extensibles. El diseño de volumen 200 incluye un componente de parámetros de inicialización 202 que incluyen información relacionada a una descripción de los parámetros del sistema de archivos de la partición. En una implementación ilustrativa, el componente de parámetros de inicialización 202 puede incluir el código de arranque de una partición definida, parámetros fundamentales para un sistema de archivos para la partición definida, y varios tipos de información de chequeo de errores. Una estructura de datos para definir por lo menos una porción de los parámetros de inicialización se describirá más abajo en referencia a la figura 4.

El diseño del volumen 200 también incluye un componente de parámetros extensible, designado como parámetros OEM 204, que definen varias estructuras de datos adicionales utilizadas en conjunto con el sistema de archivos. En una implementación ilustrativa, una manufactura original de

equipo (OEM) puede especificar varias estructuras de datos extensibles, tales como parámetros de ejecución para un medio de almacenaje, que se puede definir en el momento de la fabricación. El diseño de volumen 200 puede incluir adicionalmente un componente de tabla de asignación de archivos 206 que define la asignación de archivos y directorios. En una implementación ilustrativa, cada entrada en el componente de la tabla de asignación de archivos 206 corresponde a una entrada de 32-bit que representa a un grupo no asignado o a un grupo que no se puede utilizar. El diseño de volumen 200 puede incluir adicionalmente una serie de componentes de datos de archivos 208A-208X que corresponden a los datos almacenados de acuerdo al formato del sistema de archivos. Varias estructuras de datos para definir una porción del componente de datos de archivos 208A-208X se definirá en relación a la figura 3-10.

Volviendo a la figura 3, en un aspecto, el componente de datos de archivos 208 puede incluir una o más entradas de directorio de acuerdo a la estructura de directorios 300. En una implementación ilustrativa, la estructura de directorio 300 se organiza en entradas al directorio principal 302 y entradas al directorio secundario 304. Cada entrada de directorio principal y secundario se tipifica. Por ejemplo, en una implementación ilustrativa, los valores de tipo para las entradas a los directorios principal y secundario puede corresponder a un rango de 1-255. Las entradas al directorio principal 302 corresponden a las entradas al directorio raíz del sistema de archivos. Las entradas al directorio secundario 304 le siguen a una entrada al directorio principal y están asociadas con la entrada en el directorio principal. Las entradas al directorio secundario extienden los metadatos asociados con la entrada al directorio principal correlacionada.

En referencia aun a la figura 3, en una implementación ilustrativa, las entradas al directorio principal 312 se pueden clasificar adicionalmente como entradas al directorio principal críticas 306 y entradas al directorio principal benignas 308. Las entradas críticas al directorio principal 306 definen formatos

potencialmente diferentes para cada entrada de directorio. En una implementación ilustrativa, un ambiente operativo no monta un volumen correspondiente al formato del sistema de archivos extensibles con una entrada críticas al directorio principal, como se describirá más abajo. Ejemplos de entradas al directorio principal conocidas 306 pueden incluir asignaciones de bitmaps, tablas, etiquetas de volumen, llaves codificadas, y entradas normales al directorio. Las entradas benignas al directorio principal 308 también definen formatos potencialmente diferentes para cada entrada de directorio, pero pueden ser ignoradas por el sistema de archivos si una entrada particular benigna al directorio principal no se entiende. Las entradas benignas al directorio principal 308 se pueden asociar con otro grupo encadenado del volumen. Adicionalmente, las entradas benignas al directorio principal 308 se pueden asociar con un número de entradas al directorio secundario 304.

En una manera similar a las entradas al directorio principal 302, las entradas al directorio secundario 304 se pueden clasificar adicionalmente como entradas críticas al directorio secundario 310 y entradas benignas al directorio secundario 312. Como se describió anteriormente, las entradas críticas al directorio secundario 310 y entradas benignas al directorio secundario 312 están asociadas con una entrada benigna al directorio principal y extienden los metadatos asociados con la entrada al directorio principal. Ambas, las entradas críticas al directorio secundario 310 y las entradas benignas al directorio secundario 312 se pueden asociar con otro grupo encadenado en el volumen.

Para montar una correspondencia con el formato de sistema de archivos extensibles, el sistema de archivos implementa un procedimiento de montaje de volumen. En una implementación ilustrativa, el procedimiento de montaje del volumen intenta mirar el número de versión del volumen. Si el número de la versión no se entiende (por ejemplo el número de la versión es más alto), el volumen no se monta. Durante la enumeración normal del directorio, cualquier entrada crítica al directorio principal que no la conozca el sistema de archivos evita que el volumen se monte. Luego, varios procesos iniciados por el usuario, tales como abrir archivo, hacen que el sistema de

archivos enumere las entradas al directorio secundario. Si las entradas críticas al directorio secundario 310 no las conoce el sistema de archivos, la entrada total al directorio se omite. Adicionalmente, si las entradas benignas al directorio secundario 312 no las conoce el sistema de archivos, el directorio secundario particular desconocido benigno, se ignora.

En referencia el ahora a la figura 4, se describe un diagrama de bloque ilustrativo del componente de datos 400 para implementar el bloque de proceso de inicialización en el componente de parámetros de inicialización 202 (figuras 2). El componente de datos 400 incluye un componente de nombre 402 OEM para especificar un nombre para el formato de sistema de archivos del medio de almacenaje. El componente de datos 400 también incluye un componente de descripción de tamaño de datos 404 para especificar las características varias de los datos almacenados en el sistema de archivos. Por ejemplo, el componente que describe el tamaño de los datos 404 puede especificar un conteo de bits por sector, el número de sectores por unidad de asignación, una tabla FAT, y un conteo de sectores para todas las estructuras de datos. El componente de datos incluye un componente activo de banderas FAT 406 para especificar el número activo de FATs en el sistema de archivos. En una implementación ilustrativa, un sistema de archivos puede ser compatible con múltiples FAT para utilizarse con algunos ambientes de sistema operativo. El componente de datos 400 puede adicionalmente incluir un componente de identificación de volumen 408 para identificar un número de serial de volumen y/o un número de versión. Aún más, el componente de datos 400 puede incluir un tipo de sistema de archivos para especificar el formato de sistema de archivos para el sistema de archivos. Uno diestro en el arte relevante podrá apreciar que el componente de datos 400 puede incluir un número adicional/alternativo de filas para implementar el componente identificado anteriormente 402-410 y los componentes adicionales.

Regresando ahora a la figura 5, un diagrama de bloque ilustrativo describe el componente de datos 500 para implementar entradas al directorio

en un formato de sistema de archivos extensible. Regresando ahora a la figura 6, un diagrama de bloque describe el componente de datos 500 para implementar un nombre de archivo y extensión. El componente de datos 500 incluye un componente interno 502 para especificar si una entrada de directorio particular está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 500 incluye adicionalmente un componente de designación de tipo 504 para especificar que una entrada de directorio está asociada con una entrada de directorio normal. El componente de datos 500 incluye adicionalmente un componente de entrada de directorio secundario 504 para especificar un número de entradas secundarias asociadas con la entrada normal al directorio. El componente de datos 500 incluye un componente de atributos de archivo 508 para especificar los atributos varios del sistema de archivos para la entrada al directorio. Aún más, el componente de datos 500 incluye un componente de tiempo 510 para especificar la información de tiempo variada tal como la creación de una marca de tiempo, marca de modificación de tiempo y otro tipo de información sobre el tiempo. Adicionalmente, el componente de datos 500 incluye adicionalmente un componente de zona de tiempo 512 para especificar una zona de tiempo para la ultima marca de tiempo creada. Uno diestro en el arte relevante podrá apreciar que el componente de datos 500 puede incluir un número adicional/alternativo de filas para implementar el componente identificado anteriormente 502-512 y los componentes adicionales.

Volviendo ahora a la figura 6, un diagrama de bloque describe el componente de datos 600 para implementar un nombre y extensión de archivo. El componente de datos 600 incluye un componente interno 602 para especificar si una entrada de directorio particular está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 600 incluye adicionalmente un componente de designación de tipo 604 para especificar que la entrada al directorio está asociada con un nombre en el sistema de archivos.

El componente de datos incluye adicionalmente un componente de nombre largo de archivo 606 y un componente de nombre de archivo 608. La utilización del componente de nombre de archivo 608 se describirá más abajo. El componente de datos 600 también incluye un componente de nombre de archivo 610 para especificar el nombre del archivo. Uno diestro en el arte relevante podrá apreciar que el componente de datos 600 puede incluir un número adicional/alternativo de filas para implementar los componentes identificados anteriormente 602-610 y componentes adicionales. Adicionalmente, un nombre de archivo de entrada de directorio se puede extender con una entrada secundaria al directorio.

Volviendo ahora a la figura 7, se muestra un diagrama de bloque ilustrativo del componente de datos 700 para implementar un identificador de volumen en un formato de sistema de archivos extensibles. El componente de datos 700 incluye un componente interno 702 para especificar si una entrada particular de directorio está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 700 incluye adicionalmente un componente de designación de tipo 704 para especificar que la entrada al directorio está asociada a un identificador de volumen. El componente de datos 700 incluye adicionalmente un componente de entrada al directorio secundario 706 para especificar un número de entradas secundarias asociadas con el identificador de volumen. El componente de datos 700 también incluye un identificador de volumen 708, tal como un identificador único global. Uno diestro en el arte relevante podrá apreciar que el componente de datos 700 puede incluir un número de filas adicional/alternativo para implementar los componentes identificados anteriormente 702-708 y componentes adicionales. Adicionalmente, en una implementación ilustrativa, el componente de datos 700 corresponde a una entrada de directorio benigna que el sistema de archivos puede ignorar si no es compatible con el identificador de volumen.

En referencia ahora a las figuras 8 y 9, en una implementación ilustrativa, partes, tales como OEM, pueden definir tipos de entrada al directorio principal benignas 308 y tipos de entrada al directorio secundario benignas 312. Tal como se mencionó anteriormente, en el evento de que el sistema de archivos no reconozca o entienda ya sea el tipo de entrada benignas al directorio principal 308 o el tipo de entrada de benigna al directorio secundario 312, el sistema de archivos puede ignorar el tipo de entrada al directorio definida.

En referencia a la figura 8, un diagrama de bloque ilustrativo describe el componente de datos 800 para implementar una entrada de directorio principal benignas extensible 308 en un formato de sistema de archivos extensibles. El componente de datos 800 incluye un componente interno 802 para especificar si una entrada particular de directorio está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 800 puede incluir adicionalmente un componente de designación de tipo 804 para especificar que la entrada al directorio es una entrada benigna al directorio principal. El componente de datos 800 incluye adicionalmente un componente de entrada al directorio secundario 806 para especificar un número de entradas secundarias asociadas con el identificador de volumen. El componente de datos 800 también incluye un identificador de volumen 808, tal como un identificador único global. El componente de datos 800 puede incluir información adicional 810, tal como información de verificación y un grupo de inicio. Una persona diestra en el arte relevante podrá apreciar a que el componente de datos 800 puede incluir un número adicional/alternativo de filas para implementar los componentes identificados anteriormente 802-506 y componentes adicionales.

En referencia a la figura 9, un diagrama de bloque ilustrativo describe el componente de datos 900 para implementar una entrada benigna al directorio secundario en un formato de sistema de archivos extensibles. El componente de datos 900 incluye un componente interno 902 para especificar si una

entrada de directorio particular está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 900 incluye adicionalmente un componente de designación de tipo 904 para especificar que la entrada al directorio es una entrada al directorio principal benigna. El componente de datos 900 incluye adicionalmente un componente de entrada al directorio secundario 906 para especificar un número de entradas secundarias asociadas con el identificador de volumen. El componente de datos 900 también incluye un identificador de volumen 908, tal como un identificador único global. El componente de datos 900 puede incluir información adicional 910, tal como información de verificación y un grupo de inicio. Una persona diestra en el arte podrá apreciar que el componente de datos 900 puede incluir un número adicional/alternativo de filas para implementar los componentes identificados anteriormente 902-906 y componentes adicionales.

En una implementación ilustrativa, una entrada benigna al directorio principal y/o entrada al directorio secundario se puede asociar con una lista de información de control de acceso (ACL). La figura 10 es un diagrama de bloque ilustrativo del componente de datos 1000 para implementar una lista de control de acceso en un formato de sistema de archivos extensibles. El componente de datos 1000 incluye un componente interno 1002 para especificar si una entrada de directorio particular está en uso. En una implementación ilustrativa, el bit alto del componente de datos se fija en "1" si la entrada al directorio está en uso. El componente de datos 1000 incluye adicionalmente un componente de designación de tipo 1004 para especificar que la entrada al directorio es una entrada ACL. El componente de datos 1000 incluye adicionalmente un número de campos ACL 1006, tal como banderas ACL, indicadores de bases de datos ACL, y cosas parecidas. Una persona diestra en el arte relevante podrá apreciar que el componente de datos 1000 puede incluir un número adicional/alternativo de filas para implementar los componentes identificados anteriormente 1002-1006 y componentes adicionales.

En referencia ahora a la figura 11, se describe una rutina de creación de nombre de archivo 1100 para un formato de sistema de archivos extensibles. En el lote 1102, un sistema de archivo obtiene una petición para crear una entrada de directorio con un nombre de archivo específico. En una implementación ilustrativa, el nombre de archivo específico puede corresponder a una convención de nombres, tal como la convención en los nombres de las fotos de una cámara digital. En el lote 1104, el sistema de archivos genera un nombre de objetivo. En el lote 1106, un círculo repetitivo se comienza examinando el valor siguiente de entrada al directorio. En un tipo ilustrativo la entrada al directorio para almacenar los valores de entrada al directorio se describe en relación al componente de datos 600 (figura 6).

En el bloque de decisión 1108, se lleva a cabo una prueba para determinar si el valor del objetivo corresponde al valor actual de entrada al directorio. Sino corresponde, la rutina 1100 regresa al bloque 1106 (hasta que todas las entradas al directorio se hayan examinado). Si el valor corresponde al bloque decisión 1108, en el lote 1110, el sistema de archivos obtiene el nombre completo para la entrada potencial correspondiente al directorio. Un tipo de entrada al directorio ilustrativa para almacenar el nombre completo de la entrada al directorio se describe en relación al componente de datos 600 (figura 6). En el bloque decisión 1112, se lleva a cabo una prueba para determinar si el nombre de archivo del objeto corresponde al nombre completo de la entrada correspondiente potencial al directorio. Si lo es, la rutina 1100 termina reportando un conflicto y el sistema de archivos tienen que seleccionar un nuevo nombre. Si el archivo completo no corresponde, la rutina 1100 regresa al bloque 1106 para continuar el chequeo de valores para todas las entradas al directorio de sistema de archivos.

De acuerdo con un aspecto del presente invento, varias funcionalidades adicionales se pueden añadir a través de la especificación de un tipo de directorio específico. Por ejemplo, flujos de nombre se pueden soportar especificando una entrada de directorio de flujo de nombre. Adicionalmente, la

codificación del disco también se puede apoyar a través de la utilización de un algoritmo específico de codificación e intercambio de llaves. Aún más, la conversión de las zonas horarias se puede asociar con entradas de directorio para convertir automáticamente la zona horaria actual con la zona horaria en la que se hizo la entrada al directorio.

Mientras se han mostrado implementaciones ilustrativas y se han descrito, se pueden apreciar que varios cambios se pueden hacer sin alejarse del espíritu del alcance del invento.

La implementación del invento en el cual se reivindica la exclusividad y los privilegios exclusivos se define de la siguiente manera:

1. Un medio legible por computador que tiene componentes para almacenar datos, los componentes legibles por computador constante de;

Un componente de entrada a un directorio principal para especificar datos en un directorio raíces del sistema de archivos; y

Por lo menos un componente de entrada secundario correspondiente al componente de entrada a directorio principal y que define los meta datos asociados con el componente de entrada al directorio principal.

2. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio principal es un componente de entrada crítico al directorio principal que el sistema de archivos debe entender.

3. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio principal es un componente de entrada benigno al directorio principal que se puede ignorar si el sistema de archivos lo desconoce.

4. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio secundario es una entrada crítica al directorio secundario que hace que el componente del directorio

principal y el directorio secundario se ignoren si son desconocidos al sistema de archivos.

5. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio secundario es una entrada benigna al directorio secundario que se puede ignorar si es desconocida por el sistema de archivos.

6. El componente del medio legible por computador de la reivindicación 1 consta además del componente de entrada secundario correspondiente al componente de entrada al directorio principal y que define los metadatos asociados con la entrada al directorio principal.

7. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio principal corresponde al bitmaps de asignación que define la disponibilidad de los grupos dentro del medio de almacenaje.

8. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio principal corresponde a un identificador de volumen.

9. El componente del medio legible por computador de la reivindicación 1, donde el componente de entrada al directorio principal corresponde a un identificador de nombre de archivo.

10. El componente del medio legible por computador de la reivindicación 9, donde, el identificador del nombre de archivo incluye un nombre completo de archivo y un nombre de archivo.

11. El componente del medio legible por computador de la reivindicación 1, donde por lo menos una entrada al directorio secundario corresponde a una entrada al directorio secundario extensible.

12. El componente del medio legible por computador de la reivindicación 1, comprende además un componente de datos de manufactura que especifica las estructuras de datos de manufactura.

13. Un medio legible por computador que tiene componentes ejecutables por computador para almacenar datos, el componente de medio legible por computador consta de:

Un componente de parámetros de inicialización para especificar los parámetros de inicialización para un sistema de archivos

un componente de tabla de asignación de archivos para definir una tabla de asignación de archivos asociada con el sistema de archivos;

Los medios para especificar datos en un directorio raíces de un sistema de archivos y meta datos asociados con los datos en el directorio principal.

14. El componente del medio legible por computador de la reivindicación 13, donde los medios para especificar datos en el directorio raíces de un sistema de archivos y los metadatos asociados con los datos en el directorio principal corresponden a un componente de entrada al directorio primario para especificar datos en el sistema de archivos del directorio raíz.

15. El componente del medio legible por computador de la reivindicación 14, donde el componente del directorio principal puede corresponder a una entrada crítica al directorio principal.

16. El componente del medio legible por computador de la reivindicación 14, donde el componente del directorio principal puede corresponder a una entrada benigna al directorio principal.

17. El componente del medio legible por computador de la reivindicación 13, donde los medios para especificar datos en el directorio raíces del sistema de archivos y los metadatos asociados con los datos en el directorio raíces incluyen por lo menos un componente de entrada secundario correspondiente a los datos en el directorio raíz y que define los metadatos asociados con la entrada en el directorio principal.

18. Un medio legible por computador con componentes ejecutables por computador para almacenar datos, el componente de medios legibles por computador consta de:

un componente de parámetros de inicialización para especificar los parámetros de inicialización para un sistema de archivos;

un componente de tabla de asignación de archivos para definir una tabla de asignación de archivos asociada con el sistema de archivos;

un componente de directorio raíces para especificar datos en el directorio raíces del sistema de archivos; y

por lo menos un componente extensible de meta datos correspondiente al componente de entrada al directorio principal y que define los meta datos asociados con el componente del directorio raíz.

19. El componente legible por computador de la reivindicación 18, donde el componente del directorio raíces es un componente crítico del directorio raíces que el sistema de archivos debe entender.

20. El componente del medio legible por computador de la reivindicación 18, donde el componente de entrada al directorio principal es un componente de entrada benigna o al directorio raíz que se puede ignorar si es desconocido al sistema de archivos.

RESUMEN

Se provee un formato de sistema de archivos extensibles para medios de almacenaje portátiles. El formato del sistema de archivos extensibles incluye la especificación de tipos de entradas al directorio principal y secundario que se pueden definir a gusto. Los tipos de entrada al directorio principal y secundario se pueden clasificar adicionalmente como entradas al directorio críticas y benignas.

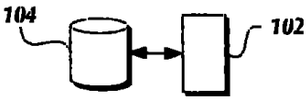


Fig.1A.

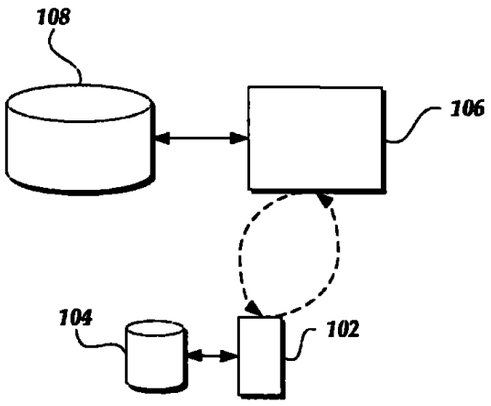


Fig.1B.

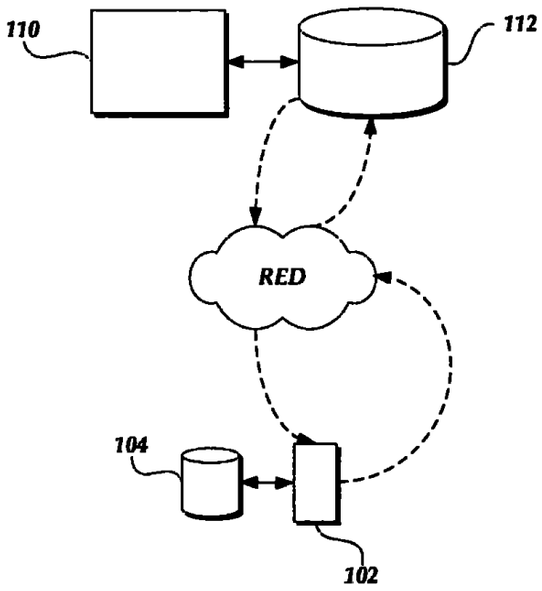


Fig.1C.

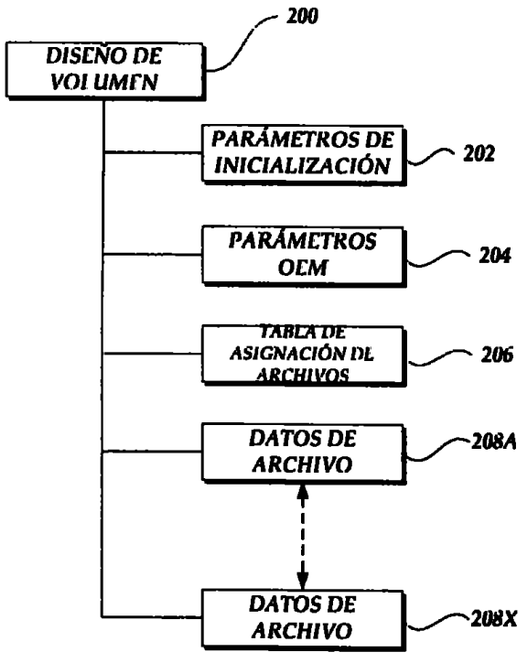

200
DISEÑO DE VOLUMEN
PARÁMETROS DE INICIALIZACIÓN
202
PARÁMETROS OEM
204
TABLA DE ASIGNACIÓN DE ARCHIVOS
206
DATOS DE ARCHIVO
208A
DATOS DE ARCHIVO
208X

Fig.2.

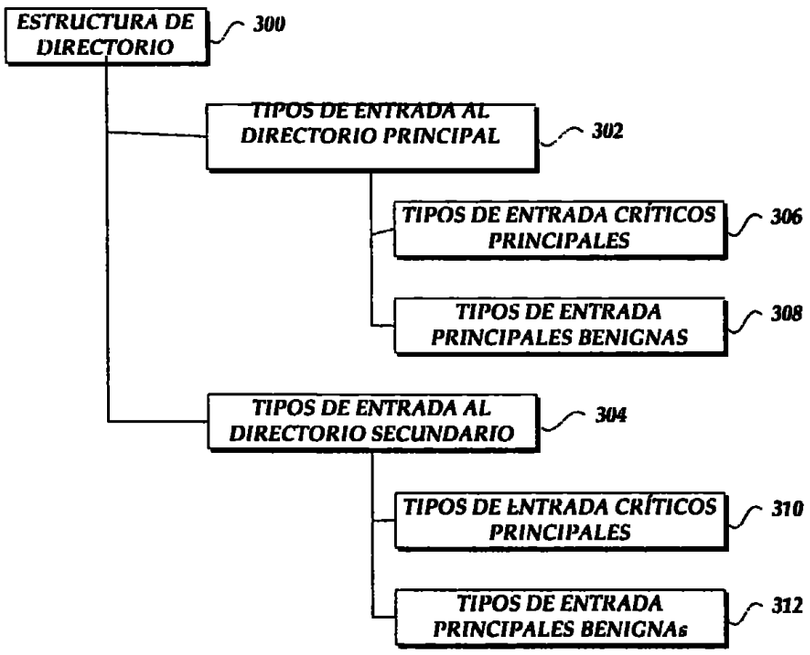

ESTRUCTURA DE DIRECTORIO
300
TIPOS DE ENTRADA AL DIRECTORIO PRINCIPAL
302
TIPOS DE ENTRADA CRÍTICOS PRINCIPALES
306
TIPOS DE ENTRADA PRINCIPALES BENIGNAS
308
TIPOS DE ENTRADA AL DIRECTORIO SECUNDARIO
304
TIPOS DE ENTRADA CRÍTICOS PRINCIPALES
310
TIPOS DE ENTRADA PRINCIPALES BENIGNAs
312

Fig.3.

NOMBRE	TAMAÑO	
NOMBRE OEM	3	402
DESCRIPTOR DE TAMAÑO DE DATOS	X	404
FAT ACTIVO	2	406
NO SERIAL DE VOLUMEN	4	408
TIPO DE SISTEMA DE ARCHIVOS	X	410

Fig.4.

NOMBRE	TAMAÑO	
EN USO	11	502
TIPO	17	504
ENTRADAS SECUNDARIAS	1	506
ATRIBUTOS	2	508
TIEMPO	X	510
ZONA HORARIA	1	512

Fig.5.

NOMBRE	TAMANO	
EN USO	11	602
TIPO	17	604
CARACTERES	1	606
NOMBRE HASH	2	608
NOMBRE DE ARCHIVO	28	610

Fig.6.

NOMBRE	SIZE	
EN USO	11	702
TIPO	17	704
ENTRADAS SECUNDARIAS	1	706
GUID	16	708

Fig.7.

NOMBRE	SIZE	800
EN USO	11	802
TIPO	17	804
ENTRADAS SECUNDARIAS	1	806
GUID	16	808
OTROS	X	810

Fig.8.

NOMBRE	SIZE	900
EN USO	11	902
TIPO	17	904
GUID	16	906
OTROS	X	910

Fig.9.

NOMBRE	TAMAÑO	1000
EN USO	11	1002
TIPO	17	1004
INFORMACIÓN ACL	X	1006

Fig.10.

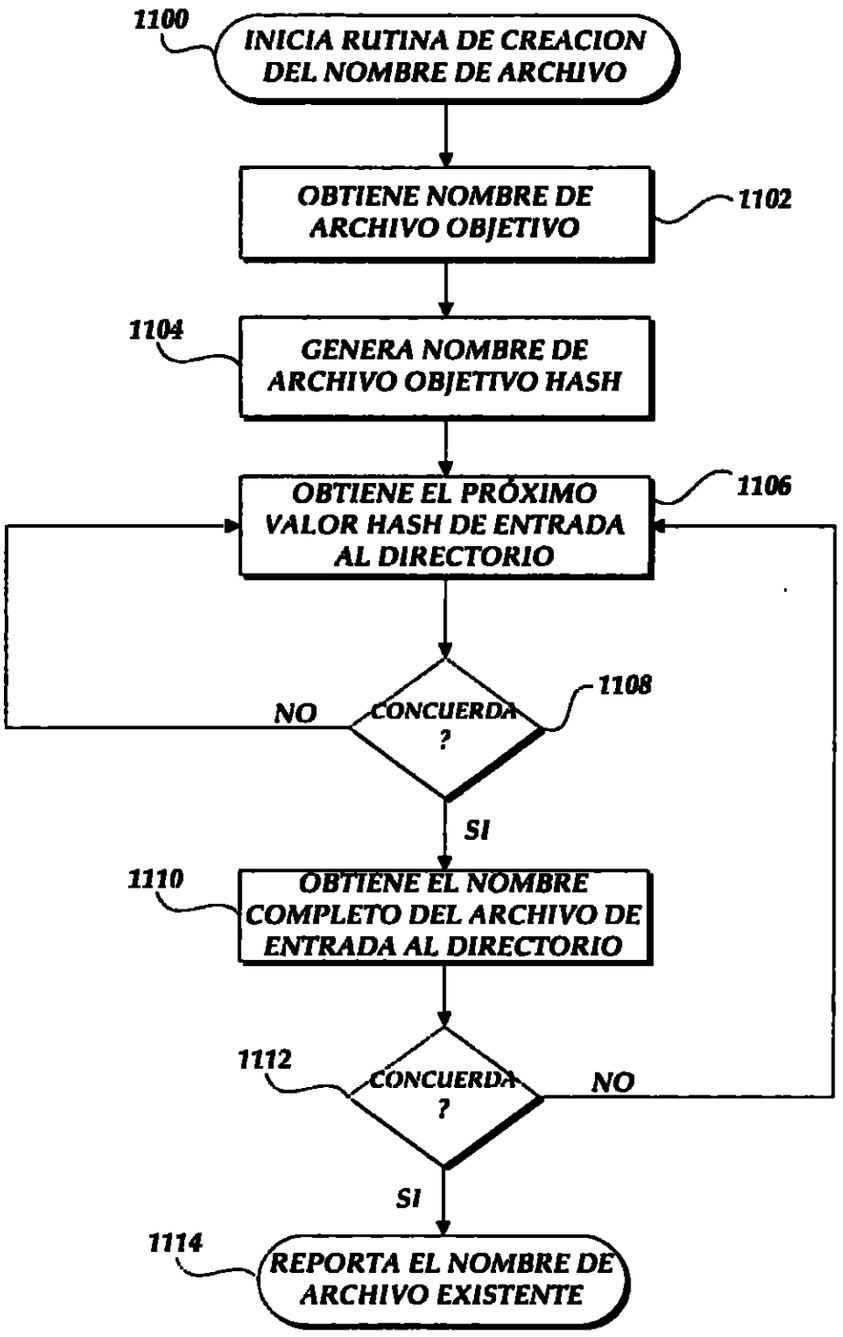

1100
INICIA RUTINA DE CREACION DEL NOMBRE DE ARCHIVO
OBTIENE NOMBRE DE ARCHIVO OBJETIVO
1102
1104
GENERA NOMBRE DE ARCHIVO OBJETIVO HASH
OBTIENE EL PRÓXIMO VALOR HASH DE ENTRADA AL DIRECTORIO
1106
NO
CONCUERDA ?
1108
SI
1110
OBTIENE EL NOMBRE COMPLETO DEL ARCHIVO DE ENTRADA AL DIRECTORIO
1112
CONCUERDA ?
NO
SI
1114
REPORTA EL NOMBRE DE ARCHIVO EXISTENTE

Fig.11.

FORMATO PARA SISTEMA DE ARCHIVOS PARA MEDIOS DE ALMACENAJE PORTÁTILES

CAMPO DEL INVENTO

El presente invento se relaciona con software y en particular con un protocolo para sistemas de archivos.

1. ANTECEDENTES DEL INVENTO FAT32 es el sistema de archivos preferido para los medios portátiles (tales como dispositivos basados en flash NAND que se utilizan comúnmente en cámaras, teléfonos celulares, etc.). La capacidad de estos dispositivos se espera que llegue a 32GB+ en los próximos años.

2. Se necesitan incrementos en la velocidad de asignación- FAT 32 tiene una tabla de búsqueda lineal.

3. El tamaño de los archivos en FAT 32 está limitado a 4GB.

4. La expansibilidad de FAT 32: FAT 32 no se puede extender fácilmente ahora puesto que todos los bits de entrada en el directorio se utilizan. El formato de nombre largo utiliza una máscara de bit que previene cualquier extensión adicional basada en máscaras de bit en las entradas del directorio.

DESCRIPCIÓN DETALLADA DE LA IMPLEMENTACIÓN PREFERIDA

DISEÑO FAT++

Un volumen FAT++ tiene el siguiente diseño.

Nombre	Desbalance (sector)	Tamaño (sectores)	Descripción
Sectores de arranque Y parámetros del BIOS Parámetro Bloque (BPB)	0	1	Este sector contiene el código de arranque de ésta partición y parámetros fundamentales del sistema de archivos que describen el tipo y el tamaño del sistema de archivos en ésta partición. El código para este sector se carga y se ejecuta en el Archivo Maestro de Arranque. Los cuatro últimos bits deben ser los OxAA550000.

Nombre	Desbalance (sector)	Tamaño (sectors)	Descripción

Extendido
Sector de arranque 1
	1	2	Este sector contiene un código adicional de arranque para la partición, y una estructura FSINFO
			con un desbalance Ox 1 E4:
			DWORD 0x37373937 -- la firma de
			FSinfo (esto difiere de FAT32)
			DWORD EraseBlockSize -- tamaño del bloque eliminado para dispositivos expresado en número de sectores. Para discos duros esto debe ser 1.
			// se pueden apoyar un total de 10 estructuras de parámetros diferentes OEM.
			struct {
			// MS assigned oem types
			DWORD OemParameterType;
			// OEM specific param structure
			UCHAR OemParameters[44]
			} OemParameterArea[10];
			Los OEMs, por ejemplo, pueden añadir parámetros de ejecución para dispositivos flash. Estos se pueden añadir con un código de utilidades de formateo OEM específico en implementaciones OEM de especificaciones FAT.
			Estos serán ignorados por las implementaciones de Windows.
			UCHAR Reservado[20];
			Los 4 últimos bits deben ser OxAA550000 para ambos sectores .
			Primeros 4 bits del segundo sector
			deben ser 0x73739373

Nombre	Desbalance (sectores)	Tamaño (sectores)	Descripcion
Reservado I	3	3	Ceros
Arranque de respaldo	6	3	Copia exacta de los sectores 0,1,2.
Reservado2	9	NumReserved-9	Sectores reservados. El numero de sectores reservado esta en el BPB. Esta área se reserva para operaciones futuras del sistema operativo, y no se deben modificar o utilizar. NumReserved generalmente es *32* sectores, pero NO siempre sera 32.
Tabla de asignación de archivos (FAT)	NumReserved	Varia	Una tabla de entradas de 32-bit que define la asignación de archivos y directorios. Cada entrada representa ya sea un grupo asignado, un grupo no asignado, o un grupo no utilizable. Cuando se define un grupo asignado, el valor para esta entrada indica la entrada FAT para el siguiente grupo en el directorio de archivos, o el final de la cadena de asignaciones.
Grupo	...	Varia	El resto de la partición son datos de archivo, divididos en asignaciones con tamaños de grupo como se definen en el BPB.

NOTAS SOBRE LA REGIÓN DEL SISTEMA DE ARCHIVOS

La siguiente sección suministra notas sobre las regiones definidas por FAT++.

IMPORTANTE: todas los campos "reservados" más abajo están definidos actualmente como cero. Sin embargo, no se debe asumir que estos campos contengan este o cualquier otro valor. Las utilidades de disco y otros programas deben mirar o modificar éstos campos reservados. Es la responsabilidad de la utilidad preservar cualquier valor contenido en estos campos.

EL SECTOR DE ARRANQUE Y EL BPB

Nombre	Desbalance (byte)	Tamaño (bites)	Descripción
jmpBoot	0	3	Instrucciones jmp al código de "arranque"
OEMName	3	8	"MSWIN5.0"
Bytes por sector	11	2	Conteo de bits por sector.
Sectores por grupo	13	1	Número de sectores por unidad de asignación.
Sectores Reservados	14	2	Número de sectores reservados luego de extender el récord de arranque más el tamaño del récord de arranque y la extensión al récord de arranque. Éstos deben estar acondicionados en forma apropiada para qué el grupo de arranque se encuentre en el límite deseado -para dispositivos flash, eso seria aproximarla al límite de un bloque eliminado. Esto le permite al área de datos en el sistema de archivo al cual se le escribe frecuentemente-para comenzar en un bloque eliminado alineado en forma apropiada.
Números de FATs	16	1	2
Entradas Reservadas	17	2	Reservado
Cero	19	2	Debe ser cero (este es el antiguo campo del sector bs field).
Descriptor de medios	21	1	Definido por el OEM
Reservado	22	2	Debe ser cero
Sectores por pista	24	2	Sectores por pista para la interrupción 13h

Nombre	Desbalance	Tamaño (bytes)	Descripción
Conteo de cabezas	26	2	Numero de cabezas para el interrupt 13h
Conteo de sectores escondidos	28	4	Sectores i 1000 escondidos en la partición de tal manera que el iniciador de arranquen se puede hacer el int un 13h.
Conteo de sectores	32	4	Número total de sectores para todas las estructuras de datos y datos para este drive.
Sectores grandes	36	4	Conteo de sectores DWORD en una sola tabla de asignación de archivos (FAT).
Banderas extendidas	40	2	Bits 0-3 : número activo de FAT. Esto definitivamente indica que FAT esta activo. 1.) Si es 0, el FAT principal está en uso. 2.) Si es 1, el FAT secundario se considera el máster. Esto no puede utilizar las implementaciones TFAT++ para indicar el FAT consistente. 3.) Si es 2, entonces el volumen está en un estado "sucio" por ejemplo el FAT principal es el correcto pero un inspectora de consistencia fILESYSTEM tiene que ejecutar sede para asegurar que la estructura FATs/bitmaps/directorio es consistente. Bits 4-15: reservados, deben ser 0
Versión FS	42	2	El bit alto es el número mayor de revisión. El byte menor es el número menor de revisión. Este documento define la versión a 1:0. **Nota: Las utilidades de disco deben respetar éste campo y no operar en volúmenes con un número de versión mayor o menor que aquella para el cual fueron diseñados.**

Nombre	Desbalance (byte)	Tamaño (bytes)	Descripción
Primer Grupo de raíces	44	4	Número de grupo del primer grupo del directorio raíz. Puede ser 2, pero no necesariamente. **Nota: las utilidades de disco que cambian la localización del directorio raíces deben hacer un esfuerzo para colocar primer grupo del directorio raíces en el primer grupo no-dañado en el drive (i.e. En el grupo 2, al menos de que esté marcado como dañado).**
Sector de información de número FS	48	2	Número de sector de la estructura FS info. Generalmente 1. Nótese que habrá una copia de la estructura FSINFO en el BackupBoot, **Pero únicamente la copia indicada por éste campo se mantendrá actualizada- i.e. Tanto el récord de arranque primario y de respaldo indicarán el mismo sector FSINFO.**
Respaldo de arranque Número sectores del Record	50	2	Si no es cero indica el número de sector de una copia del récord de arranque. Generalmente 6.
Reservado	52	12	Reservado para expansiones futuras.
Número de Drive	64	1	Número de drive Int 13h (e.g. 80h).
Reservado	65	1	Reservado (utilizado por Windows NT).
Firma de arranque	66	1	Firma de arranque extendida (29h).
Volumen ID	67	4	Número serial del volumen
Etiqueta de Volumen	71	11	Etiqueta del Volumen

Nombre	Desbalance (byte)	Tamaño	Descripción
Tipo de Sistema de archivos	82	8	"FAT++"

El FAT++ estrictamente forzará el número de versión FS requerido, tipo de sistema de archivos y el tipo de partición idéntico a la descripción anterior. Cualquier inconformidad resultará en el no reconocimiento del volumen -o peor, reconocido potencialmente como FAT32.

TABLA DE ASIGNACIÓN DE ARCHIVOS(FAT)

La tabla de asignación de archivos está empacada dentistas de entradas de 32-bits que tienen mapeo con los grupos de datos uno-a-uno. Cada entrada se define como:

Desbalance (byte)	Tamaño (bits)	Nombre	Descripción
0	28	Valor del grupo	0x00000000 grupo no localizado 0x00000001 valor inválido 0x00000002 - *OxOFFFFFF7* localización en la cadena OxOFFFFFF7 marca de grupo dañado *(pero únicamente si no es parte de una cadena de grupos o no es un número de grupo legal.* *NOTA: esto quiere decir que este puede ser un número de grupo válido en la cadena de grupos!)* OxOFFFFFF8 - OxOFFFFFFF fin del Archivo
29	4	Reservado	Reservado para uso futuro.

No hay cambio en la estructura FAT del FAT32.

BITMAP DE ASIGNACIÓN

Adicionalmente al FAT, el formato también le permite a un bitmap de un grupo disponible utilizado almacenarse en disco. Cada bit en el bitmap indica si un grupo en el rango de grupos correspondiente está libre o si todos están siendo utilizados. Uno indica que todos los grupos están en uso, y cero indica que por lo menos un grupo está libre. El bitmap tendrá tantos bits como rangos de grupos en el volumen.

Un rango de grupo es una región de grupos contiguos, si se define en términos del número de grupos. Los rangos de grupos sólo pueden ser múltiplos de los grupos en factores de dos-por ejemplo, puede ser uno, 2,4, 8,16 etc. se expresa en la entrada como uno logaritmo en base dos del número de grupos que comprende el rango de grupos. El tamaño del rango de grupo se fija en el momento del formato para todo el volumen. No se puede cambiar hasta que el volumen se vuelve a formatear.

Por ejemplo para un volumen de 1 GB con grupos de tamaño 16K, y tamaño del rango de grupo de = 1, el número de rangos de grupo será de 64K. entonces el tamaño del bitmap es de 64K/8 = 8 Kilobytes.

El tamaño del rango de grupo = 2, el número de rangos de grupo será de 32k.

El bitmaps existe únicamente como una entrada al directorio raíces del Tipo Oxl (refiérase a la próxima sección para una descripción del tipo de campo en una entrada de directorio). El formato del bitmaps en la entrada del directorio raíces es:

Nombre	Desbalance (byte)	Tamaño (bytes)	Descripción
Tipo	**0**	1	El valor es 1
Tamaño del rango del grupo	**1**	1	Granularidad del Bitmap. Esto se expresa como lograr ritmos del número de grupos. Por defecto es 0 --- indicando un rango del grupo de tamaño 2^0, por ejemplo. 1 grupo en dado caso hay 1 bit por grupo en el bitmap. Si el valor es 1, por ejemplo, hay 1 bit por 2 grupos contiguos..
Reservado	**2**	**20**	Reservado: debe ser cero
Tamaño del	**22**	**6**	Tamaño Del bitmap: expresado en la forma de little endian.
Grupo de inicio	**28**	**4**	Por grupo inicial para el bitmap: expresado en la forma de little endian.

Él FAT tendrá grupos de cadena que vinculan las asignaciones para los grupos que contienen él bitmap. Sin embargo recomendamos que él bitmap se asigne contiguo cuando él Filesystem se formatea. El bitmap contendrá 1's para los grupos que están asignados al propio bitmap.

LA ESTRUCTURA DEL DIRECTORIO

La estructura del directorio para FAT++ se ha cambiado como esta más abajo. Es incompatible con FAT32 -sin embargo las entradas al directorio son de 32-bytes cada uno. Los nombres cortos de archivo se han eliminado (pero es fácil añadirlos si se necesita). Las entradas al directorio principal sede tipifican. Hay 2 categorías de tipos:

1. **Tipos primarios.** Estas son entradas de directorio que tienen valores de tipo 0-127. El tipo primario 0 se refiere a una entrada normal visible al usuario. Estas se exponen ante el usuario directamente. Todos los demás tipos primarios (por ejemplo, tipos 1 a 127)-sólo pueden existir como entradas en el directorio raíz en FAT++. Cada tipo primario define un formato potencialmente diferente para la entrada al directorio. **Las implementaciones FAT++ no deben asumir el formato para ningún tipo primario que no entiendan.** Los bits 22-27 de una entrada de tipo primario sin embargo siempre indican el tamaño que la entrada describe y los bits 28-32 del tipo primario siempre indicaran el grupo inicial de la cadena de grupos que la entrada describe. Si el tamaño es cero, el grupo inicial debe también ser cero

Tipos secundarios: Una o más de las entradas al directorio pueden seguir a una entrada de tipo primario al directorio y están asociadas con la entrada de tipo primario al directorio. El tipo secundario extiende los metadatos contenidos en la entrada primaria al directorio. Deben seguir la entrada de tipo primario. Cada tipo secundario tiene un formato diferente para la entrada. **Las implementaciones FAT++ no deben asumir el formato para ningún tipo secundario que no entiendan.**

Entrada de Directorio Normal

Nombre	Des balance	Tamaño (bytes)	Descripción
Tipo	**0**	1	Tipo de entrada al directorio. Este formato describe las entradas tipo 0. 0-127 son tipo primario 0-es una entrada de directorio normal, que aparece enumerada 1-indica que es una entrada bitmap y se describe arriba 2-esto está reservado para uso como una entrada de clave maestra codificada para el volumen El resto están actualmente reservadas 128-255 son tipos secundarios. Estos extienden las entradas actuales de directorio. Todas las entradas de directorio del tipo primario pueden estar seguidas por 0 o más entradas de directorio de tipo secundario. Actualmente definimos los siguientes tipos secundarios: 129 es una entrada de nombre de archivo. Esto siempre seguirá a las entradas de directorio Tipo 0. Puede haber más de una entrada si el nombre del archivo no encaja en uno. La enumeración de directorio enumera las entradas tipo 0, recuperando nombres de archivos de las entradas de tipo secundario del tipo 129. Cuando una entrada de tipo primario se encuentra de nuevo, será el comienzo de una nueva entrada de directorio. El resto están actualmente reservadas

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Número de la entrada secundaria que sigue	1	1	Esto indica el número de entradas de directorio extendidas (por ejemplo del tipo 128-255) que sigue a esta entrada. Esto debe concordar con el número de entradas de directorio extendidas que sigue a esta entrada. Si no, la corrupción del FS será reconocida por Windows y un chkdsk se inicia para arreglar el problema (que podría requerir eliminar el directorio completamente .
Suma de chequeo	2	2	Suma de chequeo de 2-byte para esta entrada de directorio
Atributos	4	2	Atributos de archivo: ATTR_READ_ONLY 0x0001 ATTR_HIDDEN 0x0002 ATTR_SYSTEM 0x0004 ATTR_VOLUME_ID 0x0008 ATTR DIRECTORY 0x0010 ATTR ARCHIVE 0x0020 Los 2 bits superiores del byte de atributo bajo están reservados. El byte de atributo bajo esta en desfase 4 y el alto está en desfase 5. El byte de atributo alto representa los atributos extendidos para FAT+ que aun no están definidos También proponemos el soporte para ATTR_ENCRYPTED para los archivos/directorios. La codificación se discutirá más abajo en una sección separada
Creación de tiempo	6	8	Crea una marca de tiempo expresada en mili segundos 1, 1970 00:00:00 GMT. La marca de tiempo se expresa en pequeña endian.

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Última modificación de tiempo	14	8	Última marca de tiempo modificada expresada en milisegundos. Enero 1, 1970 00:00:00 GMT
Tamaño de archivo	22	6	Tamaño del archivo (48-bit). Esto permite un tamaño máximo de 256 terabites para un archivo, que es más grande que el volumen soportado por FAT+. Los 6 bytes estan en orden en pequeña endian.
Grupo de inicio	28	4	Numero de grupo de inicio. Esto se expresa en forma de pequeña endian: i.e. LSB tiene un desfase 12 y MSB tiene un desfase 15.

ENTRADA DE DIRECTORIO DE NOMBRE DE ARCHIVO

Siguiendo a cada entrada de directorio de tipo cero, debe seguir inmediatamente una o más entradas de directorio de tipo 128. Notese que esto es diferente a FAT32 donde las entradas LFN preceden a la entrada de nombre corto del archivo.. Estas representan al nombre del archivo para la entrada. Los nombres tienen una restricción de 255 caracteres -como FAT32- sin incluir el NULL que precede. Los caracteres son de 16-bit UNICODE.

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Tipo	0	1	128
Reservado	1	1	Reservado : debe ser cero.
Suma de	2	2	Suma de chequeo de 2-byte para esta entrada
Nombre-archivo	4	28	Caracteres de nombre de archivo UNICODE. Esto permite 14 caracteres UNICODE.

FAT++ ya no soporta las entradas '.' y '..' en un directorio. El Filesystem debe materializar esto a medida que se utiliza. Esto simplifica la lógica de renombre del directorio. Nótese también que los nombres de archivo corto se han eliminado. El archivo sólo tiene un nombre. Las entradas al directorio se pueden extender en el futuro a través de los tipos 129-255: sin embargo estos se deben colocar con la entrada de directorio de tipo primario. Por ejemplo, flujos/atributos extendidos nombrados para archivos se pueden implementar con entradas extendidas si es necesario.

REQUERIMIENTOS DE IMPLEMENTACIÓN

TRABAJANDO CON TIPOS DESCONOCIDOS

Todas las implementaciones FAT++ deben trabajar con las entradas del tipo de directorio descritas en este documento. Sin embargo es posible que en el futuro podamos añadir más tipos primarios como también extendidos (por ejemplo tanto en el rango 0-127 como también en el rango 128-255). Estos son los requerimientos para una implementación del nivel bajo FAT++ para trabajar fácilmente con extensiones:

1.) Cualquier entrada al directorio de tipo primario desconocida no debe ser visible al usuario. Segundo, estas entradas no se deben tocar, ni los grupos a los que ellas indican recogerse en la basura. Los bits 22-27 describen el tamaño de la región que la entrada describe, y los bits 28-32 describen el número del grupo de inicio (si lo hay). Entradas de tipo primario desconocidas no se deben determinar o actualizar de ninguna manera por las implementaciones.

2.) Cualquier entrada al directorio de tipo secundario se debe simplemente ignorar por una implementación. Cuando no archivos se elimina sin embargo-todas las entradas de tipo secundario se deben eliminar también.

RECONOCIMIENTO DEL SISTEMA DE ARCHIVO

Las implementaciones FAT++ compelen los requerimientos en forma estricta a diferencia de FAT32 que tienen muchas variantes. Las únicas extensiones permitidas son a través de tipos nuevos primarios/secundarios y estos siempre se deben revisar. Esto le permite a todas las implementaciones trabajar bien las unas con las otras.

SOPORTE DE CODIFICACIÓN

1. FAT++ apoya la codificación en el formato en-disco. Sin embargo no se requiere por parte de todas las implementaciones del sistema de archivos añadir soporte de codificación. La codificación en medios portátiles que se pueden mover entre Windows y dispositivos, por ejemplo, tendrán que ser compatibles con una llave simétrica producida a través de AES utilizando una clave de usuario que se puede almacenar en el medio. Proponemos la siguiente implementación para soporte de codificación: La clave de un usuario es secreta y se almacena en el volumen codificado. Esto es codificada utilizando la llave simétrica maestra.

La llave maestra simétrica se utilizará para codificar todos los archivos/datos. Nótese que ésta no es una llave por usuario, pero una llave global para ese volumen. Esto es codificación utilizando la clave. La codificación mutua asegura que si el medio se pierde la llave está segura y por lo tanto los datos no se pueden decodificar. Ahora considere el escenario:

1. El usuario inicia la codificación para el CF en su dispositivo. El dispositivo utiliza la clave para desbloquear el dispositivo y generar la llave simétrica. Si el dispositivo no soporta el PIN de desbloqueo, puede advertir al usuario para que ingrese una clave específica al medio. Ahora la llave simétrica y la clave están mutuamente codificadas y almacenadas en el medio en un archivo Tipo 2.
2. El usuario inserta la tarjeta CF en la máquina Windows.
3. El Filesystem reconoce que él volumen/directorio/archivo (como el caso sea) está codificada y advierte al usuario para una clave (esto puede ser indirectamente a través de una Cubierta, que advierte al usuario y suministra la clave al archivo del sistema).
4. Windows decodifica la llave simétrica en el volumen utilizando el pin/clave suministrada por el usuario.
5. Windows codifica la clave en texto claro en la memoria y la compara con la clave codificada almacenada en la tarjeta CF.
6. Si no son idénticas, el usuario obtiene un ERROR_ACCESS_DENIED.

De lo contrario la llave en la memoria se utilizará entonces para decodificar/codificar el contenido del volumen/archivo. Ahora el usuario inserta

de nuevo la tarjeta CF en el dispositivo.

7. El dispositivo comparara la forma codificada en-memoria de la copia de la clave de usuario del dispositivo -cuando el dispositivo se desbloquea -contra la clave en-disco. Si son idénticas, no le tiene que pedir al usuario una clave. De lo contrario, le pide al usuario una clave distinta para usar con ese volumen.
8. El dispositivo decodifica la llave maestra utilizando una clave en texto claro en-memoria. La llave única en-memoria ahora se utiliza para decodificar/codificar el contenido.

Nótese que aún no hemos propuesto ningún tamaño para el algoritmo/llave de codificación.

Una alternativa para el almacenamiento de la clave codificada en el disco es utilizar un valor único predeterminado (por ejemplo, una cookie) o una clave fraccionada de una vía (para más seguridad) que está codificada con la llave simétrica y almacenada en disco. Windows ahora decodifica la llave utilizando la clave del usuario, y una vez que obtiene la llave, la utiliza para hacer un fraccionamiento de una vía del texto claro de la clave del usuario en la memoria. Si concuerda con el valor en-disco, la llave se puede utilizar ahora para decodificar/codificar el contenido. De otra manera, Windows regresa un STATUS_ACCESS DENIED.

Los metadatos (por ejemplo en FAT, bitmaps, sectores de arranque y entradas al directorio) no se deben codificar para FAT++ puesto que el medio no se reconocerá. Únicamente el contenido se debe codificar por volumen o por archivo utilizando llaves maestros simétricas.

SOPORTE TFAT++

WinCE implementa una versión de FAT llamada TFAT, que asegura que el FAT no se corrompe. El campo NumberOfFats se utiliza para lograr la atomización en actualizaciones del FAT. Esencialmente el campo NumberOfFats se fija en 0 luego de actualizar el FAT secundario, y se fija de nuevo en 2 luego de actualizar el FAT primario. de tal manera que si el computador se cae NumberOfFats es 2, y el FAT primario se utiliza, y si es 0, el FAT secundario se copia al FAT primario como parte de la recuperación antes de cambiar de nuevo a 2.

El soporte para TFAT++ en Windows no utiliza el campo NumberOfFats. Utiliza el bit activo de FAT en los sectores de arranque. Para una descripción referirse al campo Banderas Extendidas en el sector de arranque anterior.

Un soporte más profundo incluiría la utilización del mismo algoritmo para actualizar FAT como TFAT. Nótese que en Windows, escribimos a través de todos los datos, de tal manera que en esencia esto nos protege únicamente cuando alguien intenta desenchufar un dispositivo de medios en caliente cuando hay indicadores abiertos. Este es un escenario interesante -el soporte de Windows asegurará que los dispositivos WinCE pueden continuar implementando TFAT++ para los dispositivos para los cuales tiene algún sentido (teléfonos celulares por ejemplo), y damos soporte a la recuperación en Windows.

FLUJOS CON NOMBRE

Los flujos con nombre pueden fácilmente estar soportados en FAT++ a través de un nuevo tipo de código. El código de tipo 129 se reservará con este propósito. Los flujos con nombre serían útiles en el sentido de que los metadatos que generalmente almacenan, cámaras por ejemplo, en un archivo separado o miniaturas almacenadas por Explorer, se pueden colocar en el mismo archivo, y copiar de los almacenes basados en Windows NTFS y dispositivos sin pérdida de fidelidad.

CONTROL DE ACCESO

El acceso a los archivos se puede restringir utilizando listas de control de Acceso (ACLs). FAT++ permite que los archivos tengan una lista de control de acceso a través de un tipo nuevo secundario. El código de tipo 130 se reserva para la ACLs de un archivo. Adicionalmente, un código de tipo primario-3, se reservará para servir como almacén central ACL.

Mientras que se han ilustrado y descrito implementaciones ilustrativas del invento, se podrá apreciar que varios cambios se pueden hacer sin alejarse del espíritu y el alcance del invento.

Las implementaciones del invento en las cuales se reivindica la propiedad o privilegio exclusivo se definen de la siguiente manera:

1. Todas las implementaciones descritas anteriormente.

FORMATO PARA SISTEMA DE ARCHIVOS PARA MEDIOS DE ALMACENAJE PORTÁTILES

CAMPO DEL INVENTO

El presente invento se relaciona con software y en particular con un protocolo para sistemas de archivos.

1. ANTECEDENTES DEL INVENTO FAT32 es el sistema de archivos preferido para los medios portátiles (tales como dispositivos basados en flash NAND que se utilizan comúnmente en cámaras, teléfonos celulares, etc.). La capacidad de estos dispositivos se espera que llegue a 32GB+ en los próximos años.

2. Se necesitan incrementos en la velocidad de asignación- FAT 32 tiene una tabla de búsqueda lineal.

3. El tamaño de los archivos en FAT 32 está limitado a 4GB.

4. La expansibilidad de FAT 32: FAT 32 no se puede extender fácilmente ahora puesto que todos los bits de entrada en el directorio se utilizan. El formato de nombre largo utiliza una máscara de bit que previene cualquier extensión adicional basada en máscaras de bit en las entradas del directorio.

DESCRIPCIÓN DETALLADA DE LA IMPLEMENTACIÓN PREFERIDA

DISEÑO FAT++

Un volumen FAT++ tiene el siguiente diseño.

Nombre	Desbalance (sector)	Tamaño (sectores)	Descripción
Sectores de arranque Y parámetros del BIOS Parámetro Bloque (BPB)	0	1	Este sector contiene el código de arranque de ésta partición y parámetros fundamentales del sistema de archivos que describen el tipo y el tamaño del sistema de archivos en ésta partición. El código para este sector se carga y se ejecuta en el Archivo Maestro de Arranque. Los cuatro últimos bits deben ser los OxAA550000.

Nombre	Desbalance (sector)	Tamaño (sectors)	Descripción

Extendido
Sector de arranque 1
	1	2	Este sector contiene un código adicional de arranque para la partición, y una estructura FSINFO
			con un desbalance Ox 1 E4:
			DWORD 0x37373937 – la firma de
			FSinfo (esto difiere de FAT32)
			DWORD EraseBlockSize – tamaño del bloque eliminado para dispositivos expresado en número de sectores. Para discos duros esto debe ser 1.
			// se pueden apoyar un total de 10 estructuras de parámetros diferentes OEM.
			struct {
			// MS assigned oem types
			DWORD OemParameterType;
			// OEM specific param structure
			UCHAR OemParameters[44]
			} OemParameterArea[10];
			Los OEMs por ejemplo, pueden añadir parámetros de ejecución para dispositivos flash. Estos se pueden añadir con un código de utilidades de formateo OEM específico en implementaciones OEM de especificaciones FAT.
			Estos serán ignorados por las implementaciones de Windows.
			UCHAR Reservado[20];
			Los 4 últimos bits deben ser OxAA550000 para ambos sectores .
			Primeros 4 bits del segundo sector
			deben ser 0x73739373

Nombre	Desbalance (sectores)	Tamaño (sectores)	Descripcion
Reservado I	3	3	Ceros
Arranque de respaldo	6	3	Copia exacta de los sectores 0,1,2.
Reservado2	9	NumReserved-9	Sectores reservados. El numero de sectores reservado esta en el BPB. Esta área se reserva para operaciones futuras del sistema operativo, y no se deben modificar o utilizar. NumReserved generalmente es *32* sectores, pero NO siempre sera 32.
Tabla de asignación de archivos (FAT)	NumReserved	Varia	Una tabla de entradas de 32-bit que define la asignación de archivos y directorios. Cada entrada representa ya sea un grupo asignado, un grupo no asignado, o un grupo no utilizable. Cuando se define un grupo asignado, el valor para esta entrada indica la entrada FAT para el siguiente grupo en el directorio de archivos, o el final de la cadena de asignaciones.
Grupo	...	Varia	El resto de la partición son datos de archivo, divididos en asignaciones con tamaños de grupo como se definen en el BPB.

NOTAS SOBRE LA REGIÓN DEL SISTEMA DE ARCHIVOS

La siguiente sección suministra notas sobre las regiones definidas por FAT++.

IMPORTANTE: todas los campos "reservados" más abajo están definidos actualmente como cero. Sin embargo, no se debe asumir que estos campos contengan este o cualquier otro valor. Las utilidades de disco y otros programas deben mirar o modificar éstos campos reservados. Es la responsabilidad de la utilidad preservar cualquier valor contenido en estos campos.

EL SECTOR DE ARRANQUE Y EL BPB

Nombre	Desbalance (byte)	Tamaño (bites)	Descripción
jmpBoot	0	3	Instrucciones jmp al código de "arranque"
OEMName	3	8	"MSWIN5.0"
Bytes por sector	11	2	Conteo de bits por sector.
Sectores por grupo	13	1	Número de sectores por unidad de asignación.
Sectores Reservados	14	2	Número de sectores reservados luego de extender el récord de arranque más el tamaño del récord de arranque y la extensión al récord de arranque. Éstos deben estar acondicionados en forma apropiada para qué el grupo de arranque se encuentre en el límite deseado -para dispositivos flash, eso seria aproximarla al límite de un bloque eliminado. Esto le permite al área de datos en el sistema de archivo al cual se le escribe frecuentemente-para comenzar en un bloque eliminado alineado en forma apropiada.
Números de FATs	16	1	2
Entradas Reservadas	17	2	Reservado
Cero	19	2	Debe ser cero (este es el antiguo campo del sector bs field).
Descriptor de medios	21	1	Definido por el OEM
Reservado	22	2	Debe ser cero
Sectores por pista	24	2	Sectores por pista para la interrupción 13h

Nombre	Desbalance	Tamaño (bytes)	Descripción
Conteo de cabezas	26	2	Numero de cabezas para el interrupt 13h
Conteo de sectores escondidos	28	4	Sectores i 1000 escondidos en la partición de tal manera que el iniciador de arranquen se puede hacer el int un 13h.
Conteo de sectores	32	4	Número total de sectores para todas las estructuras de datos y datos para este drive.
Sectores grandes	36	4	Conteo de sectores DWORD en una sola tabla de asignación de archivos (FAT).
Banderas extendidas	40	2	Bits 0-3 : número activo de FAT. Esto definitivamente indica que FAT esta activo. 1.) Si es 0, el FAT principal está en uso. 2.) Si es 1, el FAT secundario se considera el máster. Esto no puede utilizar las implementaciones TFAT++ para indicar el FAT consistente. 3.) Si es 2, entonces el volumen está en un estado "sucio" por ejemplo el FAT principal es el correcto pero un inspectora de consistencia fILESYSTEM tiene que ejecutar sede para asegurar que la estructura FATs/bitmaps/directorio es consistente. Bits 4-15: reservados, deben ser 0
Versión FS	42	2	El bit alto es el número mayor de revisión. El byte menor es el número menor de revisión. Este documento define la versión a 1:0. **Nota: Las utilidades de disco deben respetar éste campo y no operar en volúmenes con un número de versión mayor o menor que aquella para el cual fueron diseñados.**

Nombre	Desbalance (byte)	Tamaño (bytes)	Descripción
Primer Grupo de raíces	44	4	Número de grupo del primer grupo del directorio raíz. Puede ser 2, pero no necesariamente. **Nota: las utilidades de disco que cambian la localización del directorio raíces deben hacer un esfuerzo para colocar primer grupo del directorio raíces en el primer grupo no-dañado en el drive (i.e. En el grupo 2, al menos de que esté marcado como dañado).**
Sector de información de número FS	48	2	Número de sector de la estructura FS info. Generalmente 1. Nótese que habrá una copia de la estructura FSINFO en el BackupBoot, **Pero únicamente la copia indicada por éste campo se mantendrá actualizada- i.e. Tanto el récord de arranque primario y de respaldo indicarán el mismo sector FSINFO.**
Respaldo de arranque Número sectores del Record	50	2	Si no es cero indica el número de sector de una copia del récord de arranque. Generalmente 6.
Reservado	52	12	Reservado para expansiones futuras.
Número de Drive	64	1	Número de drive Int 13h (e.g. 80h).
Reservado	65	1	Reservado (utilizado por Windows NT).
Firma de arranque	66	1	Firma de arranque extendida (29h).
Volumen ID	67	4	Número serial del volumen
Etiqueta de Volumen	71	11	Etiqueta del Volumen

Nombre	Desbalance (byte)	Tamaño	Descripción
Tipo de Sistema de archivos	82	8	"FAT++"

El FAT++ estrictamente forzará el número de versión FS requerido, tipo de sistema de archivos y el tipo de partición idéntico a la descripción anterior. Cualquier inconformidad resultará en el no reconocimiento del volumen -o peor, reconocido potencialmente como FAT32.

TABLA DE ASIGNACIÓN DE ARCHIVOS(FAT)

La tabla de asignación de archivos está empacada dentistas de entradas de 32-bits que tienen mapeo con los grupos de datos uno-a-uno. Cada entrada se define como:

Desbalance (byte)	Tamaño (bits)	Nombre	Descripción
0	28	Valor del grupo	0x00000000 grupo no localizado 0x00000001 valor inválido 0x00000002 - *OxOFFFFFF7* localización en la cadena OxOFFFFFF7 marca de grupo dañado *(pero únicamente si no es parte de una cadena de grupos o no es un número de grupo legal.* *NOTA: esto quiere decir que este puede ser un número de grupo válido en la cadena de grupos!)* OxOFFFFFF8 - OxOFFFFFFF fin del Archivo
29	4	Reservado	Reservado para uso futuro.

No hay cambio en la estructura FAT del FAT32.

BITMAP DE ASIGNACIÓN

Adicionalmente al FAT, el formato también le permite a un bitmap de un grupo disponible utilizado almacenarse en disco. Cada bit en el bitmap indica si un grupo en el rango de grupos correspondiente está libre o si todos están siendo utilizados. Uno indica que todos los grupos están en uso, y cero indica que por lo menos un grupo está libre. El bitmap tendrá tantos bits como rangos de grupos en el volumen.

Un rango de grupo es una región de grupos contiguos, si se define en términos del número de grupos. Los rangos de grupos sólo pueden ser múltiplos de los grupos en factores de dos-por ejemplo, puede ser uno, 2,4, 8,16 etc. se expresa en la entrada como uno logaritmo en base dos del número de grupos que comprende el rango de grupos. El tamaño del rango de grupo se fija en el momento del formato para todo el volumen. No se puede cambiar hasta que el volumen se vuelve a formatear.

Por ejemplo para un volumen de 1 GB con grupos de tamaño 16K, y tamaño del rango de grupo de = 1, el número de rangos de grupo será de 64K. entonces el tamaño del bitmap es de 64K/8 = 8 Kilobytes.

El tamaño del rango de grupo = 2, el número de rangos de grupo será de 32k.

El bitmaps existe únicamente como una entrada al directorio raíces del Tipo Oxl (refiérase a la próxima sección para una descripción del tipo de campo en una entrada de directorio). El formato del bitmaps en la entrada del directorio raíces es:

Nombre	Desbalance (byte)	Tamaño (bytes)	Descripción
Tipo	0	1	El valor es 1
Tamaño del rango del grupo	1	1	Granularidad del Bitmap. Esto se expresa como lograr ritmos del número de grupos. Por defecto es 0 --- indicando un rango del grupo de tamaño 2^0, por ejemplo. 1 grupo en dado caso hay 1 bit por grupo en el bitmap. Si el valor es 1, por ejemplo, hay 1 bit por 2 grupos contiguos..
Reservado	2	20	Reservado: debe ser cero
Tamaño del	22	6	Tamaño Del bitmap: expresado en la forma de little endian.
Grupo de inicio	28	4	Por grupo inicial para el bitmap: expresado en la forma de little endian.

Él FAT tendrá grupos de cadena que vinculan las asignaciones para los grupos que contienen él bitmap. Sin embargo recomendamos que él bitmap se asigne contiguo cuando él Filesystem se formatea. El bitmap contendrá 1's para los grupos que están asignados al propio bitmap.

LA ESTRUCTURA DEL DIRECTORIO

La estructura del directorio para FAT++ se ha cambiado como esta más abajo. Es incompatible con FAT32 -sin embargo las entradas al directorio son de 32-bytes cada uno. Los nombres cortos de archivo se han eliminado (pero es fácil añadirlos si se necesita). Las entradas al directorio principal sede tipifican. Hay 2 categorías de tipos:

1. **Tipos primarios.** Estas son entradas de directorio que tienen valores de tipo 0-127. El tipo primario 0 se refiere a una entrada normal visible al usuario. Estas se exponen ante el usuario directamente. Todos los demás tipos primarios (por ejemplo, tipos 1 a 127)-sólo pueden existir como entradas en el directorio raíz en FAT++. Cada tipo primario define un formato potencialmente diferente para la entrada al directorio. **Las implementaciones FAT++ no deben asumir el formato para ningún tipo primario que no entiendan.** Los bits 22-27 de una entrada de tipo primario sin embargo siempre indican el tamaño que la entrada describe y los bits 28-32 del tipo primario siempre indicaran el grupo inicial de la cadena de grupos que la entrada describe. Si el tamaño es cero, el grupo inicial debe también ser cero

Tipos secundarios: Una o más de las entradas al directorio pueden seguir a una entrada de tipo primario al directorio y están asociadas con la entrada de tipo primario al directorio. El tipo secundario extiende los metadatos contenidos en la entrada primaria al directorio. Deben seguir la entrada de tipo primario. Cada tipo secundario tiene un formato diferente para la entrada. **Las implementaciones FAT++ no deben asumir el formato para ningún tipo secundario que no entiendan.**

Entrada de Directorio Normal

Nombre	Des balance	Tamaño (bytes)	Descripción
Tipo	0	1	Tipo de entrada al directorio. Este formato describe las entradas tipo 0. 0-127 son tipo primario 0-es una entrada de directorio normal, que aparece enumerada 1-indica que es una entrada bitmap y se describe arriba 2-esto está reservado para uso como una entrada de clave maestra codificada para el volumen El resto están actualmente reservadas 128-255 son tipos secundarios. Estos extienden las entradas actuales de directorio. Todas las entradas de directorio del tipo primario pueden estar seguidas por 0 o más entradas de directorio de tipo secundario. Actualmente definimos los siguientes tipos secundarios: 129 es una entrada de nombre de archivo. Esto siempre seguirá a las entradas de directorio Tipo 0. Puede haber más de una entrada si el nombre del archivo no encaja en uno. La enumeración de directorio enumera las entradas tipo 0, recuperando nombres de archivos de las entradas de tipo secundario del tipo 129. Cuando una entrada de tipo primario se encuentra de nuevo, será el comienzo de una nueva entrada de directorio. El resto están actualmente reservadas

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Número de la entrada secundaria que sigue	1	1	Esto indica el número de entradas de directorio extendidas (por ejemplo del tipo 128-255) que sigue a esta entrada. Esto debe concordar con el número de entradas de directorio extendidas que sigue a esta entrada. Si no, la corrupción del FS será reconocida por Windows y un chkdsk se inicia para arreglar el problema (que podría requerir eliminar el directorio completamente .
Suma de chequeo	2	2	Suma de chequeo de 2-byte para esta entrada de directorio
Atributos	4	2	Atributos de archivo: ATTR_READ_ONLY 0x0001 ATTR_HIDDEN 0x0002 ATTR_SYSTEM 0x0004 ATTR_VOLUME_ID 0x0008 ATTR DIRECTORY 0x0010 ATTR ARCHIVE 0x0020 Los 2 bits superiores del byte de atributo bajo están reservados. El byte de atributo bajo esta en desfase 4 y el alto está en desfase 5. El byte de atributo alto representa los atributos extendidos para FAT+ que aun no están definidos También proponemos el soporte para ATTR_ENCRYPTED para los archivos/directorios. La codificación se discutirá más abajo en una sección separada
Creación de tiempo	6	8	Crea una marca de tiempo expresada en mili segundos 1, 1970 00:00:00 GMT. La marca de tiempo se expresa en pequeña endian.

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Última modificación de tiempo	**14**	**8**	Última marca de tiempo modificada expresada en milisegundos. Enero 1, 1970 00:00:00 GMT
Tamaño de archivo	**22**	**6**	Tamaño del archivo (48-bit). Esto permite un tamaño máximo de 256 terabites para un archivo, que es más grande que el volumen soportado por FAT+. Los 6 bytes estan en orden en pequeña endian.
Grupo de inicio	**28**	**4**	Numero de grupo de inicio. Esto se expresa en forma de pequeña endian: i.e. LSB tiene un desfase 12 y MSB tiene un desfase 15.

ENTRADA DE DIRECTORIO DE NOMBRE DE ARCHIVO

Siguiendo a cada entrada de directorio de tipo cero, debe seguir inmediatamente una o más entradas de directorio de tipo 128. Notese que esto es diferente a FAT32 donde las entradas LFN preceden a la entrada de nombre corto del archivo.. Estas representan al nombre del archivo para la entrada. Los nombres tienen una restricción de 255 caracteres -como FAT32- sin incluir el NULL que precede. Los caracteres son de 16-bit UNICODE.

Nombre	Des balance (bits)	Tamaño (bits)	Descripción
Tipo	**0**	**1**	128
Reservado	**1**	**1**	Reservado : debe ser cero.
Suma de	**2**	**2**	Suma de chequeo de 2-byte para esta entrada
Nombre-archivo	**4**	**28**	Caracteres de nombre de archivo UNICODE. Esto permite 14 caracteres UNICODE.

FAT++ ya no soporta las entradas '.' y '..' en un directorio. El Filesystem debe materializar esto a medida que se utiliza. Esto simplifica la lógica de renombre del directorio. Nótese también que los nombres de archivo corto se han eliminado. El archivo sólo tiene un nombre. Las entradas al directorio se pueden extender en el futuro a través de los tipos 129-255: sin embargo estos se deben colocar con la entrada de directorio de tipo primario. Por ejemplo, flujos/atributos extendidos nombrados para archivos se pueden implementar con entradas extendidas si es necesario.

REQUERIMIENTOS DE IMPLEMENTACIÓN

TRABAJANDO CON TIPOS DESCONOCIDOS

Todas las implementaciones FAT++ deben trabajar con las entradas del tipo de directorio descritas en este documento. Sin embargo es posible que en el futuro podamos añadir más tipos primarios como también extendidos (por ejemplo tanto en el rango 0-127 como también en el rango 128-255). Estos son los requerimientos para una implementación del nivel bajo FAT++ para trabajar fácilmente con extensiones:

1.) Cualquier entrada al directorio de tipo primario desconocida no debe ser visible al usuario. Segundo, estas entradas no se deben tocar, ni los grupos a los que ellas indican recogerse en la basura. Los bits 22-27 describen el tamaño de la región que la entrada describe, y los bits 28-32 describen el número del grupo de inicio (si lo hay). Entradas de tipo primario desconocidas no se deben determinar o actualizar de ninguna manera por las implementaciones.

2.) Cualquier entrada al directorio de tipo secundario se debe simplemente ignorar por una implementación. Cuando no archivos se elimina sin embargo-todas las entradas de tipo secundario se deben eliminar también.

RECONOCIMIENTO DEL SISTEMA DE ARCHIVO

Las implementaciones FAT++ compelen los requerimientos en forma estricta a diferencia de FAT32 que tienen muchas variantes. Las únicas extensiones permitidas son a través de tipos nuevos primarios/secundarios y estos siempre se deben revisar. Esto le permite a todas las implementaciones trabajar bien las unas con las otras.

SOPORTE DE CODIFICACIÓN

1. FAT++ apoya la codificación en el formato en-disco. Sin embargo no se requiere por parte de todas las implementaciones del sistema de archivos añadir soporte de codificación. La codificación en medios portátiles que se pueden mover entre Windows y dispositivos, por ejemplo, tendrán que ser compatibles con una llave simétrica producida a través de AES utilizando una clave de usuario que se puede almacenar en el medio. Proponemos la siguiente implementación para soporte de codificación: La clave de un usuario es secreta y se almacena en el volumen codificado. Esto es codificada utilizando la llave simétrica maestra.

La llave maestra simétrica se utilizará para codificar todos los archivos/datos. Nótese que ésta no es una llave por usuario, pero una llave global para ese volumen. Esto es codificación utilizando la clave. La codificación mutua asegura que si el medio se pierde la llave está segura y por lo tanto los datos no se pueden decodificar. Ahora considere el escenario:

1. El usuario inicia la codificación para el CF en su dispositivo. El dispositivo utiliza la clave para desbloquear el dispositivo y generar la llave simétrica. Si el dispositivo no soporta el PIN de desbloqueo, puede advertir al usuario para que ingrese una clave específica al medio. Ahora la llave simétrica y la clave están mutuamente codificadas y almacenadas en el medio en un archivo Tipo 2.
2. El usuario inserta la tarjeta CF en la máquina Windows.
3. El Filesystem reconoce que él volumen/directorio/archivo (como el caso sea) está codificada y advierte al usuario para una clave (esto puede ser indirectamente a través de una Cubierta, que advierte al usuario y suministra la clave al archivo del sistema).
4. Windows decodifica la llave simétrica en el volumen utilizando el pin/clave suministrada por el usuario.
5. Windows codifica la clave en texto claro en la memoria y la compara con la clave codificada almacenada en la tarjeta CF.
6. Si no son idénticas, el usuario obtiene un ERROR_ACCESS_DENIED.

De lo contrario la llave en la memoria se utilizará entonces para decodificar/codificar el contenido del volumen/archivo. Ahora el usuario inserta

de nuevo la tarjeta CF en el dispositivo.

7. El dispositivo comparara la forma codificada en-memoria de la copia de la clave de usuario del dispositivo -cuando el dispositivo se desbloquea -contra la clave en-disco. Si son idénticas, no le tiene que pedir al usuario una clave. De lo contrario, le pide al usuario una clave distinta para usar con ese volumen.
8. El dispositivo decodifica la llave maestra utilizando una clave en texto claro en-memoria. La llave única en-memoria ahora se utiliza para decodificar/codificar el contenido.

Nótese que aún no hemos propuesto ningún tamaño para el algoritmo/llave de codificación.

Una alternativa para el almacenamiento de la clave codificada en el disco es utilizar un valor único predeterminado (por ejemplo, una cookie) o una clave fraccionada de una vía (para más seguridad) que está codificada con la llave simétrica y almacenada en disco. Windows ahora decodifica la llave utilizando la clave del usuario, y una vez que obtiene la llave, la utiliza para hacer un fraccionamiento de una vía del texto claro de la clave del usuario en la memoria. Si concuerda con el valor en-disco, la llave se puede utilizar ahora para decodificar/codificar el contenido. De otra manera, Windows regresa un STATUS_ACCESS DENIED.

Los metadatos (por ejemplo en FAT, bitmaps, sectores de arranque y entradas al directorio) no se deben codificar para FAT++ puesto que el medio no se reconocerá. Únicamente el contenido se debe codificar por volumen o por archivo utilizando llaves maestros simétricas.

SOPORTE TFAT++

WinCE implementa una versión de FAT llamada TFAT, que asegura que el FAT no se corrompe. El campo NumberOfFats se utiliza para lograr la atomización en actualizaciones del FAT. Esencialmente el campo NumberOfFats se fija en 0 luego de actualizar el FAT secundario, y se fija de nuevo en 2 luego de actualizar el FAT primario. de tal manera que si el computador se cae NumberOfFats es 2, y el FAT primario se utiliza, y si es 0, el FAT secundario se copia al FAT primario como parte de la recuperación antes de cambiar de nuevo a 2.

El soporte para TFAT++ en Windows no utiliza el campo NumberOfFats. Utiliza el bit activo de FAT en los sectores de arranque. Para una descripción referirse al campo Banderas Extendidas en el sector de arranque anterior.

Un soporte más profundo incluiría la utilización del mismo algoritmo para actualizar FAT como TFAT. Nótese que en Windows, escribimos a través de todos los datos, de tal manera que en esencia esto nos protege únicamente cuando alguien intenta desenchufar un dispositivo de medios en caliente cuando hay indicadores abiertos. Este es un escenario interesante -el soporte de Windows asegurará que los dispositivos WinCE pueden continuar implementando TFAT++ para los dispositivos para los cuales tiene algún sentido (teléfonos celulares por ejemplo), y damos soporte a la recuperación en Windows.

FLUJOS CON NOMBRE

Los flujos con nombre pueden fácilmente estar soportados en FAT++ a través de un nuevo tipo de código. El código de tipo 129 se reservará con este propósito. Los flujos con nombre serían útiles en el sentido de que los metadatos que generalmente almacenan, cámaras por ejemplo, en un archivo separado o miniaturas almacenadas por Explorer, se pueden colocar en el mismo archivo, y copiar de los almacenes basados en Windows NTFS y dispositivos sin pérdida de fidelidad.

CONTROL DE ACCESO

El acceso a los archivos se puede restringir utilizando listas de control de Acceso (ACLs). FAT++ permite que los archivos tengan una lista de control de acceso a través de un tipo nuevo secundario. El código de tipo 130 se reserva para la ACLs de un archivo. Adicionalmente, un código de tipo primario-3, se reservará para servir como almacén central ACL.

Mientras que se han ilustrado y descrito implementaciones ilustrativas del invento, se podrá apreciar que varios cambios se pueden hacer sin alejarse del espíritu y el alcance del invento.

Las implementaciones del invento en las cuales se reivindica la propiedad o privilegio exclusivo se definen de la siguiente manera:

1. Todas las implementaciones descritas anteriormente.

Rotulo MS No. 312054.02
(Rotulo OC No. MSFT-124795)

ASIGNACIÓN DE PATENTE

Nosotros, Ravisankar V. Pudipeddi, Vishal V. Ghotge, Sarosh C. Havewala, Ravinder S. Thind, Mark J. Zbikowski ("Cesionistas 1"), hemos inventado ("INVENTO 1") divulgado y/o reivindicado en la solicitud de patente llamada " FORMATO PARA SISTEMAS DE ARCHIVOS PARA MEDIOS DE ALMACENAJE PORTATILES" (solicitud 1), que se presento el 17 de Diciembre 2004 y le fue otorgado el numero provisional de solicitud No. 60/637,407 y Ravisankar V. Pudipeddi, Vishal V. Ghotge, Sarosh C. Havewala, Ravinder S. Thind, Mark J. Zbikowski ("Cesionistas 2") han inventado el tema ("INVENTO 2") divulgado y/o reivindicado en la solicitud llamada "SISTEMA DE ARCHIVOS EXTENSIBLE" (SOLICITUD 2) que se presento el 16 de Septiembre y se le dio el numero de solicitud US No. 11/229,485:

La corporación Microsoft, una corporación de Washington, en su propio nombre y el de sus sucesores y cesionarios, tiene derecho a, y está deseoso de adquirir la totalidad y exclusividad de los derechos, títulos e interés en el INVENTO 1 y INVENTO 2 y la APLICACIÓN 1 y la APLICACIÓN 2 (y todas las demás aplicaciones de patentes que se deriven de esta, tales como aplicaciones de continuación, en y para los Estados Unidos y sus territorios, y todos los países extranjeros ("DERIVADOS DE LA SOLICITUD");

En consideración buena y valiosa, en el recibo que el CESIONISTA 1 y CESIONISTA 2 reconoce por la presente, los CESIONISTA 1 y CESIONISTA 2 (y derivados de la solicitud)venden, asignan y transfieren hacia el CESIONARIO, el derecho completo, título e interés en el INVENTO 1 e INVENTO 2 y la SOLICITUD 1 y ; SOLICITUD 2 y derivados de la solicitud)

El CESIONISTA 1 y CESIONISTA 2 se compromete a ejecutar todos los instrumentos y documentos requeridos para el procesamiento de la SOLICITUD 1 y SOLICITUD 2(y derivados de la solicitud), para el litigio en cuanto a las cartas de patente que se deriven de ella, y con el propósito de proteger y perfeccionar el título a la SOLICITUD 1 y SOLICITUD 2.

Fecha	
10-24-05	Ravisankar V. Pudipeddi (Firma)
10-24-05	Vishal V. Ghotge (Firma)
10-24-05	Sarosh C. Havewala (Firma)
10-24-05	Ravinder S. Thind (Firma)
10-24-05	Mark J. Zbikowski (Firma)

120

DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS

OFICINA DE PATENTE Y MARCA REGISTRADAS
SECRETRARIO Y COMISIONADO ASISTENTE DE
PATENTES Y MARCAS REGISTRADAS

PTAS 500056849A

OCTUBRE 24, 2005
MAURICIO URIBE,ESQ.
1420 FIFTH AVENUE, SUITE 2800
SEATTLE, WA 98101

OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS
NOTIFICACION DEL RECORDATORIO DEL DOCUMENTO DE ASIGNACION

EL DOCUMENTO ADJUNTO HA SIDO GRABADO POR LA DIVISIÓN DE LA OFICINA DE PATENTES Y MARCAS REGISTRADAS DE E.U. UNA COPIA COMPLETA EN MICROFILME ESTÁ DISPONIBLE EN EL CUARTO DE BÚSQUEDA DE LAS ASIGNACIONES EN LA CINTA Y EL NÚMERO DE CUADRO QUE SE CITAN ABAJO.

POR FAVOR REVISE TODA LA INFORMACIÓN CONTENIDA EN ESTA NOTIFICACIÓN. LA INFORMACIÓN CONTENIDA EN ESTA NOTIFICACIÓN DE ARCHIVO REFLEJA LOS DATOS PRESENTES EN EL SISTEMA DE ASIGNACIÓN DE PATENTES Y MARCAS REGISTRADAS SI USTED ENCUENTRA ALGÚN ERROR O TIENE ALGUNA PREGUNTA CONCERNIENTE A ESTA NOTIFICACIÓN, PUEDE LLAMAR AL EMPLEADO CUYO NOMBRE APARECE EN ESTA NOTIFICACIÓN AL NÚMERO 571-272-3350. POR FAVOR ENVIE LAS SOLICITUDES PARA CORRECCIÓN A: OFICINA DE PATENTES Y MARCAS REGISTRADAS DE E.U, CORREO:ASSIGNMENT SERVICES DIVISION, P.O. BOX 1450, ALEXANDRIA, VA 22313.

FECHA DE GRABACION: 10/24/2005 CINTA/CUADRO 016677/0359
NUMERO DE PAGINAS: 3

RESUMEN: ASIGNACIÓN DE INTERÉS AL CESIONISTA (VER DOCUMENTO PARA DETALLES)

ROTULO NO. MSFT-1-24795

CESIONISTA:
PUDIPEDDI, RAVISANKAR V. FECHA DOC: 10/24/2005

CESIONISTA:
GHOTGE, VISHAL V. FECHA DOC: 10/24/2005

CESIONISTA:
HAVEWALA, SAROSH C. FECHA DOC: 10/24/2005

CESIONISTA:
THIND, RAVINDER S. FECHA DOC: 10/24/2005

CESIONISTA:
ZBIKOWSKI, MARK J. FECHA DOC: 10/24/2005

121

016677/0359 PAGINA 2

CESIONARIO:
MICROSOFT
CORPORATION
ONE MICROSOFT WAY
REDMOND, WASHINGTON 98052

NUMERO DE SERIAL: 111229485 FECHA DE ENTREGA: 09/16/2005
NUMERO DE PATENTE: FECHA DE EMICIÓN:

TITULO: SISTEMA DE ARCHIVOS EXTENSIBLE

KIMBERLY WHITE, REVISOR
DIVISIÓN DE ASIGNACIONES
OFICINA DE ARCHIVOS PUBLICOS

ASIGNACIÓN DE PATENTE	
Versión electrónica v1.1	10/24/2005
Estilo de hoja versión v1.1	500056849

TIPOS DE SUMISIÓN: NUEVA ASIGNACIÓN

NATURALEZA DE LA ENTREGA: ASIGNACIÓN

DATOS DE LA PARTE QUE ENTREGA

Nombre	Fecha ejecución
RAVISANKAR V. PUDIPEDDI	05/16/2005
VISHAL V. GHOTGE	05/20/2005
SAROSH C. HAVEWALA	05/16/2005
RAVINDER S. THIND	05/16/2005
MARK J. ZBIKOWSKI	05/16/2005

DATOS DE LA PARTE QUE RECIBE

Nombre:	Corporación Microsoft
Dirección:	One Microsoft Way
Ciudad	Redmont
Estado/país:	WASHINGTON
C.P.:	98052

NÚMEROS DE PROPIEDAD TOTAL: 1

Tipo De Propiedad	Número
Solicitud numero:	11229485

DATOS DE CORRESPONDENCIA
faxe: (206) 224-0779
la correspondencia se enviará a través del correo de EU cuando el fax no se puede enviar
teléfono: 206.695.1653
e-mail: EFILING@COJK.com
nombre de correspondencia: Mauricio A Uribe. Esq.
Línea de Dirección 1 1420 FifthAvenue, Suite 2800
Línea de Dirección 4 Seattle, WASHINGTON 98101

No. de rotulo del Abogado: MSFT-1-24795
Nombre del presentador Mauricio A Uribe. Esq.

Total anexos:1
Fuente =24795 asignación # page1.tif

SUPERINDUSTRIA Y COMERCIO Folios: 124
Radicacion : 05116723 00000000
Fecha (AMD): 2005-11 17 16 10 59
Tramite 002 PATENTES D 1 REGISTRO/ 411 PRESENTAC
Dependencia 2020 DIVISION DE NUEVAS CREACIONES

125
124

REQUISITOS NECESARIOS PARA PRESENTACION Y AGILIZACION DEL TRAMITE

		USUARIO	RADICADOR
PATENTE DE INVENCIÓN	MODELO DE UTILIDAD	( )	( )
- Indicación de que se solicita la concesión de una patente.		( )	( )
- Datos de identificación del solicitante o de la persona que se presenta la solicitud.		( )	( )
- Descripción de la invención.		( )	( )
- Dibujos de ser estos pertinentes.		( )	( )
- Comprobante de Pago de las tasas establecidas.			
COMPLETA ( )	INCOMPLETA	( )	( )
DISEÑO INDUSTRIAL ( )			
- Indicación de que se solicita el registro de Diseño Industrial		( )	( )
- Datos de identificación del solicitante o de la persona que presenta		( )	( )
- la solicitud			
- Representación gráfica o fotografica del Diseño Industrial o muestra del Material que incorpora el diseño		( )	( )
Comprobante de pago de las tasas establecidas		( )	( )
COMPLETA ( )	INCOMPLETA	( )	( )
ESQUEMA DE TRAZADO ( )			
- Indicación de que solicita el registro de un Esquema de trazado		( )	( )
- Datos de identificación del solicitante o de la persona que presenta La solicitud		( )	( )
Representación grafica del esquema de trazado		( )	( )
Comprobante de pago de las tasas establecidas		( )	( )
COMPLETA ( )	INCOMPLETA	( )	( )

Firma Responsable

Nov 17 de 2005

Fecha

1265
115

PROTOCOLO

Expediente 05 - 116723

SUPERINDUSTRIA Y COMERCIO
SISTEMA DE REGISTRO

MANTENIMIENTO DE PROTOCOLOS

Numero : 16069
Fecha : 04/08/1995
Conferido: MICROSOFT CORPORATION
Pais : US ESTADOS UNIDOS DE AMERICA
Ciudad : 234 REDMOND, WASHINGTON
Represent: S Sustituir : S Desistir : S
No. Radicacion : 95 34920 Clase: 0 Seev: 0 Seac: 0

scncia	Codigo	Ctr	Nombre
1	77	A	CARDENAS NAVAS DARIO
2	3653		CARDENAS CABALLERO EDUARDO
3	5048		CARDENAS MARTINEZ BERNARDO

Encontrados (1/1) registro(s)

Sede Centro: Carrera 13 No. 27-00 Pisos 2, 5, 7 y 10
Conmutador: 3 820840 Fax: 3 505220
Sede CAN: Tr. 40A No. 38-50 3153265 al 69
Fax: 3 505220. Línea 9800-910 165
Web: www.sic.gov.co e-mail: info@sic.gov.co
Bogotá D.C. - Colombia

REPUBLICA DE COLOMBIA

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

127
126

Bogotá,
2020

Doctor(a)
CARDENAS NAVAS DARIO
Apoderado(a) y/o Representante de
MICROSOFT CORPORATION

REFERENCIA Radicación No. 5 116723
Trámite 2
Evento 1
Actuación 430
Oficio No. 11311

Como resultado del examen de forma, realizado con fundamento en el artículo 38 de la Decisión 486 de la Comisión de la Comunidad Andina, se le comunica que:

1. Se recuerda al interesado que debe presentar, dentro de los dieciséis meses siguientes contados a partir de la fecha de presentación de la solicitud cuya prioridad se invoca, copia de la misma certificada por la autoridad que la expidió y, en los casos en que haya lugar, la traducción simple del capítulo reivindicatorio. El incumplimiento de éste plazo, acarreará la pérdida de la prioridad invocada.

2. Debe allegar copia del documento en que consta la cesión del derecho a la patente otorgado por el(los) inventor(s) al(los) solicitante(s) o a su(s) causante(s) (art. 26-k,Dec 486/2000).

3. Para efectos de la publicación en la gaceta se recomienda allegar el extracto diligenciado.

En cumplimiento del artículo 39 de la Decisión 486, se le requiere para que dentro del término de dos (2) meses siguientes ala fecha de notificación del presente oficio, complete los requisitos anteriormente enunciados. En caso de no dar cumplimiento al presente requerimiento, la solicitud se considerará abandonada y perderá su prelación.

Notifíquese el presente oficio conforme a lo dispuesto en el numeral 5.2, literal e), del capítulo quinto del título primero de la Circular Unica No.10 de 2001.

ALIX CARMENZA CESPEDES DE VERGEL
Jefe de la división de Nuevas Creaciones

El anterior requerimiento fue notificado por fijación en lista de fecha 01 DIC. 2005
adpa1